



# USER GUIDE 4



Copyright © 2000-2004 Definiens Imaging. All rights reserved. Made in Germany.

The information contained in this document is the exclusive property of Definiens Imaging. This work is protected under German copyright law and the copyright laws of the given countries of origin and applicable international laws, treaties and/or conventions. No part of this work may be reproduced in any form or by any means, electronic or mechanical, including photocopying or recording, or by any information storage or retrieval system, except as expressly permitted in writing to Definiens Imaging. All requests should be sent to the address written below.

The information contained in this document is subject to change without notice and should not be construed as a commitment by Definiens Imaging. Definiens Imaging assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used in accordance with the terms of such license.

eCognition is a protected software title.

The absence of a product or service name or logo belonging to Definiens Imaging anywhere in the text of this site does not constitute a waiver of Definiens Imaging's trademark or other intellectual property rights concerning that name or logo. All other products and brand names are trademarks and/or registered trademarks of their respective owners.

### **Authors**

Martin Baatz, Ursula Benz, Seyed Dehghani, Markus Heynen, Astrid Höltje, Peter Hofmann, Iris Lingfelder, Matthias Mimler, Malte Sohlbach, Michaela Weber, Gregor Willhauck

### **Contact**

Definiens Imaging GmbH  
Trappentreustrasse 1  
80339 München  
Germany

Tel. +49-89-23 11 80-0

Fax +49-89-23 11 80-90

eMail: [ecognition@definiens-imaging.com](mailto:ecognition@definiens-imaging.com)

Homepage: [www.definiens-imaging.com](http://www.definiens-imaging.com)

# TABLE OF CONTENTS

<b>Table of contents.....</b>	<b>4</b>
<b>1 Introduction .....</b>	<b>12</b>
How to use this user guide.....	13
Background .....	14
Gaps between the new generation of remote sensing systems and GIS .....	14
eCognition overview.....	15
Segmentation .....	16
Classification .....	17
Features for classification .....	17
Information about image objects and classification .....	18
Multi source data fusion.....	18
Vectorization .....	19
Export of results.....	19
What's new in eCognition Professional 4.0 .....	21
Large Data Handling (LDH) Version .....	21
No Data Values .....	21
Save subsets upon import.....	21
Improved manual editing of objects.....	22
Object Table .....	22
Options dialog .....	22
Further improvements and changes .....	23
<b>2 Installing eCognition .....</b>	<b>24</b>
Installing eCognition with a hardlock.....	25
Installing eCognition with a local hardlock.....	25
Installing eCognition with a server hardlock .....	26
The Hardlock Utility Software.....	28
Initialize Hardlock Server module with alf file .....	28
Export license information .....	28
Update License .....	30
Server Hardlock, Update from clipboard .....	32
Server Hardlock, Update by File .....	32



Generate Report.....	33
Diagnostix.....	34
About.....	34
<b>The Aladdin DiagnostiX Tool .....</b>	<b>35</b>
Checking for a Hardlock Module.....	35
Creating reports.....	36
Setting environment variables .....	36
Environment Variables .....	37
<b>Hardlock and License Troubleshooting .....</b>	<b>40</b>
Common errors.....	41
<b>3 Take the Plunge.....</b>	<b>43</b>
<b>Getting a first glimpse of eCognition .....</b>	<b>43</b>
Creating a new project and loading the raster data.....	44
Changing the color composition of the displayed image .....	45
Creating image objects.....	46
Obtaining information about image objects.....	48
Loading a class hierarchy.....	49
Declaring sample objects.....	51
Having a look at the descriptions of child classes.....	54
Classifying the image using class-related features .....	57
Summary.....	58
<b>4 Concepts &amp; Methods .....</b>	<b>59</b>
<b>A. The approach and basic concepts .....</b>	<b>60</b>
What is object oriented image analysis? .....	60
Basic aspects in image interpretation .....	62
Segmentation .....	65
Classification – classifiers and methods .....	68
Fuzzy classification systems .....	69
<b>B. How is object oriented image analysis realized in eCognition?.....</b>	<b>74</b>
Multiresolution segmentation of image objects.....	76
Handling of vector information .....	89
Fuzzy classification in eCognition.....	94
The classification process .....	104
Feature overview .....	106
Features and terms for the fuzzy class description .....	110
Classification-based segmentation and correction of image objects .....	148

Multisource data fusion .....	152
Classification evaluation in eCognition .....	155
Aspects of human perception .....	163
References .....	166
<b>5 Functional Guide.....</b>	<b>169</b>
How to use eCognition.....	169
Overview.....	170
eCognition workflow .....	170
Importing and Exporting.....	172
Creating new eCognition projects .....	172
Importing image layers .....	173
Adding additional layers to a running eCognition project .....	178
Assigning layer aliases .....	179
Defining no data values.....	179
Loading and saving eCognition projects .....	181
Importing and exporting eCognition files .....	181
Export object shapes .....	185
Navigation and Visualization .....	189
Navigating within the scene .....	189
Navigating within the image object hierarchy.....	190
Navigating within the groups hierarchy .....	191
View settings .....	192
Edit highlight color .....	197
Multiwindow functionality.....	197
Image Object Generation I: Multiresolution Segmentation .....	198
Multiresolution segmentation .....	198
Layer weights .....	199
Level.....	200
Scale parameter.....	201
Composition of the homogeneity criterion.....	201
Type of neighborhood .....	202
Use obsolete (V2.1) segmentation .....	203
Deleting segmented levels .....	203
How to produce image objects suited to your classification purpose .....	203
Influence of bit depth of raster data .....	204
Working with image layers of different radiometric resolution .....	204

Constructing an image object hierarchy with more than one level .....	204
Classifying differently scaled objects .....	205
Using different data for segmentation and classification .....	206
Generating sub-objects for line analysis .....	207
Classification based multiresolution segmentation .....	207
Scale parameter analysis .....	207
<b>Working with Polygons .....</b>	<b>211</b>
Creating polygons .....	211
Displaying polygons .....	212
Working with polygon and skeleton features .....	213
Displaying skeleton .....	214
Deleting polygons .....	214
<b>Information on Image Objects and Features .....</b>	<b>215</b>
Information on single image objects .....	215
Features .....	217
Classification .....	217
Class evaluation .....	218
Analyzing and comparing image object attributes .....	218
Feature view .....	218
2D feature space plot .....	220
Sample editor .....	220
<b>Classification Introduction .....</b>	<b>222</b>
Introduction .....	222
Inserting a new class .....	224
General class settings .....	225
Editing the class description .....	226
Inserting an expression .....	226
Moving expressions and editing the hierarchy of logical operations .....	227
Inverting expressions .....	228
Editing an expression .....	228
Deleting an expression .....	228
Deleting a class .....	228
Deleting a class hierarchy .....	229
<b>Classification Basics .....</b>	<b>230</b>
Nearest neighbor .....	230
Inserting nearest neighbor or standard nearest neighbor .....	231
Defining the feature space .....	231
Inserting sample objects manually .....	232
Inserting sample objects automatically using the TTA mask .....	234

Storing samples as a TTA mask .....	236
Deleting sample objects .....	238
Fuzzy rule base .....	238
Membership function type .....	239
Value range .....	241
Maximum and minimum values .....	241
Generating membership functions automatically .....	242
Operators .....	242
Hierarchy of logical operators .....	244
Features and expressions .....	244
Object features .....	244
Class-related features .....	245
Feature distance .....	246
Object oriented texture analysis .....	247
Think ahead when using class-related features .....	247
Similarities .....	248
The classification process .....	249
Classification without class-related features .....	249
How to handle class-related features .....	250
Simple classification with class-related features .....	251
Deleting classification results .....	252
<b>Classification Advanced .....</b>	<b>253</b>
The class hierarchy .....	253
Inheritance .....	253
Groups .....	255
The interrelations between inheritance and groups .....	255
Editing the hierarchical structure .....	257
Multiple inheritance, multiple membership .....	257
Basic strategies for creating class hierarchies .....	258
Overview: how to acquire information .....	259
Assessing the quality of a new sample .....	260
Finding the features that best separate the classes .....	261
2D feature space plot .....	265
Getting the spectral histogram of an image layer .....	265
Fast nearest neighbor classification by manual training: click and classify .....	265
Using fuzzy rules to formulate classification concepts .....	267
Differentiating classes using contextual information (class-related features) .....	267
Using class-related features with nearest neighbor .....	269
Creating customized features .....	272
Arithmetic features .....	273
Relational features .....	274
Comparison of arithmetic and relational features .....	275

Image object information dialog .....	276
Object table .....	277
Classification .....	277
Class evaluation .....	278
<b>Image Object Generation II:</b>	
<b>Classification-based Segmentation/Refinement.....</b>	<b>281</b>
Definition of structure groups .....	281
Object generation.....	282
Classification-based fusion of image objects.....	283
Classification-based multi-resolution segmentation .....	287
Classification-based object cut .....	289
<b>Accuracy Assessment and Statistics .....</b>	<b>291</b>
The Accuracy Assessment dialog box.....	291
Best classification result .....	293
Error matrix based on TTA mask .....	294
Image layer histograms .....	298
<b>Automation of Operations.....</b>	<b>299</b>
The protocol tool bar.....	299
Recording protocols.....	299
Executing protocols .....	300
Editing protocols.....	301
<b>Manual Editing of Image Objects .....</b>	<b>302</b>
Selecting objects to fuse or classify .....	302
Manual object fusion .....	302
Manual classification.....	303
Manual object cut.....	305
<b>6 User Interface.....</b>	<b>307</b>
<b>Overview .....</b>	<b>307</b>
List of eCognition dialogs.....	307
<b>Menu .....</b>	<b>312</b>
<b>Dialogs .....</b>	<b>319</b>
2D Feature Space Plot .....	319
Accuracy Assessment.....	320
Apply Standard Nearest Neighbor to Classes .....	322
Apply TTA Mask to Level.....	322

Auto Cut Parameters .....	323
Class Description .....	324
Class Hierarchy .....	325
Classification-based Segmentation .....	327
Classification Settings .....	328
Colors .....	329
Conversion Table .....	329
Create Customized Feature .....	330
Create Polygons .....	332
Create Project .....	332
Create TTA Mask from Level .....	334
Customize .....	334
Define Brightness .....	336
Delete Classification .....	337
Delete Level .....	338
Delete Samples of Selected Classes .....	338
eCognition Hardlock Utility .....	339
Edit Standard Nearest Neighbor Feature Space .....	340
Edit Highlight Colors .....	341
Edit Layer Mixing .....	341
Edit Minimum Membership Value .....	343
Export Classification .....	343
Edit Parametrized Feature .....	344
Export Image Objects .....	345
Export Image Object Shapes .....	346
Export Scale Parameter Analysis .....	347
Feature Space Optimization .....	347
Feature Statistics .....	350
Feature View .....	351
Help Keyboard .....	352
Image Layer Histograms .....	352
Image Object Information .....	353
Input Mode .....	354
Insert Expression .....	355
Layer Properties .....	355
Membership Function .....	356
Membership Function Parameters .....	358
Message Console .....	358
Multiresolution Segmentation .....	359
Object Table (not available in the LDH version) .....	360
Options .....	361
Pan Window .....	362
Project Info .....	362

Protocol Editor .....	362
Sample Editor .....	363
Sample Navigation .....	365
Sample Selection Information .....	365
Scale Parameter Analysis .....	367
Select Layer .....	369
Select Level .....	369
Select Displayed Features .....	369
Select Features for Statistic .....	370
Select ID Column .....	370
Select Operator for Expression .....	371
Select Single Feature .....	372
Set Nearest Neighbor Function Slope .....	372
Statistics .....	373
Subset Selection .....	374
System Info .....	375
User Information .....	375
View Settings .....	375
<b>7 Guided Tours .....</b>	<b>378</b>
<b>Tour 1: Landsat TM subset of Orange County (California, USA) .....</b>	<b>379</b>
<b>Tour 2: Analysis of the degree of urban impervious surface .....</b>	<b>402</b>
<b>Tour 3: High Resolution Aerial Scan .....</b>	<b>432</b>
<b>Tour 4: Radar image of a tropical rain forest in Kalimantan (Indonesia) .....</b>	<b>460</b>
<b>Tour 5: Aerial Photo and Lidar Surface Model of Odense (Denmark) .....</b>	<b>470</b>

# 1 INTRODUCTION

## Welcome to eCognition's fascinating world of object oriented image analysis!

eCognition follows an object oriented approach towards image analysis. It provides you with a whole bundle of innovative features and techniques for automated image analysis. You will be surprised by the multitude of additional information that can be extracted from image data after segmenting it into image objects and by the possibilities to handle even textured or low contrast data, such as very high resolution (VHR), airborne, or radar data.

The concept behind eCognition is that important semantic information necessary to interpret an image is not represented in single pixels, but in meaningful image objects and their mutual relationships. The basic difference, especially when compared to pixel-based procedures, is that eCognition does not classify single pixels, but rather image objects which are extracted in a previous image segmentation step.

To turn this into action, eCognition offers a whole set of tools: Using a patented segmentation algorithm, eCognition allows homogeneous image object extraction in any desired resolution. This entails the simultaneous representation of image information on different scales. The segmentation procedure detects local contrasts and was especially developed to work even on highly textured data, such as VHR or radar imagery. Based on image objects, the problem of multisource data fusion is tackled by enabling parallel evaluation of image information of arbitrary source. eCognition features a set of interfaces which make information about image objects, features and classification transparent and accessible. The classification process is based on fuzzy logic, to allow the integration of a broad spectrum of different object features such as spectral values, shape, or texture for classification. Utilizing not only image object attributes, but also the relationship between networked image objects, results in sophisticated classification incorporating local context.

Combining all these features allows you to address image analysis tasks that have not been accessible until now. This powerful and universal method for object oriented image analysis significantly extends the range of image analysis applications and turns remote sensing data into more accurately classified geographic information for various purposes.



## How to use this user guide

We recommend that you start with the chapter “Take the Plunge”. It guides you through an entire example exercise. You do not need any previous knowledge of eCognition for this chapter. The purpose is to go through an example step by step to get a first impression of how to work with eCognition, its look and feel.

Find out about the different functionalities of eCognition and its characteristics in a brief eCognition overview on the following pages.

Following that, we recommend that you work through the “Guided Tours” for an introduction to the handling of the software by means of different examples and case studies. Take the time to work through these guided tours carefully. They cover practically all important features of the software and different approaches to image analysis problems. Have in mind, however, that the multitude of applications possible with eCognition cannot all be covered by the guided tours!

Have a look at the “Concepts & Methods” parallel to the “Guided Tours” to learn about the background and procedures used in eCognition. You will find themes on “what is object-oriented image analysis?” multiresolution segmentation, fuzzy classification, and a systematic overview of all features used for the classification of image objects. In addition, theoretical concepts are described.

The chapter “Functional Guide” shows you in detail how to use eCognition, in all its different functionalities. Important steps for an image analysis are described, e.g., importing and exporting raster data, multiresolution segmentation, classification, and important strategies for the use of eCognition.

Use the chapter “User Interface” in parallel to both the “Functional Guide” and the “Concepts & Methods” chapters. It provides a description of eCognition’s tool bar, menus and dialogs. Similar information is given in the tool tips in the software.

You can find some basic rules for the handling of eCognition as well as helpful advice or solutions to problems you might be confronted with under “Strategies & FAQs.”

The chapter “Index” is a table of contents comprised of the structure of this user guide, its main chapters and headings.

You can find important eCognition related terms and key words in alphabetical order with a short explanation and hyperlinks to a more detailed description in the chapter “Glossary.” Links placed throughout the user guide make it easier to navigate within and between chapters and serve as a dictionary of crucial features, or simply to refresh your memory.

## Background

### Gaps between the new generation of remote sensing systems and GIS

For years, many efforts have been made to develop automated procedures for updating GIS databases using remote sensing image data. However, the situation is still characterized by a considerable operational gap.

An increasing amount of very high resolution imagery (VHR) of astonishing quality provided by new digital airborne and space-borne sources has entered the remote sensing market. It is characterized by high user interpretability, rich information content, sharpness, accuracy, high image clarity and integrity. Although this kind of data diminishes the problem of allocating individual pixels to their most likely class, their rich information content dramatically aggravates the process of pixel labeling.

At the same time, the GIS market is asking for plug-and-play, ready-to-use products from these sources. Polygons representing objects of interest with correct labeling are needed to update GIS databases.

However, in many cases such objects are heterogeneous—shadowed areas, for instance, cause ambiguities—and a lot of knowledge and local context information is needed to extract real world objects properly.

In contrast to the strong need for automated technologies, available state-of-the-art image analysis procedures—basically pixel-based approaches—are limited. Typically, they have considerable difficulties dealing with the rich information content of VHR data; they produce a characteristic, inconsistent salt-and-pepper classification, and they are far from being capable of extracting objects of interest. Therefore, the vast majority of operational projects can be realized only by means of massive human interaction. The visual interpretation of an IKONOS 11 km x 11 km scene can take several days, for instance. The production of geoinformation from remote sensing image data is therefore still expensive.

#### eCognition's advantages

- Integrated system featuring a whole set of essential tools for automated image analysis
- Analysis even on textured or low contrast data, e.g., VHR satellite imagery, airborne or radar data
- Easy adaptation of image object resolution to specific image data and tasks
- Transparent, adjustable fuzzy classification
- Very easy and efficient training of a nearest neighbor knowledge base: click and classify
- Formulation and analysis of complex semantic tasks, analysis of contextual information and classification of land use
- Raster and vector representation of image objects simultaneously
- Analysis on an arbitrary number of layers
- Multi source data integration: analysis of arbitrary data types simultaneously, e.g., different resolutions, GIS layers, elevation data....
- Export of labeled polygons for updating GIS databases
- Intuitive handling
- Different transparent and accessible interfaces displaying information about image objects, features and classification for each single step

The existing constraints on automated data interpretation are so profound that an efficient integration of remote sensing and GIS is still a matter for research and development. The automated allocation and extraction of real world geographic objects from high resolution remotely sensed data is the central challenge for both the remote sensing and the GIS communities within the next few years.

In this situation eCognition with its revolutionary object oriented approach is opening new paths and perspectives. Find out with the help of this guide how eCognition supports you in extracting geoinformation from remote sensing data.

## eCognition overview

eCognition is based on an object oriented approach to image analysis. It is explicitly designed to work even on VHR or radar imagery and includes the option to develop knowledge bases for elaborate classification of local context and land use. The basic difference to pixel-based procedures is that eCognition does not classify single pixels, but rather image object primitives that are extracted in a previous image segmentation step. For this purpose eCognition features multiresolution segmentation, a patented procedure for image object extraction. It allows the segmentation of an image into a network of homogeneous image regions at any chosen resolution. These image object primitives represent image information in an abstracted form. Serving as building blocks and information carriers for subsequent classification, they offer some basic advantages:

- Beyond purely spectral information, image objects contain a lot of additional attributes which can be used for classification: shape, texture and—operating over the network—a whole set of relational / contextual information.
- Multiresolution segmentation separates adjacent regions in an image as long as they are significantly contrasted—even when the regions themselves are characterized by a certain texture or noise. Thus, even textured image data can be analyzed.
- Each classification task has its specific scale. Only image objects of an appropriate resolution permit analysis of meaningful contextual information. Multiresolution segmentation provides the possibility to easily adapt image object resolution to specific requirements, data and tasks.
- Homogeneous image objects provide a significantly increased signal-to-noise ratio compared to single pixels as to the attributes to be used for classification. Thus, independent of the multitude of additional information, the classification is more robust.
- Segmentation drastically reduces the sheer number of units to be handled for classification. Even if a lot of intelligence is applied to the analysis of each single image object, the classification works relatively fast.

- Using the possibility to produce image objects in different resolutions, a project can contain a hierarchical network with different object levels of different resolutions. This structure represents image information on different scales simultaneously. Thus, different object levels can be analyzed in relation to each other. For instance, image objects can be classified as to the detailed composition of sub-objects.
- The object oriented approach which first extracts homogeneous regions and then classifies them avoids the annoying salt-and-pepper effect of the more or less spatially finely distributed classification results which are typical of pixel-based analysis.

For more elaborate purposes eCognition additionally supports classification-based segmentation. Following classification, even very different types of objects can be merged to come up with objects of interest, for instance house roofs including chimneys, or urban areas including urban greens.

Beside the production and handling of networked image objects, the second key domain of eCognition's engine is its fuzzy classification system, which makes it possible to use the full advantages of the information contained in image objects and their mutual relations. It supports, on the one hand, very simple, rapid classification using a fuzzy nearest neighbor classifier. Individual image objects are marked as typical representatives of a class, and then the rest of the scene is classified ("click and classify!"). On the other hand, it allows the user to formulate concepts and knowledge about the relevant image content by means of fuzzy rules and, thus, to come up with a knowledge base which can also process elaborate contextual information.

## Segmentation

Basic to eCognition's procedures is multiresolution segmentation, a patented technique for image object extraction. It was developed to extract image objects at different optional resolutions (fine or coarse structures) in high quality. This technique has been adapted to finding image objects even in textured data, such as SAR images, satellite data of high resolution, or airborne data. It provides the possibility to easily adapt the extraction of meaningful image object primitives to specific tasks and image data.

In many cases image objects of interest cannot be extracted following a relatively general homogeneity criterion. For this reason, eCognition provides techniques for classification-based segmentation of image objects, merging even heterogeneous areas or allowing a classification-based refinement of image object shapes.

The different segmentation techniques can be used to construct a hierarchical network of image objects. Each level in this hierarchical network is produced by a single segmentation run. The hierarchical structure represents the information of the image

data at different resolutions simultaneously. Fine objects are sub-objects of coarser structures. Thus, each object “knows” its context, its neighborhood and its sub-objects. Operating on this network, interrelations between objects can be defined, e.g., “relative border length to class forest,” for utilizing this additional and often essential context information.

## Classification

eCognition supports different supervised classification techniques and different methods to train and build up a knowledge base for the classification of image objects. The frame of eCognition’s knowledge base for the analysis and classification of image objects is the so-called class hierarchy. It contains all classes of a classification scheme. The classes can be grouped in a hierarchical manner allowing the passing down of class descriptions to child classes on the one hand, and meaningful semantic grouping of classes on the other. This simple hierarchical grouping offers an astonishing range for the formulation of image semantics and for different analysis strategies.

Classification is conducted by fuzzy logic. Fuzzy classification delivers not only the assignment of one class to an image object, but the degree of assignment to all considered classes. The strategies for class assignment are transparent and therefore easier to adapt than if neural networks were being applied. Fuzzy logic even supports the combining of very different kinds of features within one class description by means of different logical operations.

Class descriptions are performed using a fuzzy approach of nearest neighbor or by combinations of fuzzy sets on object features, defined by membership functions. Whereas the first supports an easy click and classify approach based on marking typical objects as representative samples, the latter allows inclusion of concepts and expert knowledge to define classification strategies.

Both approaches to class descriptions can be combined in the knowledge base to achieve the most suitable strategy for each class. Developing a knowledge base for a specific classification task basically means editing, in detail, the class description of each class.

## Features for classification

In eCognition, features for classification are computed based on image objects, not on single pixels. Therefore, classification can address an astonishingly broad spectrum of different kinds of information. Beyond spectral information there is shape information, texture information and—operating over the network of image objects—many different relational or context features. For each feature this information is computed

per object considering its actual shape and size. Thus, the typical failures of filter operations, especially on borders between different types of areas, are avoided.

Powerful tools are so-called class-related features. They allow evaluation of the relation, for instance embedding or distance, of an image object to a specific class in the neighborhood.

eCognition contains features for object oriented texture analysis. By regarding the sub-objects of an image object, average attributes such as spectral standard deviation, contrast, or shape can be analyzed. It is also possible to analyze the composition of classified sub-objects. As the resolution of sub objects can easily be adapted to represent specific texture structures, object oriented texture analysis with eCognition is a powerful tool.

After vectorization, eCognition offers a set of features for shape analysis of image objects based on polygons and the objects' skeletons. These features allow access to explicit shape properties.

Based on this already extensive set of features, users additionally find a toolbox for customizing their own features. Thus, arbitrary features can be combined using different arithmetic operations, or image objects can be evaluated as to specific feature values in the particular neighborhood. These options are not limited to spectral information but can be applied to any chosen feature provided by eCognition.

## **Information about image objects and classification**

Complimentary to the bundle of attributes on which classification can act and the possibility of using contextual information, eCognition provides a whole set of different interfaces for detailed information about image objects, features and classification. Different tools visualize the attributes of image objects, indicating which features could be used to describe and distinguish classes. The image object information interface supports the detailed evaluation of each single image objects for any arbitrary class, allowing comprehensive understanding and adaptation of the entire rule base.

## **Multi source data fusion**

eCognition offers a variety of possibilities for simultaneously using different data types for analysis. In the segmentation process, different layers can be weighted as to their suitability for shaping resulting image objects. The knowledge base for classification allows the highly specific use of information from given image data.

Given these options, a multitude of methods are feasible. Objects resulting from a segmentation of one layer can be evaluated using information from another or several

other image layers. Differently scaled image objects can be created from image layers of different resolutions. Thus, information from different layers can be represented on different levels in the image object hierarchy, allowing for the evaluation of the different information layers in relation to each other.

## Vectorization

eCognition allows for the automated extraction of polygons based on image objects. Polygons are used to display outlines of image objects independent of the zoom factor; they are developed for explicit shape analysis of image objects or they can be used for the vector export of results. After vectorization, eCognition simultaneously holds image objects in raster and vector representations.

## Statistics and accuracy assessment

After classification, an elaborate statistic tool allows analysis of the network of classified image objects. For all objects of a specific class, a statistic can be produced concerning any chosen set of attributes. Thus, even relational features which were not used for the classification of the image objects themselves can be evaluated, such as embedding, or distances to specific other classes. The statistic tool supports the direct extraction of geoinformation from a scene. The results of a statistical analysis can therefore be exported, and for the purpose of automation the export can be included in the image analysis protocol.

An important issue is the detailed analysis of accuracy after classification. eCognition comes with different options for computing user's, producer's and overall accuracies. Additionally, the spatial distribution of fuzzy classification stability or highest membership value can be rendered over the scene. Each object's membership to a class can be visualized over the whole data set.

## Export of results

eCognition comes with different options for exporting results. To update a GIS database, the classified image objects in the scene can be exported in vectorized format as polygons with an attached attribute list. Beside the classification, this list can contain any chosen property for each image object. A similar option is the export of image objects by means of a thematic raster layer together with an attached attribute list.

Instead of exporting the whole structure of image objects, in many cases it might be enough to directly export statistical results, since they already contain the geoinformation of interest.

A further possibility is to export the whole scene as an image layer using the current view settings.

## Automation

eCognition allows for the recording, execution and editing of image analysis protocols in order to automate analysis processes. These protocols can record not only all procedures for image analysis, such as segmentation and classification, but also statistical evaluation and export of results. Such protocols can be used to run a user-defined analysis on more than one data set, once you have created a robust knowledge base. Consequently, the workflow process of the same classification tasks can be automated to a large degree, if working on comparable image data.

## Concluding remark

The object oriented approach to image analysis differs from pixel-based methods in many aspects. Indeed, if you are a remote sensing expert used to the common procedures of pixel-based analysis, our guide will challenge you to become acquainted with different approaches: eCognition means letting go of typical expectations and well-known approaches to image analysis.

Understanding the importance of the strategic approach to an analysis problem comes with mastering the new techniques. In many cases, eCognition offers more than only one solution to a certain problem. Going through the different “Guided Tours” carefully will help you to get an overview of the different possibilities.

eCognition is made for image analysis and not for image processing. Therefore, it does not contain procedures which allow modification of image data. If that is what you need, e.g., for preprocessing your image data, we refer you to one of the various software solutions available for image processing.

And now, enjoy your work with eCognition!

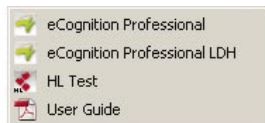


## What's new in eCognition Professional 4.0

The new eCognition Professional 4.0 features includes:

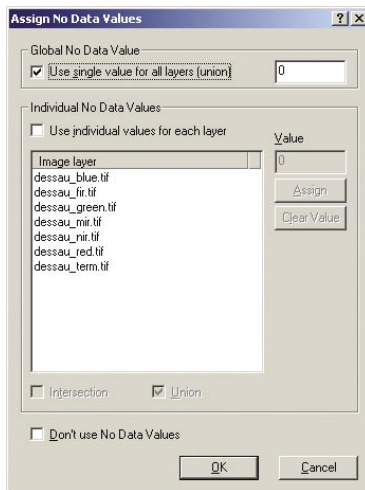
### Large Data Handling (LDH) Version

Until now, large datasets could not be processed (segmented) due to a lack of address space in the 32 bit Windows operating system. With version 4.0, virtual pointers have been added to the software structure which leads to an increased number of objects that can be addressed. As a result eCognition Professional 4.0 will be able to load and segment images of up to 46,000 x 46,000 pixel in size. Large data handling also means that there will be two separate eCognition executables; one normal and one LDH. This is similar to what we had for eCognition version 1 when there was a normal and a HD version. Of course, every customer will get both executables.



### No Data Values

It is now possible to define no data values for an eCognition project. This means that for defined values within the image data no image objects are generated. Consequently, these areas are not further treated for any analysis later-on. No data values can be defined for all channels individually or for the whole project.

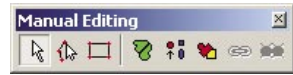


### Save subsets upon import

Subsets selected in the input dialog can now be stored as a own file. They are just subsetted from input data, having the same file structure but covering a smaller area.

## Improved manual editing of objects

Several selection tools for manual editing have been added. These tools allow the user to select multiple objects and either classify or merge them. The tools are polygon selection, rectangle selection and polyline selection.



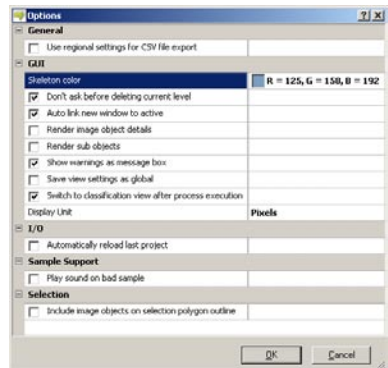
## Object Table

The object table gives an overview of objects of selected classes along with selected features. This table can be sorted by the different entries and on selection of an entry, the view focuses on the selected object (and vice versa). The displayed information can also be exported to a .csv file.

**Please note: The Object Table is not available in the LDH version.**

## Options dialog

An options menu has been added, which allows the user to edit several options regarding GUI settings, display preferences, tool settings etc.



## Further improvements and changes

Several smaller improvements and additions have been made in order to accelerate work with eCognition and to broad its functionalities:

- **Legend:** eCognition now features a legend. It displays the color icon and the name of all classes displayed in the active view.
- **Deleting Multiple Levels:** In the dialog to delete levels, now multiple levels can be selected and deleted.
- **Customized Features:** The customized features dialog has been extended to allow more complex calculations.
- **Export Statistics:** For the export of statistics, additional statistic types and operations have been added.
- **Export Scale Parameter Analysis:** now you can export the results from the Scale Parameter Analysis as an attributed ESRI shape file. Each line in the result file holds the analysis values. Besides, all additional statistics of the Scale Parameter Analysis are saved in a CSV file.
- **User Information Dialog:** A user information dialog has been added to allow identification as to who generated and altered which project or other eCognition output.
- **Segmentation Dialog Redesign:** The segmentation dialog has been slightly re-designed. Instead of four different values, two sliders are used to define the segmentation parameters.
- **New Object Features:** Some new object features have been introduced, which can be used for classification, creation of customized features or statistics. The most important new ones are global features which describe some global scene statistics and can be used beneficial to create customized features.
- **User Annotations:** All features which are created by the user, such as class descriptions, customized features and so forth can now be annotated by a users comment.
- **Additional Predefined Membership Functions:** some more “classic“ pre-defined membership functions have been added such as linear or “V“ functions.

## 2 INSTALLING ECOGNITION

Installing eCognition with a hardlock .....	25
Installing eCognition with a local hardlock .....	25
Installing eCognition with a server hardlock .....	26
<b>The Hardlock Utility Software .....</b>	<b>28</b>
Initialize Hardlock Server module with alf file .....	28
Export license information .....	28
Update License .....	29
Server Hardlock, Update from clipboard .....	31
Server Hardlock, Update by File .....	32
Generate Report .....	33
Diagnostix .....	34
About .....	34
<b>The Aladdin DiagnostiX Tool .....</b>	<b>34</b>
Checking for a Hardlock Module .....	35
Creating reports .....	36
Setting environment variables .....	36
Environment Variables .....	37
Hardlock and License Troubleshooting .....	40
Common errors .....	41

## Installing eCognition with a hardlock

The software is protected by hardlock keys, which are connected either to the parallel, serial or USB port of the computer. There are two different types of keys: local hardlocks and server hardlocks. Depending on the type of hardlock, different installation configurations are necessary.



**Single License  
Hardlock LPT**  
connection to either  
parallel or serial port  
for single  
licenses



**Single License  
Hardlock USB**  
connection to USB port  
for single licenses



**Server License  
Hardlock LPT**  
connection to parallel  
port for network  
licenses up to 250 users



**Server License  
Hardlock USB**  
connection to USB port  
for network licenses up  
to 250 users

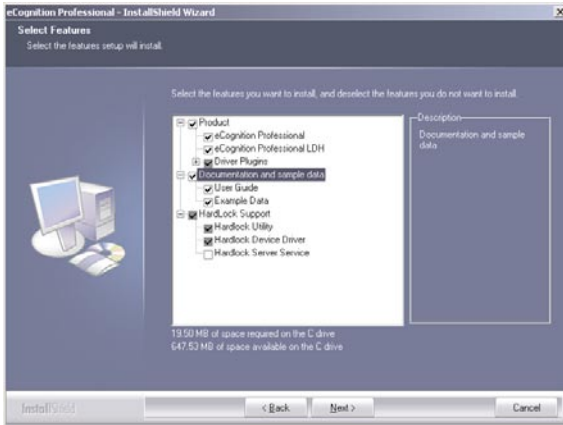
## Installing eCognition with a local hardlock

A local hardlock provides a license for one single eCognition system. Just install the eCognition software and the hardlock driver software is installed automatically.

For the installation you need:

- The hardlock
- The eCognition software CD.

The components to install differ depending on your needs. There are two eCognition versions, **eCognition Professional** and **eCognition Professional LDH**. We recommend installing both versions. eCognition LDH features a different address space management which enables it to handle large data, hence LDH for Large Data Handling.



The image shows the recommended installation configuration for eCognition with a single hardlock.

**Note:** Attach the hardlock to the parallel or USB port **before** running the eCognition software.

### Installing eCognition with a server hardlock

The server hardlock provides a floating license with up to 250 licenses to any computer in the network where the server hardlock is installed.

For a server hardlock there are two basic setup options. Option one is to install only the hardlock server software, the other option is to install both the eCognition software as well as the hardlock server.

For both types of installation you need:

- the server hardlock
- the eCognition software CD
- the \*.alf license file, which is provided to you via email or attached to the software package on a floppy disk

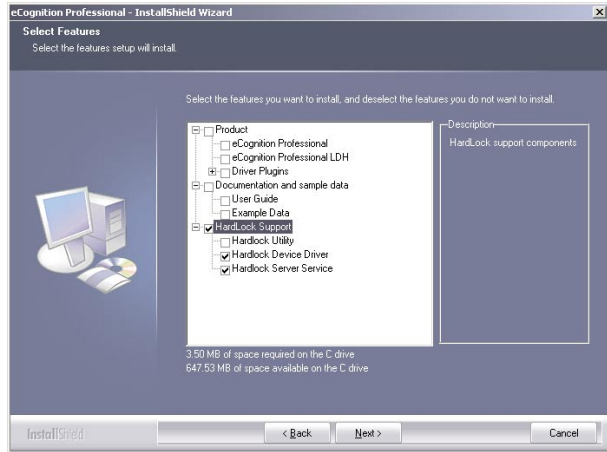
**Note:** The installation of the Hardlock Server software is only mandatory on the PC where the Server Hardlock is plugged on.

### Installing only the Hardlock Server Software

In case you want the license to be provided by a dedicated license server you need to install only the hardlock server software. For this purpose deactivate all items in the installation feature selection and select the items “Hardlock Driver”, “Hardlock Server” and „Hardlock Server Service“ only.

Then follow the installation instructions. The picture to the right shows the setup option which installs **only** the hardlock server software.

**Note:** Attach the hardlock to the parallel or USB port **before** installing the eCognition software.

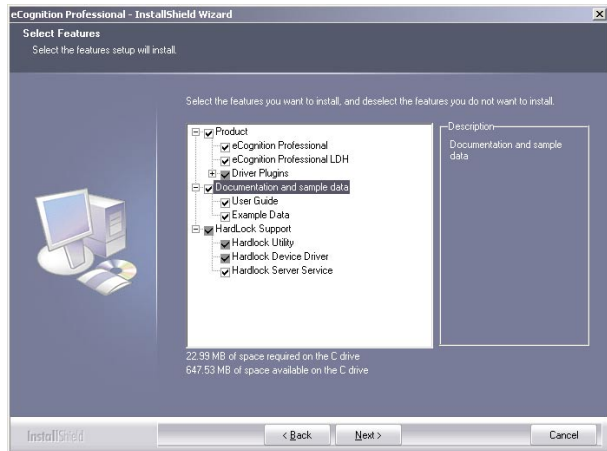


### Installing the Hardlock Server and eCognition Software

If you want the license to be provided by a license server, which is also used to run eCognition, you need to install the eCognition software components and the hardlock server software. Select the features as shown in the image below and follow the installation instructions.

The picture to the right shows the setup option which installs the hardlock server software **and** eCognition.

**Note:** Attach the hardlock to the parallel or USB port **before** installing the eCognition software.



## The Hardlock Utility Software

The Hardlock Utility software can be found in the start menu section of eCognition. It contains the most important license utility tools to manage and troubleshoot eCognition licenses. Select one of the subordinate options on the left side of the window.

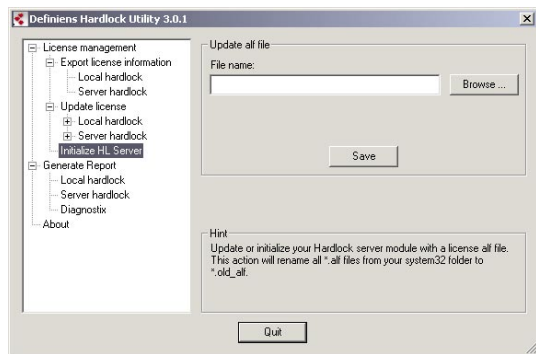
### Initialize Hardlock Server module with alf file

This option is used to initialize or update your Hardlock Server service with an alf file. The alf file is crucial to run eCognition with a server Hardlock since it provides additional license information. This has not to be done if you use a Local Hardlock.

You have to initialize or update your Hardlock Server module after installing it for the first time or if you got a new \*.alf file from your License Contact Person (e.g. in case the old one is corrupt).

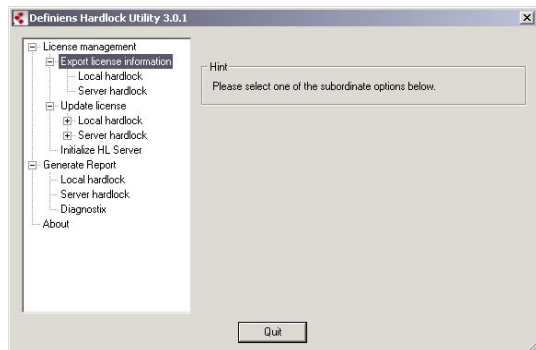
To initialize or update your Hardlock Server service, select “License Management > Initialize HL Server” and browse to the alf file which was provided to you via e-mail or attached to the software package on a floppy disk. Press “Save” to update or initialize the alf file.

The alf file will be simply copied to your system32 folder. This will also rename all previously present \*.alf files from your system32 folder to \*.old\_alf to prevent problems with old or corrupt alf files.



### Export license information

To order license updates, you have to provide your eCognition license contact person with the license information on your computer or network.



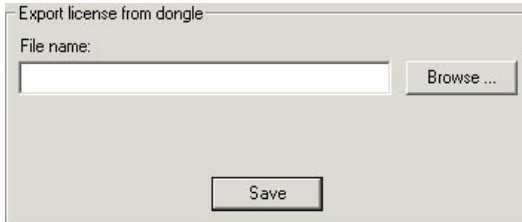


Select “license management > Export license information”.

### Local Hardlock

Use the option Local hardlock in case you have a hardlock locally connected to your PC.

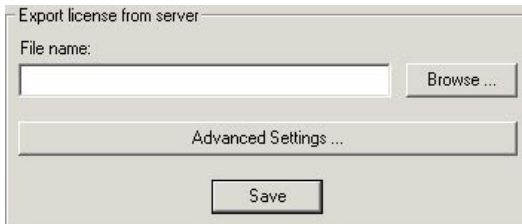
Choose a filename and press “Save” to store your license information in a file.



### Server Hardlock

Use the option “Server Hardlock” in case you want to get the license information of a server hardlock. The server hardlock is detected automatically. If the hardlock is not found automatically use the Advanced Settings to manually insert the IP-Address or the name of the PC where the Hardlock Server is installed.

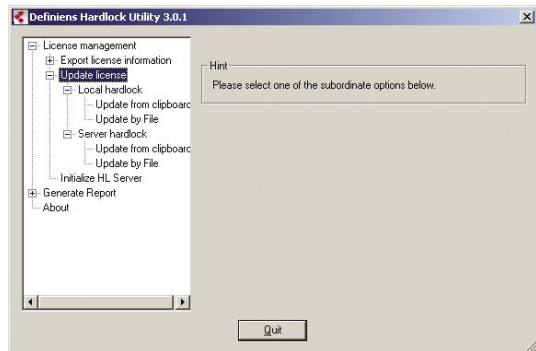
Choose a filename and press “Save” to store your license information in a file.



Send the produced \*.ctv file to your license contact person.

## Update License

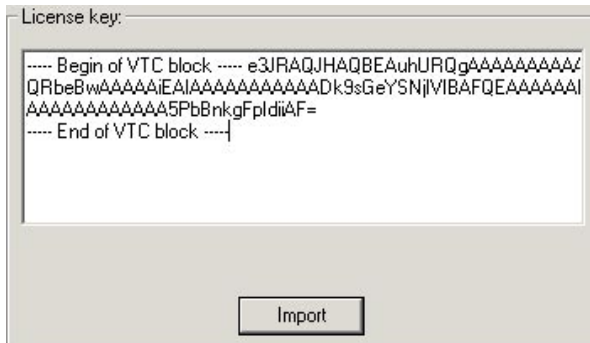
When a license update is provided by your eCognition license contact person this is either done using an \*.exe file, a \*.vtc file or a vtc block. The \*.exe file can be executed directly. To apply the \*.vtc file, use the option “License Management > Update License” of the Hardlock Utility.



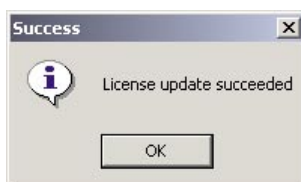
### Local Hardlock, Update from Clipboard

To update the license of a Hardlock locally connected to this PC using a vtc block select the option “Local hardlock > Update from clipboard”.

Paste the license string from an update email sent to you by your eCognition license contact person into the “License key window” (see below) and press “Import”.



After pressing Import your license should be updated.

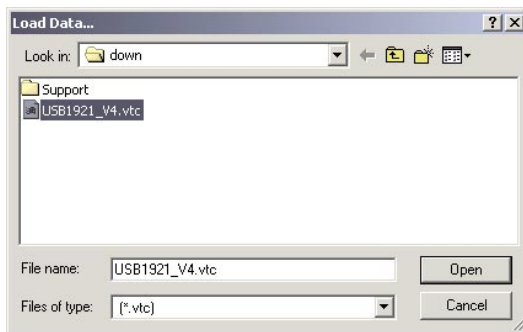
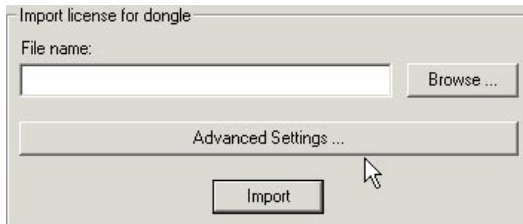


This action will only work for hardlocks locally connected to this PC.

## Local Hardlock, Update by File

To update the license of a hardlock locally connected to this PC using a \*.vtc select the option “Local hardlock > Update by file”.

Browse to the \*.vtc file sent to you by your eCognition license contact person and press “Import”.



After pressing Import your license should be updated.



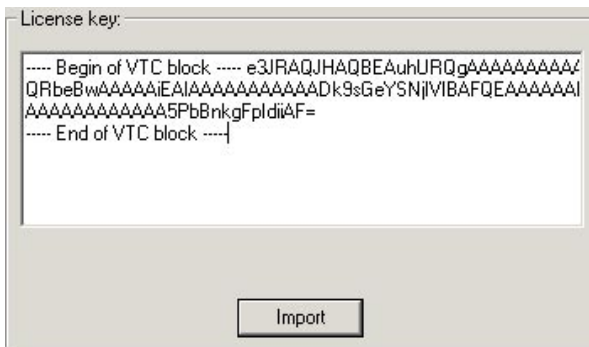
This action will only work for hardlocks locally connected to this PC.

## Server Hardlock, Update from clipboard

To update the license of a server hardlock using a \*.vtc select the option “Server hardlock > Update from clipboard.”

Paste the license string from an update email sent to you by your license contact person into the License key window (see below) and press “Import”.

The server hardlock is detected automatically. If the hardlock is not found automatically use the “Advanced Settings...” to manually insert the IP-Address or the name of the computer the Hardlock Server is installed on.



After pressing “Import” your license should be updated.



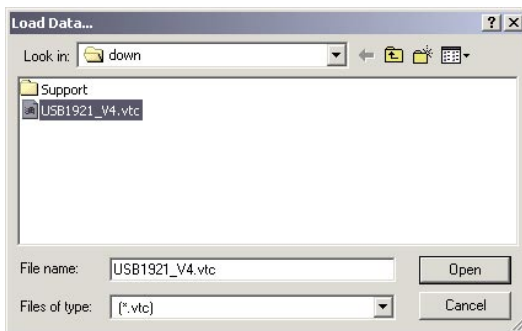
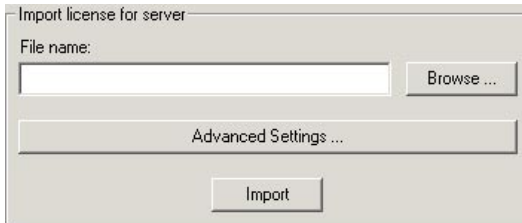
This action will only work for server hardlocks.

## Server Hardlock, Update by File

To update the license of a server hardlock using a \*.vtc select the option “Server hardlock > Update by file”.

Browse to the file sent to you by your license contact person and press “Import”.

The server hardlock is detected automatically. If the hardlock is not found automatically use the “Advanced Settings...” to manually insert the IP-Address or the name of the license server.



After pressing “Import” your license should be updated.



This action will only work for server hardlocks.

## Generate Report

Reports are used to access the licence status of the hardlock.

Select “Generate Report”.

## Local Hardlock

Use the option “Local hardlock” in case you have a hardlock locally connected to your PC. This function creates a report of a hardlock locally connected to your PC.

Press “Create” to create the report.

The result is shown in the report window.

By pressing “Save” the report is saved to a text file. If you contact the eCognition Support you might be asked to send this file.

Action	Result
HL Test Version	3.0.1
Search local Hardlock	found
Connect to Hardlock	ok
Checking FUS	SN: 1921 (hw:00000781)
Checking expire date	07/07/2004
eCognition Product Line	0 licenses
Analysis Engine	0 licenses
Data Control	0 licenses
DUE engine	0 licenses
eCognition Acceptance Tests (internal only)	0 licenses
eCognition Client	0 licenses
eCognition Forester	0 licenses
eCognition Client ProSMART Edition	0 licenses
eCognition Developer (internal only)	0 licenses
eCognition Elements	0 licenses
eCognition FREE Client	0 licenses
eCognition Professional ProSMART Edition	0 licenses
eCognition Professional	1 licenses
eCognition Versions	
Version 4.0	
Large data handling	
Large data handling	
Hardlock Version	
Hardlock API	3.82
Hardlock Server	
Environment Variables	
MLEC_FREQ	1000
NLS_ENABLE	101.1.143
NLS_WAIT	2000

## Server Hardlock

To update create a report of the license information of a server hardlock select the option “Server hardlock”. The server hardlock is detected automatically. If the hardlock is not found automatically use the “Advanced Settings” to manually insert the IP-Address or the name of the license server.

Press “Create” to create the report. The result is shown in the report window.

By pressing “Save” the report is saved to a text file. If you contact the eCognition Support you might be asked to send this file.

Action	Result
HL Test Version	3.0.1
Search remote Hardlock	found
Connect to Hardlock	ok
Checking FUS	SN: 1576 (hw:00000633)
Checking expire date	unlimited license
eCognition Product Line	50 licenses
Analysis Engine	50 licenses
Data Control	50 licenses
DUE engine	50 licenses
eCognition Acceptance Tests (internal only)	50 licenses
eCognition Client	50 licenses
eCognition Forester	50 licenses
eCognition Client ProSMART Edition	50 licenses
eCognition Developer (internal only)	50 licenses
eCognition Elements	50 licenses
eCognition FREE Client	50 licenses
eCognition Professional ProSMART Edition	50 licenses
eCognition Professional	50 licenses
eCognition Versions	
Version 1.0	
Version 2.0	
Version 3.0	
Version 4.0	
Beta	
Large data handling	
Challenger Product Line	50 licenses
Challenger Developer Studio	50 licenses
Challenger Enterprise Client	50 licenses
Analysis Engine	50 licenses
Data Control	50 licenses
DUE engine	50 licenses

## Diagnostix

To provide detailed system information and setting of environment variables, a separate tool called DiagnostiX is used. Press “Starting Aladdin DiagnostiX” to start the diagnostics tool.

## About

Here you will find information about the Version of the Hardlock Utility Software. This can be useful if you when communicating with your eCognition license contact person.



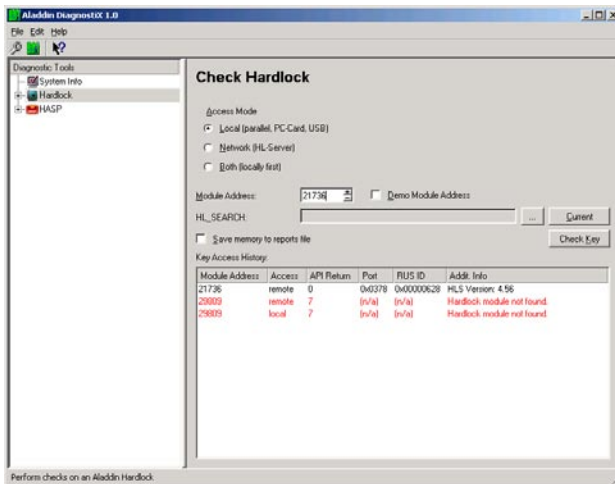
## The Aladdin DiagnostiX Tool

The Aladdin DiagnostiX utility collects relevant information of your system and of your Hardlock module. This information will help you and the eCognition support team to solve problems you may encounter when using eCognition. For installation and troubleshooting purposes, the following functions are required:

- Checking for a Hardlock module.
- Creating reports on Aladdin devices and their environment.
- Defining Hardlock environment settings.
- Viewing System Data.

### Checking for a Hardlock Module

From the DiagnostiX tools pane (left) select “Hardlock“. The Check Hardlock screen appears in the main pane (right). All Hardlock access results are summarized and tabulated in the Key Access History.



To check a Hardlock module

1. Select the Access Mode type.
  - To check the Hardlock module on the local machine select local (parallel, PC-Card, USB).
  - To check the Hardlock module on the network select remote (HL-Server).

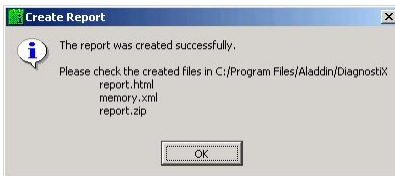
2. Enter the **21736** into the module address field
3. Check “Save memory” to report file if you want data in Hardlock module memory to be included to the generated report file.
5. Click “Check Key”.

Details for the access are displayed in the Key Access History pane.

## Creating reports

The function “Create Report” creates report of the PC, Hardlock and network settings to allow support to identify possible setup problems which may occur during installation. To create a report, select „Create Report“ from the Edit menu. A message box will appear to indicate that the report has been successfully created.

**Note:** Check the Hardlock modules as described in chapter „Checking for a hardlock module“ before you create a report. Check both remote and local on a PC where the Hardlock Server is installed and on a PC where only eCognition is installed.



The zip file can then be sent to the eCognition support team.

## Setting environment variables

In some cases environment variables need to be set for the hardlock software. Such cases can be usage of a server hardlock locally, hardlocks in complex networks, slow network connections etc.

Defining Hardlock Environment settings:

1. Change to the „Hardlock Environment“ part of the DiagnostiX tool.
2. Select either „System“ or „Current user“ in the Hardlock Environment screen. Selecting „Current User“ implies that setting modifications will only apply to the



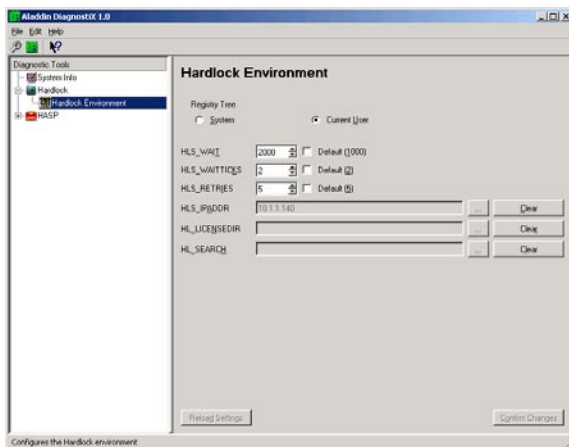
current user on the system. Only users with the required administrative rights will be able to modify System settings.

3. Set any of the following parameters:

HLS\_WAIT  
HLS\_WAITTICKS  
HLS\_RETRIES  
HLS\_IPADDR  
HL\_LICENSEDIR  
HL\_SEARCH.

For further details on the parameters see below.

4. To activate environment settings parameters, click Confirm Changes.
5. To restore current settings click Reload Settings.



## Environment Variables

### HLS\_WAIT

Use this parameter to set the timeout in milliseconds for the 32-bit API. Use a higher value if you work in a slow network

Syntax: HLS\_WAIT

Range: 20-3000

Default: 1000

**HLS\_WAITTICKS**

Use this parameter to set the timeout for the 16-bit API. Use a higher value if you work in a slow network

Syntax: HLS\_WAITTICKS

Range: 1-999

Default: 2

**HLS\_RETRIES**

Use this parameter to set the number of retries before timeout. Use a higher value if you have a bad network connection or an overloaded network.

Syntax: HLS\_RETRIES

Range: 2-30

Default: 5

**HLS\_IPADDR**

Use this parameter to set the server IP address. Use this parameter if a client cannot find the PC where the Hardlock Server is installed.

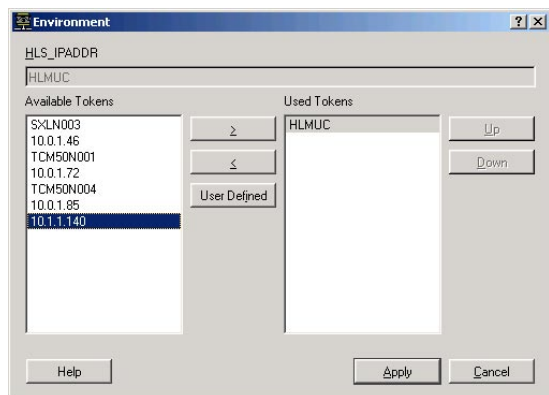
Syntax: HLS\_IPADDR

Range: IP address

Default: None

Setting the parameter:

1. Click the corresponding button for this parameter in the Hardlock Environment screen. The HLS\_IPADDR String Editor opens.
2. From the Available Tokens list select a token and use the >> button to move a token to your Used token list. Your selection appears in the list as well as the HLS\_IPADDR current parameter field.
3. To define your own tokens, click User Defined. Enter the token in the field provided. Click OK. The token you have defined should appear in the Used Token list as well as the HLS\_IPADDR current parameter field.
4. Review your selections. To remove an item from the Used Token list, select the token and use the << button.
5. To save your settings, and close the String



Editor, click OK. Your selections appear in the HLS\_IPADDR field in the Hardlock Environment screen.

- To clear the modified values for this parameter, click Clear.

### HL\_LICENSEDIR

Use this parameter to direct a Server Hardlock to where the Aladdin License Files (ALFs) are stored. This function is needed when you want to use a server hardlock locally. The default location of the \*.alf file is „C:\WINDOWS\system32“.

Syntax: HL\_LICENSEDIR

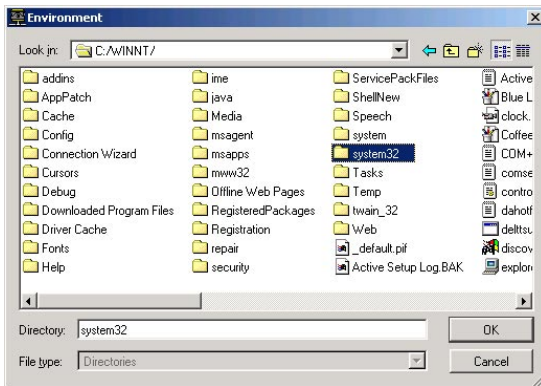
Range: An address on the local drive

Default: None

**Note:** This variable is set by default while the installation routine of Hardlock Server software.

Setting the parameter:

- Click the corresponding button. A separate window opens.
- Point the path to or enter the name of the license directory. Click OK. The window closes and name of the directory appears in the HL\_LICENSEDIR field.
- To clear any modified values for this parameter, click Clear.



### HL\_SEARCH

Use this parameter to determine where to search for the Hardlock.

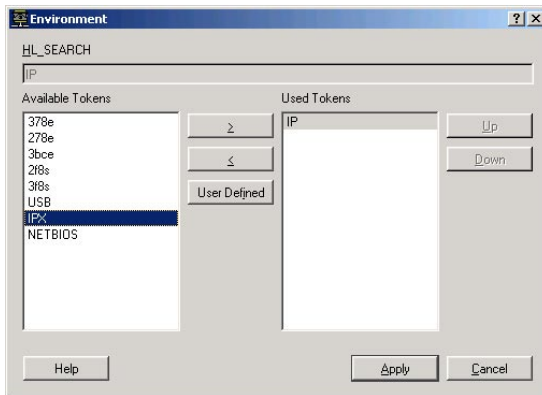
Syntax: HL\_SEARCH

Range: Parallel, serial port, USB, IP/IPX/NETBIOS

Default: None

Setting the parameter:

1. Click the corresponding button for this parameter in the Hardlock Environment screen. The HL\_SEARCH String Editor opens.
2. From the Available Tokens list select a token and use the >> button to move a token to your Used token list. Your selection appears in the list as well as the HL\_SEARCH current parameter field.
3. To define your own tokens, click User Defined. Enter the token in the field provided. Click OK. The token you have defined should appear in the Used Token list as well as the HL-SEARCH current parameter field.
4. Review your selections. To remove an item from the Used Token list, select the token and use the << button.
5. To save your settings, and close the String Editor, click OK. Your selections appear in the HL\_SEARCH field in the Hardlock Environment screen.
6. To clear the modified values for this parameter, click Clear.



## Hardlock and License Troubleshooting

When you start eCognition, a valid license is sought. A connection to the hardlock server is established and a license request is sent to the hardlock server. The Hardlock Server software package is installed as a Windows service on the server that is listening for these requests. In this process error may happen.

Below is a list of various errors which may be encountered:

## Please read this – It saves time!

The first thing you should do when encountering problems with the hardlock is to create report files using the DiagnostiX tool and send them to

**support.ecognition@definien.com**

along with a short description of the problem you encounter. If you have a local hardlock, a local report is sufficient. If you have a server hardlock create local and remote tests on both the server PC and the client PC. Please make sure that hardlock checks are included in the tests. For details see „checking for a hardlock module“ and „creating reports“. In addition screenshots of the „Test Module“ function of the HLTest utility are most helpful.

## Common errors

### Error 7: Hardlock not found:

This can have different reasons. The most obvious is that no hardlock is connected, or - in the case of a serial port hardlock – that hardlocks of other software products create conflicts. In some cases this error is also displayed when the hardlock has expired. Clarity can be obtained by executing the hardlock utility and checking the status. If more than one hardlock is used, the eCognition hardlock should be connected first to the port. Laptops sometimes have special settings for their parallel ports or even have them disabled. This can also lead to a hardlock not being detected.

### Error: ALF License file not found

This error occurs in connection with server hardlocks. It indicates that either the \*.alf file is not present or the hardlock can not access it. To solve this problem try adding the hardlock to the server (see „Installing Hardlock Server“). The second possible reason is that you are trying to work locally with a server hardlock. To allow this the environment variable HL\_Licensedir needs to be set. (see „environment variables“).

### Error 28: Date fake detected:

To prevent the user from changing the date and time settings of his computer and thereby avoiding the expiry date of a hardlock, the date and time settings are checked. If they are changed in between two sessions, the hardlock is disabled. Therefore, it is advisable to check whether the date settings of two computers are identical before transferring the hardlock from one computer to another.

**Error 37: Expiry date reached**

The hardlock needs to be updated. You can request an update executable at [license.ecognition@definiens.com](mailto:license.ecognition@definiens.com). It can be sent via email. Please always specify the serial number of your hardlock.

**Error 1009: Cannot open hardlock driver:**

Either the driver is not installed at all or a wrong driver is installed. Make sure to install the appropriate driver.

**Error installing hardlock driver:**

This can happen if a driver which is not suited for the operating system is used, or more likely if the user has no administration rights to the computer.

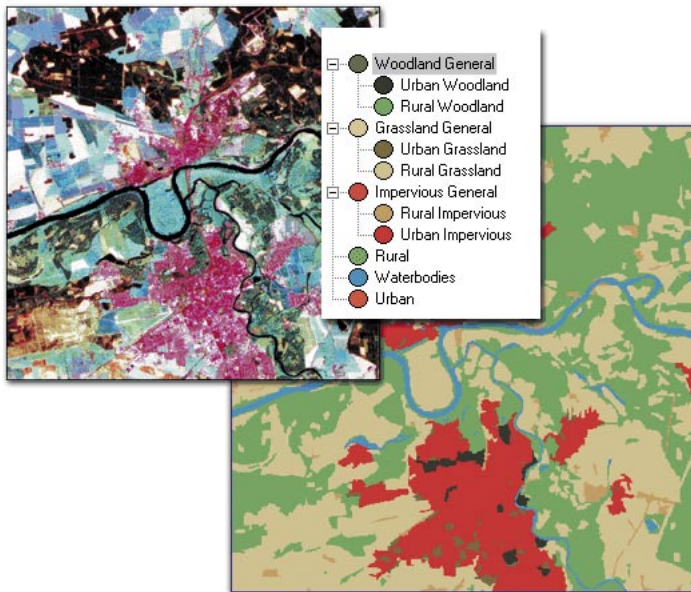
# 3 TAKE THE PLUNGE

## Getting a first glimpse of eCognition

Rather than going into too much detail too soon, we recommend that you start with this chapter rather than the theoretical chapters that follow. Work through “Take the Plunge” first to get a feel for eCognition’s user interface and its most basic features.


Since eCognition is based on a new approach to image analysis, taking this short trip may sometimes make you feel as if you have been “thrown into the deep end.” This is just what is intended. The purpose of this chapter is to go through an example step by step without having to worry about the whys and wherefores of your actions. The goal is to get a first impression of how to work with eCognition. All features of eCognition will be explained systematically later on.

The first example you will work on is a subset of a LANDSAT TM scene. It shows the town of Dessau on the Elbe River in Saxony-Anhalt, Germany. The goal is to classify the image so that a thematic map of the area can be created.



Courtesy of Ministry of Environmental Affairs of Sachsen-Anhalt, Germany

## Creating a new project and loading the raster data

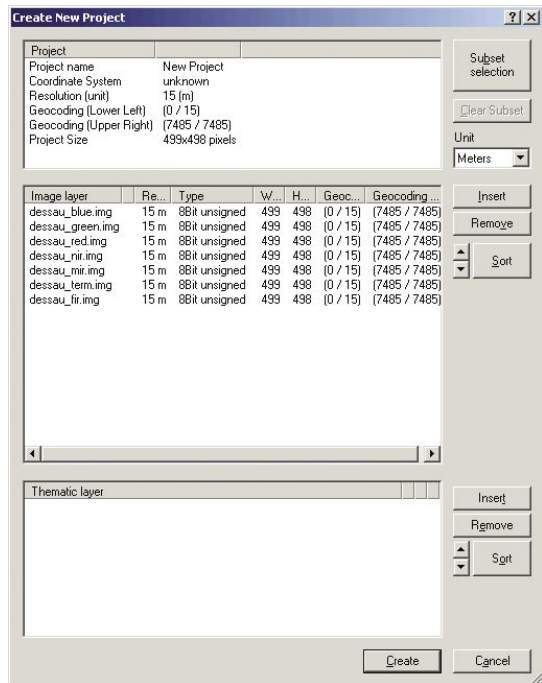
1. From the “Project” menu choose “New...” or click  in the tool bar.
2. Navigate to the directory “...\data\plunge\.”
3. Select the following .img image files and click “Open.”

- dessau\_blue.img
- dessau\_fir.img
- dessau\_green.img
- dessau\_mir.img
- dessau\_nir.img
- dessau\_red.img
- dessau\_term.img

4. Change the order of the layers by marking them and by using the arrows to the left of the “Sort” button.


5. Click “Create.”

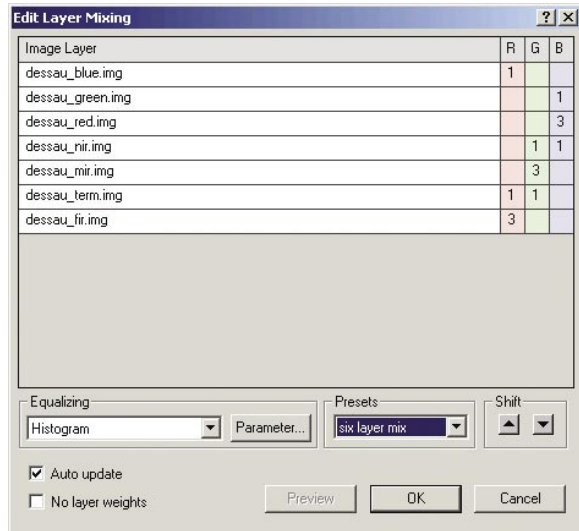
The new project is now being started and raster layers are being imported.




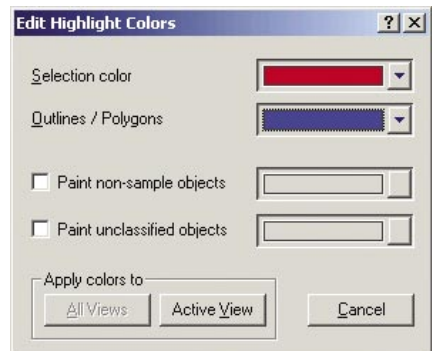


## Changing the color composition of the displayed image

1. Select “Layer Mixing...” from the “View” menu or click  in the tool bar to open the layer mixing dialog.
2. Select “Histogram” under “Equalizing” to apply a histogram stretch.
3. Choose “six layer mix” as “Presets.”
4. Click “OK.”




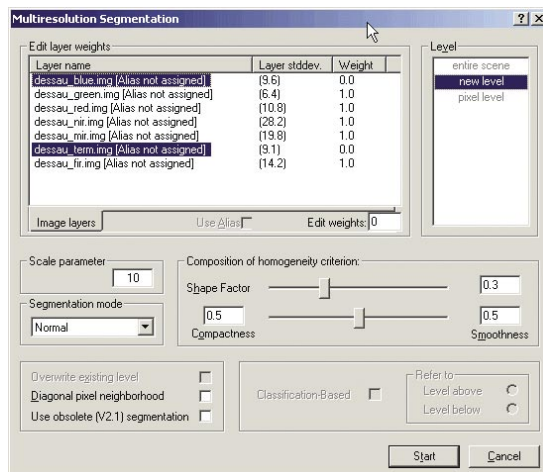
5. Choose “Edit Highlight Colors” from the “View” menu or click  in the tool bar to change the highlight color setting.
6. Select red as the selection color.
7. Click “Active View” to activate these highlight color settings.




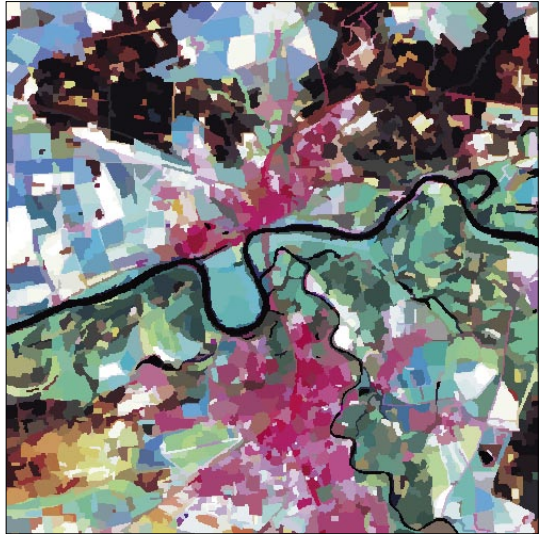
## Creating image objects

Now that you have created a project, you can move on to making your first object oriented image analysis. Object oriented processing of image information is the main feature of eCognition. For this reason, the first step in eCognition is always to extract image object primitives, which will become the building blocks for subsequent classifications. You will now produce such image objects with multiresolution segmentation. Multiresolution implies that it is possible to generate image objects at any chosen resolution.




1. From the “Segmentation” menu choose “Multiresolution Segmentation...” or click  in the tool bar.
2. Weight “dessau\_blue.img” and “dessau\_term.img” **0** in the field “Select and edit weights.” They will not be considered for segmentation.
3. Insert **10** in the field “Scale Parameter.”
4. Choose “Normal” in the field “Segmentation Mode.”
5. Set the “Shape Factor” to **0.3** in the section “Composition of homogeneity criterion.” The weight for “Color” is updated automatically to **0.7**.
6. Weight “Smoothness” with **0.5**. “Compactness” is also updated automatically.
7. Ensure that “Diagonal pixel neighborhood” is disabled.
8. Click “Start” to start the segmentation process.

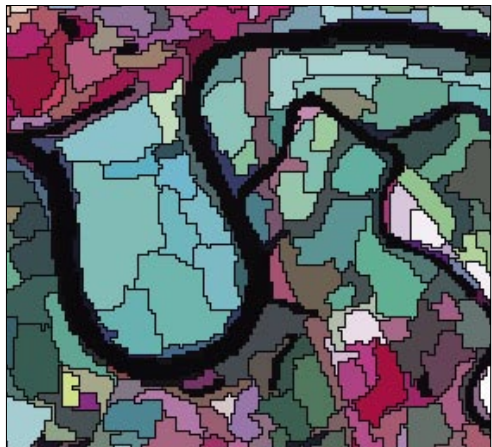
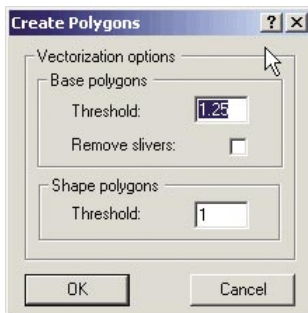




When the segmentation process is finished, there are different ways to display the image objects. By default the image objects are transparent and highlighted only on mouse-click. To view the image objects colored in their mean value, click the  icon or change the view settings for image data from “Pixel” to “Object mean.”



To view the borders of the objects there are two possibilities in eCognition:

- Create polygons  to view borders of objects (“Polygons > Create Polygons...” or click ). To show or hide the polygons use the  button. Since polygons use memory they should only be created if features based on polygons or skeletons will be used in the later classification or if vector files will be exported.



- Use the  button to show or hide outlines without creating polygons first. Besides, it is possible to view the outlines in the color of the classification of the corresponding object. To show the outlines colored in the classification colors you can use the  button after a classification is performed.

In this case polygons are created for the visualization in the following.

This visualization helps you to check if the extracted image objects fit your purpose or not. If you now change the image data settings back to the pixel mode in the view settings, you can observe the object borders while still seeing the single pixels.

In comparison to a single pixel, an image object offers substantially more information. Move on to find out how to access it

### Obtaining information about image objects

The “Image Object Information” dialog provides detailed information about the selected image objects as to its features and classification. There is no classification information available yet, since you have not classified the image.


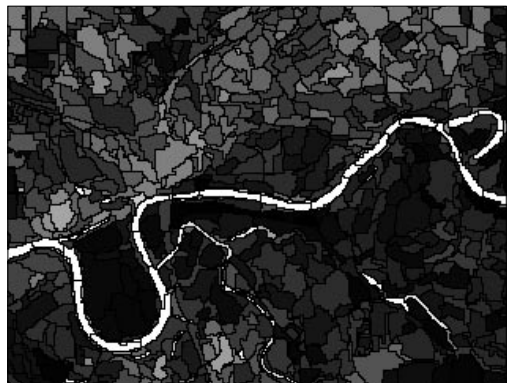
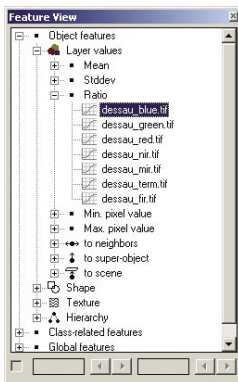
1. Open the image object information dialog by clicking the icon  if it is not yet open.
2. Click an arbitrary image object.



Image Object Information 1	
Feature	Val
<b>Layer values</b>	
Mean dessau_blue.img	71.86
Mean dessau_red.img	30.57
Mean dessau_green.img	30.71
Mean dessau_nir.img	62
Mean dessau_mir.img	63.57
Mean dessau_term.img	151.29
Mean dessau_fir.img	30.43
Brightness	62.92
Max.Diff.	1.921
<b>Statistics</b>	
Stddev dessau_blue.img	3.356
Stddev dessau_red.img	4.686
Stddev dessau_green.img	2.814
Stddev dessau_nir.img	2.673
Stddev dessau_mir.img	4.371
Stddev dessau_term.img	0.4518
Stddev dessau_fir.img	2.969
<b>Ratios</b>	
Ratio dessau_blue.img	0.1632
Ratio dessau_red.img	0.069413
Ratio dessau_green.img	0.069737
Ratio dessau_nir.img	0.1408
Ratio dessau_mir.img	0.1443
Ratio dessau_term.img	0.3435
Ratio dessau_fir.img	0.069089
Features Classification Class Evaluation	


This dialog gives you all the necessary information about one single object. When creating a class hierarchy, this dialog helps you to find features which separate one class from another. Another tool which helps you with this task is the feature view. The feature view allows you to display one feature for all image objects. The image objects are rendered in gray values which correspond to the feature value. The brighter an object is, the higher is its feature value for the selected feature.

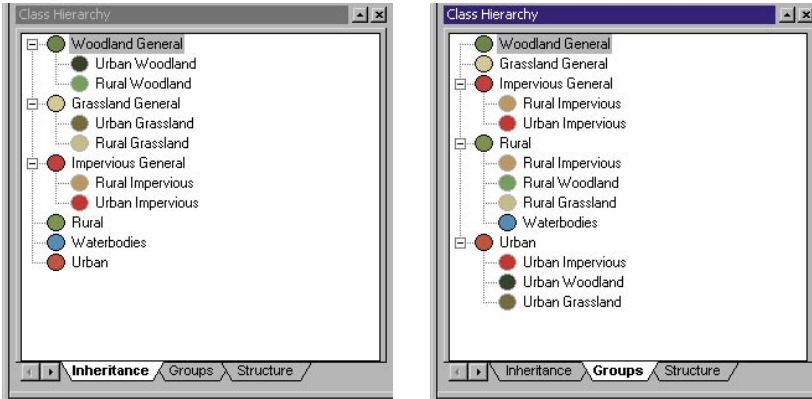
3. From the “Tools” menu choose “Feature View...”
4. Select the feature “Object features > Layer values > Ratio > dessau\_blue.img” by double-clicking it. The objects will then be colored according to their feature value for the selected feature. A high gray value represents a high feature value, a low gray value a low feature value. You can visualize features also out of every other dialog where features are selected, for instance the “Insert Expression” dialog or the “Select displayed Feature” dialog. In these cases you open a pop up menu with a right click and select “Update range”.



## Loading a class hierarchy

Up to now, you have extracted image objects and learned how to call up information contained in them. This information will be used for classifying the previously segmented image objects. As this is an introductory tour of eCognition, an already existing class hierarchy will be loaded into the project for this purpose.

1. From the menu item “Classification” choose “Open Class Hierarchy...” or click  in the tool bar. The “Class Hierarchy” editor is now open.
2. Select the item “Classification > Load Class Hierarchy...” in the menu bar.
3. Select the file “plunge.dkb” and open it.



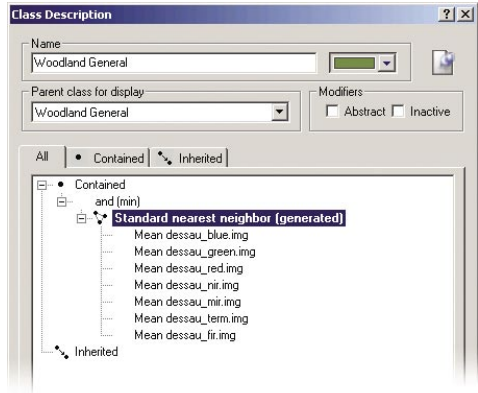
At this point a crucial aspect of the knowledge base structure in eCognition will be introduced. As you can see above, there are three different registers within the class hierarchy dialog: “Inheritance,” “Groups” and “Structure.” Inheritance and groups define dependencies between classes concerning classification of image objects. They complement each other: while child classes inherit feature descriptions from their parent classes in the “Inheritance” register, the “Groups” register summarizes child classes in meaningful semantic groups. Its potential becomes obvious when you compare the hierarchical structure of classes under inheritance with that of groups: while child classes in the inheritance hierarchy differentiate a general parent class by, e.g., evaluating their embedding in a certain context, the most varied child classes can be aggregated into groups under a superior class, resulting in a summarized land use classification rather than a mere land cover classification.

The “Structure” register has a completely different purpose: it is only used for classification-based segmentation, so it does not play a role here.

In the new class hierarchy displayed above, note the three parent classes called *Woodland General*, *Grassland General*, and *Impervious General*. Each of these parent classes has two child classes in the inheritance register. These child classes contain additional features distinguishing them from the parent class as to embedding or nonembedding in an urban context. The classes *Rural*, *Waterbodies* and *Urban* do not spawn child classes. As you can see in the groups register, *Urban* and *Rural* are only used to combine the rural and the urban child classes, respectively. For this reason, *Rural* and *Urban* do not contain their own feature descriptions in this specific case. *Waterbodies* will not be distinguished here.

4. Open the class description of *Woodland General* in the “Inheritance” hierarchy by double-clicking the class. Alternatively, you can click the class with the right mouse button and choose the item “Edit Class” in the following pop-up menu.

The “Class Description” dialog is now open. What you see is the feature description of the class *Woodland General*. In this case it only consists of a so-called standard nearest neighbor. Nearest neighbor is a classifier used to classify image objects based on given sample objects within a defined feature space. As you can see, this feature space consists here of the seven layer mean values (for the seven TM bands). The samples will be introduced in the following step. Click the other classes to see whether they contain the “Standard nearest neighbor” expression as well.



**Note!** In contrast to other possibilities of formulating a class description allowing the analysis of complex dependencies, nearest neighbor is eCognition’s solution for the quick and simple classification of image objects just by clicking typical image objects as representative samples of a class.

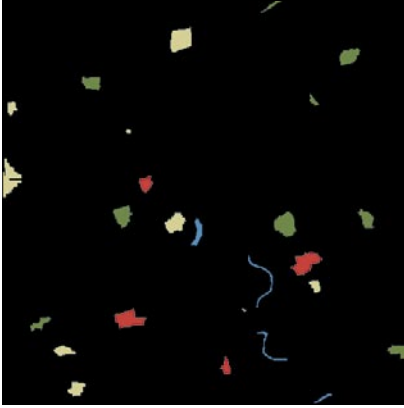
5. Click “OK” to close the class description again.

### Declaring sample objects

As already mentioned above, you will perform a classification using nearest neighbor. Training or test areas (TTA) can be imported into eCognition either by manually selecting them or by means of the so-called TTA mask. In eCognition, image objects which function as samples for a nearest neighbor are referred to as samples or sample objects.

1. Choose “Load TTA Mask...” from the “Samples” menu.
2. Load the file “TTAMask\_dessau.asc” as training and test areas mask.
3. Load the file “TTA-Mask\_dessau.txt” as a conversion table.
4. Answer the question of whether you want to create classes from the conversion table with “No.”






The TTA mask is now shown in the view window.

5. Open the conversion table by selecting “Edit Conversion Table” in the “Samples” menu.
6. Select “Link by name” to automatically link the classes from the TTA mask to the classes of the class hierarchy and close the dialog.
7. Choose “Create Samples from TTA Mask” from the “Samples” menu.

8. Click “OK” in the following two dialogs.

You now have created a segmented image with the sample objects displayed in the respective class color.

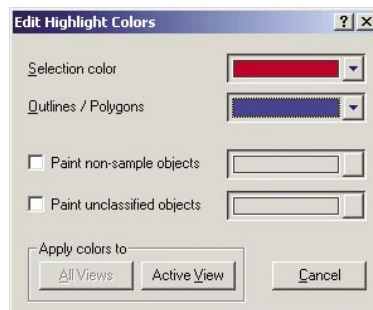
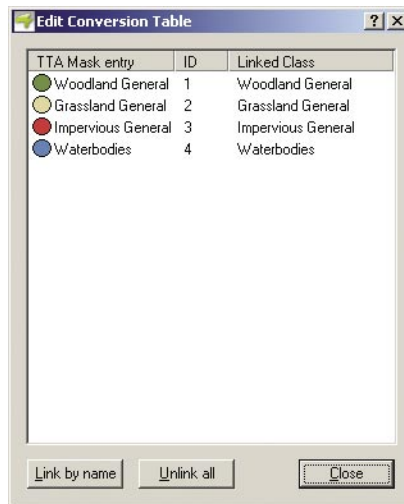
9. To get a better view of the sample objects in contrast to all other image objects, you can change the color display of nonsample objects by opening the edit highlight colors dialog and changing the display mode for nonsample objects.

10. Select the menu item “Samples > Open Sample Editor...” or click  in the tool bar.

The sample editor is now open. This is the central tool for working with sample objects.

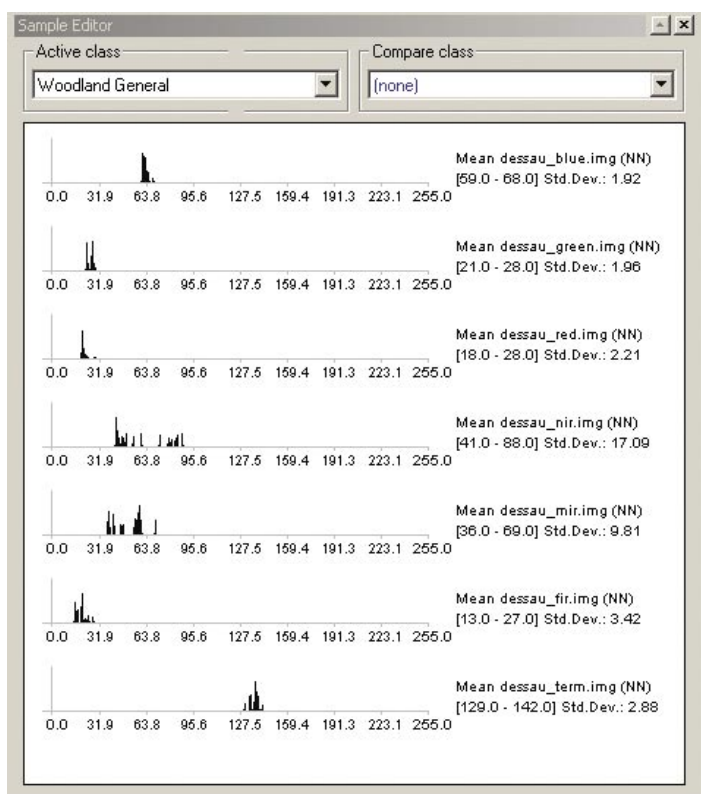
11. Select *Woodland General* in the “Active class” field.

Each of the small columns in the histograms represents the feature value of one or more samples. As a whole, the sample editor shows the feature signa-





ture of the class selected in the field “Active class.” The sample objects represented by the columns are the ones just created in the TTA mask.



## Classifying image objects

After a number of sample objects have been declared as initial information for a nearest neighbor classification, you can start the classification process.

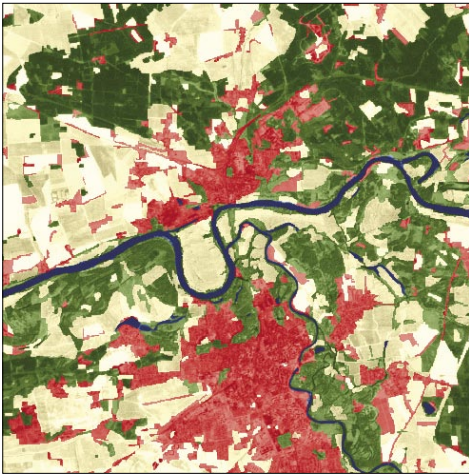
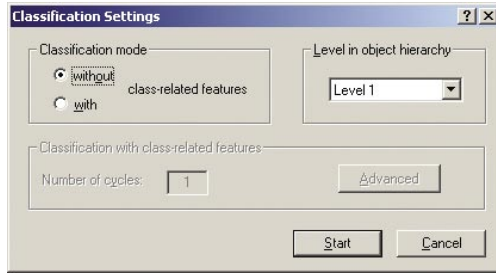
1. Choose “Classify...” from the “Analysis” menu to open the “Classification Settings” dialog.

First, you will classify without class-related features. In doing so, only the base classes will contain class-related features.

- Click “Start” to initiate the classification process.

Once the classification process is finished, the result is shown in the view window.

You can obtain information about the classification of an image object by moving the mouse over it. A tool tip shows you the actual assignment value. Again, detailed information can be obtained in the image object information dialog. Click any image object to view its feature information as well as classification evaluation. There are three basic functions in the “Image Object Information” dialog:



The “Features” folder gives information about all the features of an object; the “Classification” folder contains information about the current classification of the object as well as its alternative assignments. The “Class Evaluation” folder gives detailed information about the evaluation of one class for one object.

- Switch among the different folders to get an impression of the available information.

### Having a look at the descriptions of child classes

As mentioned above, the class descriptions of the child classes use class-related features. Take a look at such a class description now.

- Open the class description of *Urban Grassland* by double-clicking that class in the “Class Hierarchy” dialog.

The whole classification system in eCognition is based on fuzzy logic. The class description of each class is the key editor for the knowledge base.

Note that each class description is divided into two parts: contained and inherited features. “Contained” features are all those features which are directly contained in the class. They can only be edited in this specific class. “Inherited” features are features inherited from all parent classes of the selected class in the inheritance hierarchy. They will be changed along with changes made in the parent classes.

Classifiers, namely membership functions (in this case using the feature “Rel. area of...”) or nearest neighbor, provide the initial information for further evaluation of the class of a specific object. Each classifier returns a fuzzy

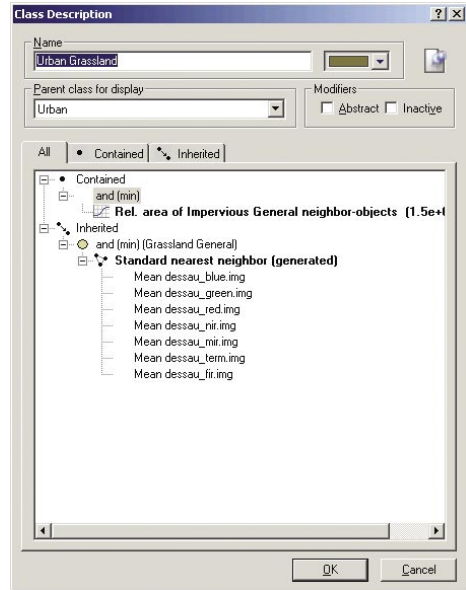
value between 0 (no assignment at all) and 1 (full assignment). In addition, fuzzy logic operators such as “and (min),” “or (max),” or “mean (arithm)” link different terms in order to come up with a superior fuzzy result, again returning a fuzzy membership value of between 0 and 1.

Classifiers, namely membership functions (in this case using the feature “Rel. area of...”) or nearest neighbor, provide the initial information for further evaluation of the class of a specific object. Each classifier returns a fuzzy value between 0 (no assignment at all) and 1 (full assignment). In addition, fuzzy logic operators such as “and (min),” “(max),” “mean (arithm)” or “mean (geo.)” link different terms in order to come up with a superior fuzzy result, again returning a fuzzy membership value of between 0 and 1.

In this case, the standard nearest neighbor has been inherited from the parent class *Grassland General*. Check the hierarchy to verify this. The selected feature for the distinction between *Rural* and *Urban Grassland* objects is the relative area of objects of the class *Impervious General* within a perimeter of 10 pixels around the object in question.

2. Double-click the feature “Rel. area of Impervious General Neighbor-objects (10)” to open the “Membership Function” dialog.

The following information is given in the dialog: an object will receive a membership value of 1 for *Urban Grassland*, if the relative area of objects classified as *Impervious Ge-*

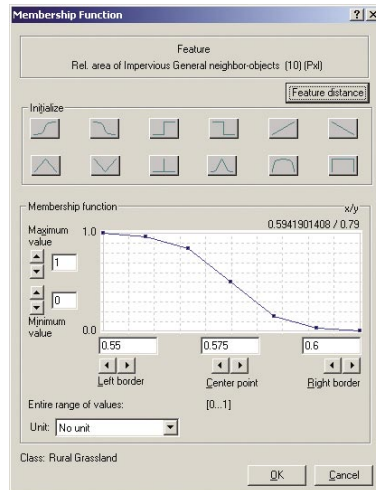
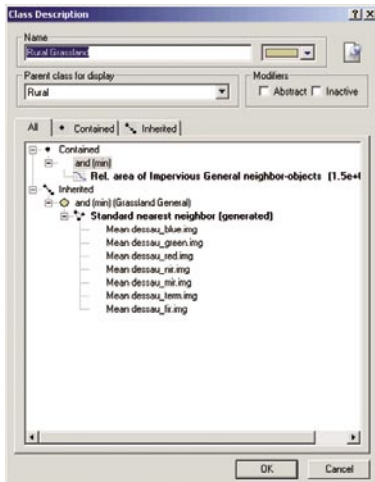
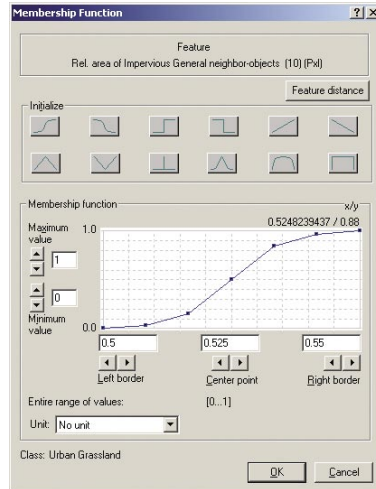


neral (in relation to all objects) in its neighborhood is higher than 0.55. If it is 0.5 or lower, the returned value will be 0. All feature values between 0.5 and 0.55 are translated into a membership value between 0 and 1 according to the blue function slope.

3. Close the membership function and the class description.
4. Open the class description and membership function of *Rural Grassland*.

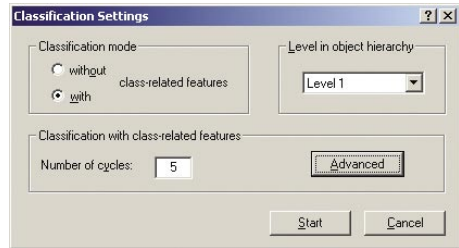
Note that the same feature has been used as for *Urban Grassland*. But this time, the membership function has been edited inversely, so that a *Grassland General* object is definitely classified *rural* if the relative area of objects classified as *Impervious General* within a 10 pixel distance is less than 0.55.

5. Close the membership function and the class description.





## Classifying the image using class-related features

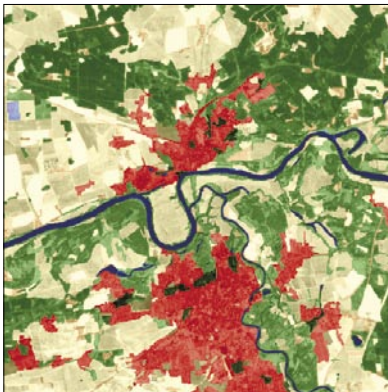
The classification so far used only non-class-related features. Due to the logic used in eCognition, all classes containing class-related features were deactivated in this classification run. To also use the classes containing class-related features, the settings for the classification have to be changed from “without class-related features” to “with class-related features.”



1. Open the “Classification Settings” dialog again (menu item “Classification > Classify... “).
2. Enable classification “with class-related features” by activating the appropriate classification mode.
3. Change the “number of cycles” to 5.
4. Click “Start.”

The classification result is displayed in the view window. Use the green arrows   in the tool bar to navigate through the groups hierarchy (meaningful semantic groups).

5. Move the mouse over an image object to obtain information about its classification, as explained earlier in this chapter.



The classification of the LANDSAT TM subset is now finished.

## Summary

In this chapter you

- loaded raster data for multiresolution segmentation to create image objects,
- got an impression of how to obtain information transported by separate image objects and by the image as a whole,
- loaded a class hierarchy and became familiar with its structure and its purpose as the knowledge base in eCognition,
- declared sample objects to help you with your classification after editing them in the sample editor,
- applied two different methods for the classification of parent and child classes: without class-related features, and with class-related features. The nearest neighbor was used as a classifier for the parent classes; the child classes were classified with the help of membership functions,
- had a closer look at the class description, its structure and classifiers in order to subsequently familiarize yourself with editing classes and knowledge bases.

You have successfully worked your way through this introductory chapter of the user guide. Hopefully, the water was not too deep in this, your first experience of eCognition, and you now feel compelled to delve deeper into eCognition's world of object oriented image analysis!

# 4 CONCEPTS & METHODS

A. The approach and basic concepts .....	57
What is object oriented image analysis? .....	57
Basic aspects in image interpretation.....	59
Segmentation .....	62
Classification – classifiers and methods.....	65
Fuzzy classification systems.....	66
 B. How is object oriented image analysis realized in eCognition?.....	71
Multiresolution segmentation of image objects .....	73
Handling of vector information.....	86
Fuzzy classification in eCognition .....	91
The classification process.....	101
Feature overview .....	103
Features and terms for the fuzzy class description.....	107
Classification-based segmentation and correction of image objects.....	145
Multisource data fusion.....	149
Classification evaluation in eCognition.....	152
Aspects of human perception .....	160
References .....	163

## A. The approach and basic concepts

In this chapter you become acquainted with the basics of object oriented image analysis and general aspects of image interpretation, and an overview will be given about segmentation and classification methods, with a particular focus on fuzzy classification.

### What is object oriented image analysis?

eCognition is based on an object oriented approach to image analysis. In contrast to traditional image processing methods, the basic processing units of object oriented image analysis are image objects or segments, and not single pixels. Even the classification acts on image objects. One motivation for the object oriented approach is the fact that the expected result of many image analysis tasks is the extraction of real world objects, proper in shape and proper in classification. This expectation cannot be fulfilled by common, pixel-based approaches.

Although eCognition is of course a specific combination of different contributing procedures, there are some basic characteristic aspects of the underlying object oriented approach, independent of the particular methods.

Directly connected to the representation of image information by means of objects is the networking of these image objects. Whereas the topological relation of single, adjacent pixels is given implicitly by the raster, the association of adjacent image objects must be explicitly worked out, in order to address neighbor objects. In consequence, the resulting topological network has a big advantage as it allows the efficient propagation of many different kinds of relational information.

Each classification task addresses a certain scale. Thus, it is important that the average resolution of image objects can be adapted to the scale of interest. Image information can be represented in different scales based on the average size of image objects. The same imagery can be segmented into smaller or larger objects, with considerable impact on practically all information which can be derived from image objects. Thus, specific scale information is accessible.

Furthermore, it is possible to represent image information in different scales simultaneously by different object layers. Bringing different object layers in relation to each other can contribute to the extraction of further valuable information.

This can be achieved, for instance, by a hierarchical networking and representation of image objects. Besides its neighbors, each object also knows its sub-objects and super-



objects in such a strict hierarchical structure. This allows precise analysis of the substructures of a specific region and is not possible without a strict hierarchical structure. Furthermore, based on sub-objects, the shape of super-objects can be changed.

It should be emphasized in this context that even single pixels or single-pixel objects are a special case of image objects. They represent the smallest possible processing scale.

An astonishing characteristic of object oriented image analysis is the multitude of additional information which can be derived based on image objects. Beyond tone, this is shape, texture, context, and information from other object layers. Using this information, classification leads to better semantic differentiation and to more accurate and specific results. In a conceptual perspective, the available features can be distinguished as:

- intrinsic features: the object's physical properties, which are determined by the pictured real world and the imaging situation – basically sensor and illumination. Such features describe color, texture and form of the objects.
- topological features: features which describe the geometric relationships between the objects or the whole scene, such as being left, right, or in a certain distance to a certain object, or being in a certain area within the image.
- context features: features which describe the objects' semantic relationships, e.g., a park is almost 100 % surrounded by urban areas.

Based on classification the processing of image objects can be done locally in a specific way. From the moment that an object is classified as *forest*, for instance, local intelligence can be applied and in principle everything which is done from that moment with this object and its networked environment can be done with a *forest* logic. Instead of processing all areas of an image with the same algorithms, a differentiated procedure can be much more appropriate. This is a specific strength of object oriented image analysis.

Characteristic for the object oriented approach is, finally, a circular interplay between processing and classifying image objects. Based on segmentation, scale and shape of image objects, specific information is available for classification. In turn, based on classification, specific processing algorithms can be activated. In many applications the desired geoinformation and objects of interest are extracted step by step, by iterative loops of classifying and processing. Thereby, image objects as processing units can continuously change their shape, classification and mutual relations.

Similar to human image understanding processes, this kind of circular processing results in a sequence of intermediate states, with an increasing differentiation of classification and an increasing abstraction of the original image information. On each step of abstraction, new information and new knowledge is generated and can be used benefi-

cially for the next analysis step. Thereby, the abstraction not only concerns shape and size of image objects, but also their semantics. It is interesting that the result of such a circular process is by far not only a spatial aggregation of pixels to image regions, but also a spatial and semantic structuring of the image's content. Whereas the first steps are more data driven, more and more knowledge and semantic differentiation can be applied in later steps. The resulting network of classified image objects can be seen as a spatial, semantic network. After successful analysis, much interesting, additional information can be derived just by processing requests over this network.

The object orient approach is in principal independent of the specific segmentation and classification techniques. However, the right choice of processing methods can add a lot of power to the procedure, and the right training and classification methods can give the user the full advantage of the approach's potential. Find more about how the object oriented approach to image analysis has been realized in eCognition in part B of this chapter.

## Basic aspects in image interpretation

### About scale in image processing

Scale is a crucial aspect of image understanding. For remote sensing and GIS applications, see for example the descriptions in Quattrochi, D.A. & Goodchild, M.F., 1997. Although in the domain of remote sensing a certain scale is always presumed by pixel resolution, the desired objects of interest often have their own inherent scale. Scale determines the occurrence or nonoccurrence of a certain object class. The same type of objects appears differently at different scales. Vice versa, the classification task and the respective objects of interest directly determine a particular scale of interest.

Note the difference between scale and resolution: as resolution commonly expresses the average size of area a pixel covers on the ground, scale describes the magnitude or the level of abstraction on which a certain phenomenon can be described. Thus, studying an image from different levels of scale instead of at different resolutions eases its analysis. The relevant typical remote sensing examples are urban areas or ecosystems. Assuming you are viewing a large, high-resolution image of a city. When looking very close you can recognize single houses, buildings, roads and other urban objects. If you enlarge your viewing distance, you cannot discover single buildings, but rather different housing areas or even quarters. They typically can be distinguished by different textures and by different size and shape, too. The quarter's texture comprises its objects and structures on a smaller scale – houses, roads, gardens etc. – and it is especially determined by their tone, shape, and also by their topological relationships.

When placing the image on a wall and observing it from the distance of a few steps, you might just discover the city itself and maybe some surrounding agricultural areas or forests. You might realize that the surrounding agricultural areas and forests are of comparable size to the city. You will agree that cities, forests and agricultural areas are of similar scale. The same is valid for trees and houses in another magnitude. In an abstract conceptualization, the existence of structure on different scales simultaneously is an aspect of fractal geometry and topology in nature and, thus, also in imagery.

Moreover, there is a kind of hierarchy in both our lingual description and conceptualization of different phenomena and the structure of the real world objects which we are detecting. This hierarchy is obviously determined by scale: abstracting houses, buildings, roads and other objects, one obtains settlement areas or even quarters. The aggregation of several settlement areas yields a town. Ecosystems show analogous patterns: combining several trees builds a group of trees and combining more trees or groups of trees builds a forest.

Forests and towns seem to have a similar abstraction level. Both are of comparable scale and both are of high semantic abstraction. The hierarchical scale dependencies between the affected object classes are obvious: quarters are substructures of cities, and houses are substructures of quarters.

These hierarchical scale dependencies are implicitly self-evident in each observation and description of real world structures. However, reflecting, and especially explicit representation of, these patterns can add valuable information to automated image understanding methods. The possibility to classify houses in an urban area simply is much higher than if they were located in forests, for instance. Thus, in order to analyze an image successfully it is necessary to represent its content on several scales simultaneously and to explore the hierarchical scale dependencies among the resulting objects.

It is obvious that these relationships and dependencies cannot be analyzed by just changing the resolution of the imagery. This would, moreover, lead to the loss of a lot of useful information.

#### **Image semantics – mutual relations between image objects**

One of the most important aspects of understanding imagery is information about context. There are two types of contextual information: global context, which describes the situation of the image – basically, time, sensor and location – and local context, which describes the mutual relationships or the mutual meaning of image regions. It is obvious that the processing of context information is always consciously or subconsciously present in human perception and contributes essentially to its great capabilities.

In order to receive meaningful context information, image regions of the right scale must be brought into relation. This scale is given by the combination of classification task and the resolution of the image data.

Imagine for instance the classification task to identify parks in very high resolution imagery. A park is always a large, contiguous vegetated area. This different scale distinguishes parks from gardens. Additionally, parks are distinguished from pastures, for example by their embedding in urban areas. Single neighboring buildings are not a sufficient condition to describe parks. However, their neighborhood to single buildings is a suitable criterion for distinguishing gardens from pasture.

This simple example already shows how much the available context information depends on the scale of the structures which are brought into relation. This astonishing fact explains why it is so difficult or even impossible to describe meaningful context relations using pixel-based approaches. Only representing image information based on image objects of the appropriate scale enables one to handle image semantics. Additionally, in order to make image objects aware of their spatial context it is necessary to link them. Thus, a topological network is created.

This network becomes hierarchical when image objects of different scale at the same location are linked. Now each object knows its neighbors, its sub- and super-objects. This additionally allows a description of hierarchical scale dependencies. Together with classification and mutual dependencies between objects and classes, such a network can be seen as a spatial semantic network.

The fact that image understanding always means dealing with image semantics was until now not sufficiently covered by the capacity of digital image analysis, especially in the field of remote sensing.

#### Uncertainties and vagueness in remote sensing information extraction

Various types of uncertainty influence information extraction from remote sensing data. First of all, there are many factors which influence the processes of data acquisition, data processing and image generation, and which differ from scene to scene, even if the data comes from the same sensor. A very basic, inherent problem of earth observation data is that land cover can look different, depending on the season, time of day, light conditions and weather. Furthermore, the same type of objects appears highly differently depending on the sensor and the resolution.

The dependency between features and land cover or land use is mostly only roughly modeled and vagueness is inherent even in the concepts of land cover and land use. Sensor measurements – the basic source for image pixels – have limited radiometric resolution even after careful calibration of the instrument. The geometric resolution

in remote sensing – and in any data acquisition process – is limited as well. This effect leads to class mixture within one resolution cell: if a resolution cell covers water-land transition, the relevant pixel represents to some degree water and to some degree the land cover of the shore area.

The image generation process converts sensor measurements to image data. Additionally, these data have to be compressed to reduce requirements for archiving and data transmission. In most cases, these data processing steps cause artifacts and ambiguities which lead to additional uncertainty in the final image data.

Usually only vague concepts exist for land cover and land use. There is no exact threshold between densely and sparsely populated area, or between low and high vegetation. Whenever thresholds are defined in terms of numbers, they are mostly unsatisfactory idealizations of the real world and therefore subsequently lead to problems during classification and performance estimation of the classification.

Information retrieval from remote sensing databases is based to a large extent on vague knowledge. Especially important context information is typically only expressed in terms of vague linguistic rules. For example, if trees are “nearly completely” surrounded by urban area, they are assigned to the class *park*.

Furthermore, in many cases the desired information for a specific classification task is not, or not sufficiently, contained in the available image data. This can be caused by spatial or radiometric resolution, because the signal to noise ratio is too low, or simply because the sensor does not deliver different signals for the different types of information of interest.

If these uncertainties are not taken into account in information extraction, classification will not be robust and transferable. There are several approaches, so called soft classifiers, which take these uncertainties into account.

## Segmentation

Segmentation is the subdivision of an image into separated regions. For many years, procedures for image segmentation have been a main research focus in the area of image analysis. Many different approaches have been followed. However, few of them lead to qualitatively convincing results which are robust and applicable under operational settings. One reason is that segmentation of an image into a given number of regions is a problem with an astronomical number of possible solutions. The high number of degrees of freedom must be reduced to the one or the few solutions which satisfy the given requirements. Another reason is that in many cases regions of interest are heterogeneous; ambiguities arise and the necessary discerning information is not directly available. Requirements concerning quality, performance – size of data set and

processing time – and reproducibility can be fulfilled at the same time only by very few approaches.

In image segmentation the expectation is in many cases to be able to automatically extract the desired objects of interest in an image for a certain task. However, this expectation ignores the considerable semantic multitude that in most cases needs to be handled to successfully achieve this result, or it leads to the development of highly specified algorithms applicable to only a reduced class of problems and image data.

Of course there is a variety of methods for generating image objects which cannot be summarized here, and each of them has its advantages and disadvantages. Some are fully automated while others are semi-automatic. Giving a rough overview, according to recent research in image understanding, image segmentation methods are split into two main domains: knowledge driven methods (top-down) vs. data driven methods (bottom-up). In top-down approaches the user already knows what he wants to extract from the image, but he does not know how to perform the extraction. By formulating a model of the desired objects, the system tries to find the best method(s) of image processing to extract them. The formulated object model gives the objects' meaning implicitly. In bottom-up approaches the segments are generated based upon a set of statistical methods and parameters for processing the whole image. As such, bottom-up methods can also be seen as a kind of data abstraction or data compression. But, as with clustering methods, in the beginning the generated image segments have no meaning, they can better be called: image object primitives. It is up to the user to determine what kind of real world objects the generated image objects represent. The basic difference between both approaches is: top-down methods usually lead to local results because they just mark pixels or regions that meet the model description, whereas bottom-up methods perform a segmentation of the complete image. They group pixels to spatial clusters which meet certain criteria of homogeneity and heterogeneity.

In order to obtain image object primitives as basic processing units, the object oriented approach to image analysis requires complete segmentation of an image. Therefore, a rough overview is given of the most common bottom-up approaches to image segmentation.

Some of the simplest approaches are all types of global thresholding. The spectral feature space is separated into subdivisions, and pixels of the same subdivision are merged when locally adjacent in the image data. Typically, this method leads to results of relatively limited quality. Oversegmentation and undersegmentation – i.e., separating into units which are too small or merging regions that do not belong to each other – take place easily without good control of meaningful thresholds. Local contrasts are not considered or not represented in a consistent way and the resulting regions can widely differ in size.

Region growing algorithms cluster pixels starting from a limited number of single seed points. These algorithms basically depend on the set of given seed points and often suffer from a lack of control in the break-off criterion for the growth of a region.

Often used in operational applications are different types of texture segmentation algorithms. They typically obey a two-stage scheme [Mao & Jain 1992, Hofmann & al. 1998]: 1. In the modeling stage characteristic features are extracted from the textured input image and range from spatial frequencies [Hofmann & al. 1998], MRF models [Mao & Jain 1992, Panfwni & Healey 1995] and co-occurrence matrices [Haralick & al. 1973] to wavelet coefficients [Salari & Zing 1995], wave packets [Laine & Fan 1996] and fractal indices [Chaudhuri & Sarkar 1995]. 2. In the optimization stage features are grouped into homogeneous segments by minimizing an appropriate quality measure. This is most often achieved by a few types of clustering cost functions [Mao & Jain 1992, Hofmann & al. 1998, Manjunath & Chellappa 1991]. Although texture segmentation leads to reproducible, for specific applications excellent and sometimes high speed results, they are mostly only applicable to a limited number of types of image data, texture types and problems. Texture often must be very regular to be recognized. Results cannot be achieved on any chosen scale.

Common alternatives are knowledge-based approaches. They try to incorporate knowledge derived from training areas or other sources into the segmentation process [Gorte 1998]. These approaches mostly perform a pixel-based classification, based on clustering in a global feature space. Segments are produced implicitly after classification, simply by merging all adjacent pixels of the same class. In doing so, these approaches are typically not able to separate different units or objects of interest of the same classification. Furthermore, the information on which classification can act typically is limited to spectral and filter derivatives.

A further relatively common procedure is watershed segmentation [Wegner & al 1997]. It got its name from the manner in which the algorithm segments regions into catchment basins. Typically, the procedure first transforms the original data into a gradient image. The resulting gray tone image can be considered as a topographic surface. If we flood this surface from its minima and if we prevent the merging of the waters coming from different sources, we partition the image into two different sets: the catchment basins and the watershed lines. The catchment basins should theoretically correspond to the homogeneous gray level regions of this image. This method works for separating essentially convex and relatively smooth objects of interest that even may touch slightly in relatively homogeneous image data. When it works, it is convenient, fast and powerful. However, for remote sensing data, which typically contain a certain noise and not always strong contrasts, this method is typically not able to achieve appropriate results.

## Classification – classifiers and methods

Usually classifying means assigning a number of objects to a certain class according to the class's description. Thereby, a class description is a description of the typical properties or conditions the desired classes have. The objects then become assigned (classified) according to whether they have or have not met these properties/conditions. In terms of database language one can say the feature space is segmented into distinct regions which leads to a many-to-one relationship between the objects and the classes. As a result each object belongs to one definite class or to no class. Classic classifiers in remote sensing (e.g., maximum-likelihood, minimum-distance or parallelepiped) thereby assign a membership of 1 or 0 to the objects, expressing whether an object belongs to a certain class or not. Such classifiers are usually also called hard classifiers since they express the objects' membership to a class only in a binary manner. In contrast, soft classifiers (mainly fuzzy systems and/or Bayes classifiers) use a degree of membership/a probability to express an object's assignment to a class. The membership value usually lies between 1.0 and 0.0, where 1.0 expresses full membership/probability (a complete assignment) to a class and 0.0 expresses absolutely nonmembership/improbability. Thereby the degree of membership/probability depends on the degree to which the objects fulfill the class-describing properties/conditions. A main advantage of these soft methods lies in their possibility to express uncertainties about the classes' descriptions. It makes it also possible to express each object's membership in more than just one class or the probability of belonging to other classes, but with different degrees of membership or probabilities. With respect to image understanding these soft classification results are more capable of expressing uncertain human knowledge about the world and thus lead to classification results which are closer to human language, thinking and mind. In other words: soft classifiers are more honest than their hard counterparts. But unfortunately many applications using land use or land cover information are unable to handle soft classification results. Thus, soft classification results must be hardened, which can lead to shammed classification truths and accuracies.

Regarding classification methods, they can basically be separated into supervised and unsupervised methods. While supervised methods ask the user how the desired classes look, unsupervised methods are almost user independent. They rather can be seen as statistical grouping methods, sorting objects by their properties into clusters with similar properties. Since unsupervised methods work almost automatically, supervised methods have to be trained by the user – usually either by taking samples or by describing the classes' properties. Therefore, the class-describing information must be as accurate, representative and complete as possible, which is in most cases effectively impossible. Hence, a class description can only be a general estimation of the desired class's properties. Estimating the properties also means assuming a more or less known uncertainty about the class description or a known vagueness about the properties' measured values.



Formulating these uncertainties can only be achieved using soft classifiers. Comparing unsupervised and supervised classification methods, each has its advantages and disadvantages. Unsupervised methods are noticeably faster than supervised ones, but since they are just a special way of sorting algorithms, their results have to be interpreted by the user – which can be tough in some cases and lead to numerous repetitions of the classification with slightly adjusted parameters. Another advantage of unsupervised classifiers is their ability to analyze the objects' statistics completely and systematically. Thus, the results of an unsupervised classification can give useful indications of detectable classes, but, in general, formulating uncertainty is only possible if related to the classification parameters, not to the classes and their properties themselves. In contrast, supervised classification methods can be more labor intensive since the user has to describe the classes' properties either explicitly or by taking samples as typical representatives. Their advantages are firstly their usually higher quality and a priori counting and naming of the classes and secondly the possibility to formulate explicit class-related uncertainty. In cases of misclassifications especially the latter point eases the investigation for these reasons. But the class descriptions themselves are also easier to understand since they should be a result of human reasoning and thus easier to investigate.

### Fuzzy classification systems

The most powerful soft classifiers are classifiers based on fuzzy systems. Fuzzy logic is a mathematical approach to quantifying uncertain statements. The basic idea is to replace the two strictly logical statements “yes” and “no” by the continuous range of  $[0 \dots 1]$ , where 0 means “exactly no” and 1 means “exactly yes.” All values between 0 and 1 represent a more or less certain state of “yes” and “no.” Thus, fuzzy logic is able to emulate human thinking and take into account even linguistic rules. Fuzzy classification systems are well suited to handling most vagueness in remote sensing information extraction. Parameter and model uncertainties are considered as using fuzzy sets defined by membership functions. Instead of the binary “true” and “false” multivalued fuzzy logic allows transitions between “true” and “false.” Additionally, there are more or less strict realizations of the logical operations “and” or “or.”

The output of a fuzzy classification system is a fuzzy classification, where the membership degree to each land cover or land use class is given for each object. This enables detailed performance analysis and gives insight into the class mixture for each image object. This is a major advantage of soft classification. The maximum membership degree determines the final classification to build an interface to crisp (boolean) systems. Fuzzy systems consist of three main steps, fuzzification, fuzzy rule base and defuzzification, which are briefly described in the following.

## Fuzzification

Fuzzification describes the transition from a crisp system to a fuzzy system. It assigns a membership degree (membership value) between 0 and 1 to each feature value. The membership value is defined by a so-called membership function. Depending on the shape of the function, the transition between “yes” and “no” can be crisp (for a rectangular function) or fuzzy (see fig. 1, set M).

The set of feature values which produce a membership value higher than 0 can be called a fuzzy set. In general, the broader the membership function, the vaguer the underlying concept; the lower the membership values, the more uncertain is the assignment of the set.

The combining of different features within a fuzzy system is always done after the feature is fuzzified. Therefore, all input values for fuzzy combinations are in the range between 0 and 1, independent of the dynamic of the originally crisp features. This simplifies working in a high-dimensional feature space with different dynamics and features of various types, e.g., backscatter from different sensors, geographic information, texture information and hierarchical relations.

For successful classification a deliberate choice of membership function is crucial. This allows the introduction of expert knowledge into the system. The better the knowledge about the real system is modeled by the membership functions, the better the final classification result.

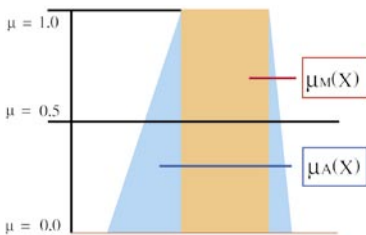


Fig. 1: rectangular and trapezoidal membership functions on feature  $x$  to define crisp set  $M$  (red) and fuzzy set  $A$  (blue)  $\mu_A(X)$  over the feature range  $X$ .

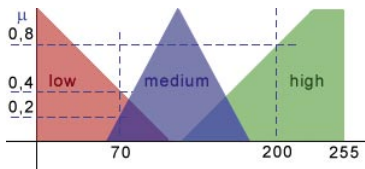


Fig. 2: The membership functions on feature  $x$  define the fuzzy set low, medium and high for this feature.

It is possible to define more than one fuzzy set on one feature, e.g., to define the fuzzy sets *low*, *medium* and *high* for one object feature. The more the memberships overlap, the more objects are common in the fuzzy sets and the vaguer the final classification. Fig. 2 shows three fuzzy sets defined for the feature  $x$ : *low*, *medium* and *high*. They are characterized by overlapping triangular membership functions. For an image object with a feature value of  $x = 70$ , the membership to class *low* is 0.4, to class *medium* is 0.2 and to class *high* is 0.0. If the feature value  $x$  equals 200, the membership to the classes is 0.0, 0.0, 0.8, respectively.

### Fuzzy rule base

A fuzzy rule base is a combination of fuzzy rules, which combine different fuzzy sets. The simplest fuzzy rules are dependent on only one fuzzy set.

Fuzzy rules are “if – then” rules. If a condition is fulfilled, an action takes place. Referring to fig. 2, the following rule could be defined: “If” feature  $x$  is low, “then” the image object should be assigned to land cover  $W$ . In fuzzy terminology this would be written: If feature  $x$  is a member of fuzzy set *low*, then the image object is a member of land cover  $W$ . According to the definition in fig. 2, in case feature value  $x = 70$ , the membership to land cover  $W$  would be 0.4, in case  $x = 200$ , the membership to land cover  $W$  would be 0.

To create advanced fuzzy rules, fuzzy sets can be combined. An operator returns a fuzzy value that is derived from the combined fuzzy sets. How this value is derived depends on the operator. The basic operators are “and” and “or.” “and” represents the minimum, meaning that the minimum value of all sets defines the return value. “or” represents the maximum value, meaning that the maximum value of all sets defines the return value. The results are very transparent and ensure independence of the sequence of logic combinations within the rule base (A “and” B gives the same result as B “and” A). In addition a hierarchic structure following common logic (e.g., A “or” (B “and” C) equals (A “or” B) “and” (A “or” C)) can be created easily.

A fuzzy rule base delivers a fuzzy classification, which consists of discrete return values for each of the considered output classes (see fig. 3). These values represent the degree of class assignment.

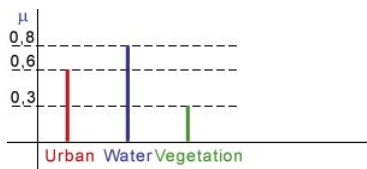


Fig. 3: fuzzy classification for the considered land cover classes urban, water and vegetation. The image object is a member of all classes to various degrees [ $\mu_{urban}(object) = 0.6$ ,  $\mu_{water}(object) = 0.8$ ,  $\mu_{vegetation}(object) = 0.3$ ]

Please consider that fuzzy classification gives a possibility for an object to belong to a class, while classification based on probability give a probability to belong to a class. A

possibility gives information on a distinct object. A probability relies on statistics and gives information on many objects. Whereas the probability of all possible events adds up to one, this is not necessarily true for possibilities.

The higher the return value for the most possible class, the more reliable the assignment. In the example above, the membership to water  $\mu_{\text{water}}(\text{object}) = 0.8$  is rather high and in most applications this object would therefore be assigned to the class *Water*. The minimum membership value an object needs to have in order to be assigned to a class can be defined. See [Functional Guide > Classification Basics > The classification process](#). The bigger the difference between highest and second highest membership value, the clearer and more stable the application. Classification stability and reliability can be calculated and visualized within eCognition. See [Functional Guide > Accuracy Assessment and Statistics](#).

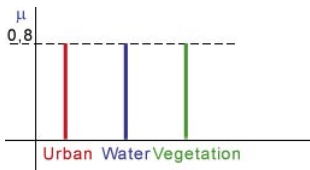


Fig. 4: fuzzy classification for the considered land cover classes Urban, Water and Vegetation. The equal membership degrees indicate an unstable classification between these classes for the considered image object. If the land cover classes can usually be distinguished on the data set, this result indicates that all land cover classes are within the image object to a similar degree. The high membership value shows that the assignment to this class mixture is reliable  $\mu_{\text{urban}}(\text{object}) = \mu_{\text{water}}(\text{object}) = \mu_{\text{vegetation}}(\text{object}) = 0.8$ .

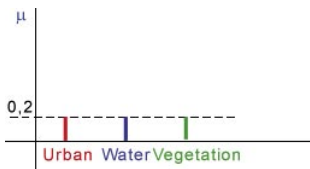


Fig. 5: fuzzy classification for the considered land cover classes Urban, Water and Vegetation. The equal membership degrees again indicate an unstable classification between these classes for the considered image object, as in fig. 4. However, the low membership value  $\mu_{\text{urban}}(\text{object}) = \mu_{\text{water}}(\text{object}) = \mu_{\text{vegetation}}(\text{object}) = 0.2$  indicates a highly unreliable assignment. Assuming a threshold of a minimum membership degree of 0.3, no class assignment will be given in the final output.

Fuzzy classification with its enhanced analysis of classification performance of class mixture, classification reliability and stability is possible, because fuzzy classification is one powerful approach to soft classification. The results of the fuzzy classification are an important input for information fusion in current and future remote sensing systems with multidata sources. The reliability of class assignments for each sensor can be used to find the most possible and probable class assignment. A solution is possible, even if there are contradictory class assignments based on different sensor data, e.g., optical sensors are regarded as being less reliable than radar sensors if there is a heavy fog.

### Defuzzification

To produce results like maps for standard land cover and land use applications, the fuzzy results have to be translated back to a crisp value, which means that an object

is either assigned to a class or not. In the classification step, usually the class with the highest membership degree is chosen. For this type of output, the rich measures of uncertainty of the fuzzy classification are lost.

Defuzzification is the reverse process of fuzzification. It delivers a crisp classification. If the membership degree of a class is below a certain value, no classification is performed to ensure minimum reliability.

Further information on fuzzy systems in image analysis and remote sensing can be found in the following references.

## B. How is object oriented image analysis realized in eCognition?

eCognition is consequently built upon an object oriented processing of image information, as described in the first section of this chapter. However, specific methods for segmentation and classification have been chosen in order to endorse the approach, increase its capabilities and optimize the handling. Different procedures complete the whole with data input and output, vectorization, training, interfaces for information, accuracy assessment and statistics.

For initial segmentation it turned out that practically all known segmentation procedures do not fulfill the requirements concerning the need for image object primitives as qualitatively good abstractions of the original image information, in any chosen resolution and as appropriate building blocks for further classification and processing. eCognition therefore uses a segmentation procedure called multiresolution segmentation which has been developed by Definiens Imaging.

Based on the possibility to generate image object primitives in any chosen scale, eCognition allows the production of more than one object level and the connection of these levels in a hierarchical manner. Embedded in this hierarchical network of image objects, each object knows its adjacent objects, its sub-objects and its super-object. By connecting the objects vertically, access to scale and advanced texture properties is possible. The object hierarchy allows representation of image information in different scales simultaneously.

Special vectorization algorithms can be applied and vector information added to the image objects. Doing so, the image objects in eCognition come in a simultaneous raster and vector representation. The polygons are used to display outlines to compute shape features and export results in vector format.

The whole classification domain is based on fuzzy logic. Fuzzy logic supports an intuitive and transparent editing and handling of even complex rule sets. The frame for the knowledge base for classification is the class hierarchy, which contains all classes of the classification scheme.

Each class can be described by fuzzy rules, which base either on one-dimensional membership functions or on a nearest neighbor classifier which can even operate on a multidimensional feature space. Both are supervised classification methods. While the first can be edited directly and enable the user to formulate knowledge about the image content, the latter needs appropriate sample objects to determine the desired class's properties. Samples can be selected manually (click and classify) or based on training areas

masks. Different classifiers can be combined within one class description using fuzzy logic, which provides different operators such as “and” and “or,” for instance. Classification results can be differentiated and improved by using semantic context information: as soon as objects are classified according to their intrinsic and topological features, classification can be refined using semantic features – mostly by describing neighborhood relationships or the composition of sub-objects.

The class hierarchy supports semantic grouping of classes. This can be used to assign classes of different attributes to a common class of superordinated semantic meaning. In this case, the superior class does not need its own explicit class description. *urban green* and *urban impervious* can be grouped to the class *urban*, for instance. A special advantage in this case is to express context relations to the superior class: “embedded in urban” addresses *urban green* as well as also *urban impervious* objects.

Additionally, the class hierarchy allows a grouping of classes for the purpose of inheritance of class descriptions to child classes. A class such as grassland can be differentiated by inheriting its class description to child classes such as *urban green* and *agricultural grassland*, for instance. This gives structure to the knowledge base: the level of detail of a class description rises, the deeper the hierarchy branches.

With these possibilities the class hierarchy allows the efficient creation of a well-structured knowledge base of astonishing semantic richness. Together with fuzzy classification this adds a lot of power to the object oriented approach to image analysis. Note: class hierarchy is another thing and independent of image object hierarchy.

At the end the objects’ shape can be improved by classification and using knowledge-based segmentation. Usually this leads to new image objects with new properties and semantic relationships, which in turn can be classified according to their newly generated features.

Find in the following a description of all essential methods, including the description and definition of all features for classification.

## Multiresolution segmentation of image objects

### Requirements

Segmentation is not an aim in itself. The purpose of image analysis can be a land cover / land use classification or the extraction of objects of interest. However, objects of interest can in many cases be of considerable heterogeneity; see for instance the roofs in fig. 1. A segmentation procedure following a relatively general homogeneity criterion will in most cases not be able to directly extract the final areas or objects of interest. For the object oriented approach to image analysis in eCognition, image objects resulting from a segmentation procedure are therefore intended to be rather image object primitives, serving as information carriers and building blocks for further classification or other segmentation processes. In this sense, the best segmentation result is one that provides optimal information for further processing.



Fig. 1: high resolution airborne data. Note the considerable heterogeneity of objects of interest, such as roofs.

For this purpose multiresolution segmentation was developed, a new, patented segmentation procedure. It allows the largely knowledge-free extraction of homogeneous image object primitives in any chosen resolution, especially taking into consideration local contrasts. It generally can be applied to a very large range of data types; it works on an arbitrary number of channels simultaneously and is especially suited for textured or low contrast data such as radar or VHR images.

In order to receive optimal raw material for object oriented image analysis, the following aims were defined at the beginning of the development of multiresolution segmentation:

- A segmentation procedure should produce highly homogeneous segments for the optimal separation and representation of image regions.
- Since each image analysis problem deals with structures of a certain spatial scale based on specific data, the average size of image objects must be adaptable to the scale of interest.



- Almost all attributes of image objects – tone, texture, form and relations to adjacent regions – are more or less scale-dependent. Only structures of similar scale are of comparable quality and have comparable attributes. For this reason resulting image objects should be of more or less same magnitude.
- The segmentation procedure should be universal and applicable to a large number of different data types and problems.
- Segmentation results should be reproducible.
- Earth observation often produces large data sets. The segmentation procedure should therefore be as fast as possible.

A strong and experienced source for the evaluation of segmentation techniques is the human eye. By applying segmentation procedures to the automation of image analysis, the activity of visual digitizing is replaced. No segmentation result – even if it is quantitatively evaluated – will be fully convincing if it does not satisfy the human eye. Consistent handling of local contrasts is a prerequisite for reaching this goal, as is the segmentation of image regions of a more or less similar dimension, and finally, for producing image objects of minimal border smoothness, even at the cost of spectral homogeneity in strongly textured data.

At first view self-evident, however in reality nontrivial expectation is reproducibility of segmentation results for the same region, even if it appears in different subsets of the same scene. The most common procedures for segmentation have difficulties at this point. They perform optimisation and separation in a global feature space. The global feature space however depends on the particular subset of the scene. The same image area being part of different subsets will segment differently by means of such procedures, always dependent on the particular feature space of the respective subset.

Definite measurement and comparability of segmentation results is a necessary prerequisite for the evaluation of a segmentation technique. Given a certain definition of heterogeneity for image objects and a certain average size of image objects, possible criteria are:

- (1) Average heterogeneity of image objects should be minimized; or
- (2) Average heterogeneity of image objects weighted by their size should be minimized.

For the development of multiresolution segmentation, preference was given to the second criterion. Without changing its meaning it can be reformulated as:

- (2) Average heterogeneity of pixels should be minimized. The heterogeneity of the image object to which it belongs is assigned to each pixel.

This second formulation emphasizes the requirement that each part of an image must contribute equally to the evaluation of the heterogeneity of the segmentation results.

The result of a segmentation process in this conceptualisation leads to image object primitives as a first approximation to image objects of the real world, concerning a given problem. However, this approximation should be a universal high-quality solution applicable to many problems and even work on textured image data of arbitrary type.

### The procedure

Multiresolution segmentation is a bottom up region-merging technique starting with one-pixel objects. In numerous subsequent steps, smaller image objects are merged into bigger ones. Throughout this pairwise clustering process, the underlying optimization procedure minimizes the weighted heterogeneity  $nh$  of resulting image objects, where  $n$  is the size of a segment and  $h$  an arbitrary definition of heterogeneity. In each step, that pair of adjacent image objects is merged which stands for the smallest growth of the defined heterogeneity. If the smallest growth exceeds the threshold defined by the scale parameter, the process stops. Doing so, multiresolution segmentation is a local optimisation procedure.

To achieve adjacent image objects of similar size and thus of comparable quality, the procedure simulates the even and simultaneous growth of segments over a scene in each step and also for the final result. Thus, the procedure starts at any point in the image with one-pixel objects. A treatment sequence based on a binary counter guarantees a regular spatial distribution of treated objects. However, for obvious reasons, such a sequence contains a stochastic, historical element.

### Computation of the heterogeneity criterion

For the description of spectral or color heterogeneity in eCognition the sum of the standard deviations of spectral values in each layer weighted with the weights for each layer  $w_c$  are used:

$$1) \ h = \sum_c w_c \cdot \sigma_c$$

In many cases the exclusive minimization of spectral heterogeneity leads to branched segments or to image objects with a fractally shaped borderline, see fig. 2. This effect is even stronger in highly textured data, such as radar data.



Fig. 2: segmentation of a LANDSAT scene with exclusive optimization of spectral homogeneity

For this reason it is useful in most cases to mix the criterion for spectral heterogeneity with a criterion for spatial heterogeneity, in order to reduce the deviation from a compact or smooth shape. Heterogeneity as deviation from a compact shape is described by the ratio of the de facto border length  $l$  and the square root of the number of pixels forming this image object.

$$2) \ h = \frac{l}{\sqrt{n}}$$

A further possibility of describing shape heterogeneity is the ratio of the de facto border length  $l$  and the shortest possible border length  $b$  given by the bounding box of an image object parallel to the raster.

$$3) \ h = \frac{l}{b}$$

The application of this definition of heterogeneity optimises the smoothness of the shape of resulting image objects. Fig. 3 shows a segmentation result that was gained by mixing these two shape criteria with the spectral criterion. The image objects are of a more compact form and show much smoother edges. Although they might not be as homogeneous with respect to color, they are more satisfying to the human eye.



Fig. 3: segmentation of a LANDSAT scene with optimization of spectral homogeneity together with shape homogeneity

These three criteria for heterogeneity can be applied in a mixed form (see also dialog box “Multiresolution Segmentation”). Criteria 2) and 3) are additionally summarized to a general shape criterion. Especially in strongly textured data, such as radar, the shape criterion helps to avoid a fractal shaping of objects. In the following, you find how these parameters are transformed into fusion values which determine the fit of two adjacent segments.

In order to determine the outcome of the segmentation algorithm, the user can define several parameters, like the scale parameter, the single layer weights and the mixing of the heterogeneity criterion concerning tone and shape. A comparison of a fusion value with the scale parameter defines the break-up criterion. As mentioned above, the scale parameter is a measure for the maximum change in heterogeneity that may occur when merging two image objects. Internally, this value is squared and serves as the threshold which terminates the segmentation algorithm. When a possible merge of a pair of image objects is examined, a fusion value between those two objects is calculated and compared to the squared scale parameter.

The heterogeneity criterion consists of two parts: a criterion for tone and a criterion for shape. The spectral criterion is the change in heterogeneity that occurs when merging two image objects as described by the change of the weighted standard deviation of the spectral values regarding their weightings. The shape criterion is a value that describes the improvement of the shape with regard to two different models describing ideal shapes.

The overall fusion value  $f$  is computed based on the spectral heterogeneity  $h_{color}$  and the shape heterogeneity  $h_{shape}$  as follows:

$$4) \quad f = w \cdot h_{color} + (1 - w) \cdot h_{shape}$$

where  $w$  is the user defined weight for color (against shape) with  $0 \leq w \leq 1$ .

The standard deviations themselves are weighted by the object sizes :

$$5) \quad h_{color} = \sum_c w_c \left( n_{Merge} \cdot \sigma_c^{Merge} - \left( n_{Obj1} \cdot \sigma_c^{Obj1} + n_{Obj2} \cdot \sigma_c^{Obj2} \right) \right)$$

The shape criterion again consists of two subcriteria for smoothness and compactness. The calculation of these criteria is explained above. Based on the user defined weights,

$$6) h_{shape} = w_{cmpct} \cdot h_{cmpct} + (1 - w_{cmpct}) \cdot h_{smooth}$$

is calculated as being the user defined weight for the compactness criterion.

Again, the change in shape heterogeneity caused by the merge is evaluated by calculating the difference between the situation after and before the merge. This results in the following methods of computation for smoothness and compactness:

$$7) h_{smooth} = n_{Merge} \cdot \frac{l_{Merge}}{b_{Merge}} - \left( n_{Obj1} \cdot \frac{l_{Obj1}}{b_{Obj1}} + n_{Obj2} \cdot \frac{l_{Obj2}}{b_{Obj2}} \right) \text{ and}$$

$$8) h_{cmpct} = n_{Merge} \cdot \frac{l_{Merge}}{\sqrt{n_{Merge}}} - \left( n_{Obj1} \cdot \frac{l_{Obj1}}{\sqrt{n_{Obj1}}} + n_{Obj2} \cdot \frac{l_{Obj2}}{\sqrt{n_{Obj2}}} \right)$$

with  $n$  being the object size,  $l$  the object perimeter and  $b$  the perimeter of the bounding box.

Throughout the segmentation procedure, the whole image is segmented and image objects are generated based upon several adjustable criteria of homogeneity or heterogeneity in color and shape. Adjusting the so-called scale parameter indirectly influences the average object size: a larger value leads to bigger objects and vice versa. Additionally the influence of shape as well as the image's channels on the object's homogeneity can be adjusted. During the segmentation process all generated image objects are linked to each other automatically.

Find a detailed description in Baatz & Schäpe, 2000.

## Results

Fig. 4 shows how multiresolution segmentation, for instance, works on VHR data, an example for textured data. Image regions of different texture are separated, even when they are of similar spectral mean value (see, e.g., trees and meadows). Multiresolution segmentation, in particular, extracts regions of local contrast. The algorithm uses a description of heterogeneity weighted by the size of image objects. Small areas, which deviate in tone from their surrounding, are therefore merged with the surrounding. This is often the case in earth observation data and enables eCognition in principle to work even on rather noisy data. However, if these smaller areas are of interest, multi-resolution segmentation could be applied with a smaller scale parameter, extracting in principal image objects of smaller average size. The typical result of a segmentation run is: bigger homogeneous image objects (e.g., meadow), smaller heterogeneous image objects (e.g., trees) and smaller homogeneous image objects embedded in a high contrast. In addition, multiresolution segmentation yields image objects of similar scale. For this reason, grassland in this image is represented by several homogeneous image objects. All those objects however can be assigned to the same land use class by the classification.



Fig. 4: multiresolution segmentation result of VHR data

The number of degrees of freedom to separate an image into a given number of segments is astronomical. Multiresolution segmentation is an unsupervised, constrained local optimization procedure minimizing the weighted heterogeneity of image objects. As described above, the distributing treatment sequence contains a certain historicity. This can result in slight differences between segmentation results, especially for borders between image objects of low contrast, such as the meadow segments in the picture; however, the principal quality of segmentation does not change. In most cases such similar image objects will be assigned to the same class in the classification process, i.e., the border between them

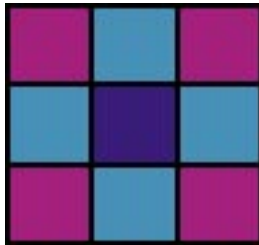
is not significant. Borders between regions of significant contrast will be represented in a reproducible way. If contrasts of interest are not considered, they can be extracted by choosing a smaller scale parameter in a reproducible way. The more homogeneous image data are, and the higher the portion of spectral homogeneity in comparison to

shape homogeneity in the homogeneity criterion, the higher the degree of reproducibility. eCognition initializes the random number generator for the distributing treatment sequence for each run with the same value, leading to 100 % reproducible segmentation results for the same subset of a scene.

Note that neither human digitizing nor segmentation methods optimizing a global features space result in completely reproducible results. With its local optimization procedure eCognition is especially able to extract local contrast, comparable to the human eye. The quality is furthermore independent of the subset of the scene, as there are no decisions made based on distributions in a global feature space, which of course changes from subset to subset of the same scene.

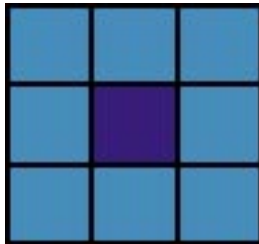
### Plane and diagonal neighborhood

eCognition distinguishes between two definitions of pixel neighborhood, see fig. 5:



Plane 4-neighborhood:

Pixels and objects are only defined as neighbors if they are connected at their plane borders.



Diagonal 8-neighborhood:

Pixels and objects are defined as neighbors if they are connected at a plane border or a corner point.

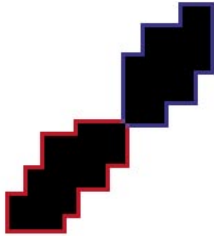
Fig. 5: plane and diagonal neighborhood

Note that the choice of diagonal or plane neighborhood in the first application of multiresolution segmentation in an eCognition project determines the definition of neighborhood for all other segmentation and classification procedures. In most cases the plane 4-neighborhood is the appropriate choice, which also works significantly faster than the diagonal 8-neighborhood.

Only in image data of relatively coarse resolution with structures of interest near to the pixel scale (e.g., streets in LANDSAT data), is the diagonal 8-neighborhood of advantage. Imagine an image object, e.g., a street, that is at one point connected by only the

corner points of two pixels. Segmentation with plane 4-neighborhood will lead to two objects, because the objects are not connected at their plane borders, see fig. 6. The diagonal 8-neighborhood approach recognizes the connection at the corner points and the segmentation results in a single object.

Plane 4-neighborhood  
(two objects)



Diagonal 8-neighborhood  
(one object)

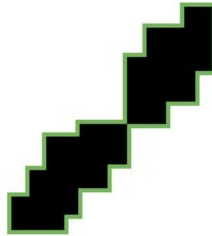


Fig. 6: a thin structure in plane and in diagonal neighborhood

The time to perform a segmentation is longer if you use the diagonal 8-neighborhood. In most cases, especially when using VHR data, even line objects cover an *area* and the plane 4-neighborhood is the adequate segmentation modulus. Use diagonal 8-neighborhood only in cases where structures of interest are close to the pixel scale, such as streets in LANDSAT data.

#### Internal composition of the heterogeneity criterion (fusion values)

To determine the outcome of the segmentation algorithm, the user is able to define several parameters, like the scale parameter, the amount of color and shape, and the single layer weights. This section describes how this user defined parametrisation is treated by the software.

As mentioned above, the scale parameter is a measure for the maximum change in heterogeneity that may occur when merging two image objects. Internally, this value is squared and serves as the threshold which terminates the segmentation algorithm. When a possible merge of image objects is examined, a fusion value between those two objects is calculated which can be compared directly to the squared scale parameter.

The fusion value mentioned above consists of two parts: A color criterion and a shape criterion. The color criterion is the change in heterogeneity that occurs when merging two image objects, as described by the change of the weighted standard deviation of the spectral values regarding their weightings. The shape criterion is a value that descri-



bes the improvement of the shape with regard to two different models describing ideal shapes.

The overall calculation of the fusion value  $f$  out of the color heterogeneity  $h_{color}$  and the shape heterogeneity  $h_{shape}$  is performed as follows:

$$f = w \cdot h_{color} + (1 - w) \cdot h_{shape}$$

where  $w$  is the user defined weight for color with  $0 \leq w \leq 1$ .

The color criterion  $h_{color}$  is the weighted mean of all changes in standard deviation for each channel  $c$ . The standard deviations  $\sigma_c$  themselves are weighted by the object sizes  $n_{Obj}$ :

$$h_{color} = \sum_c w_c \left( n_{Merge} \cdot \sigma_c^{Merge} - \left( n_{Obj1} \cdot \sigma_c^{Obj1} + n_{Obj2} \cdot \sigma_c^{Obj2} \right) \right)$$

The shape criterion  $h_{shape}$  consists of two parts: smoothness and compactness. The calculation of these criteria is explained above. For the shape criterion they have to be mixed using the user defined weights:

$$h_{shape} = w_{cmpct} \cdot h_{cmpct} + (1 - w_{cmpct}) \cdot h_{smooth}$$

with  $0 \leq w_{cmpct} \leq 1$  being the user defined weight for the compactness criterion. Again the change in shape heterogeneity caused by the merge is evaluated by calculating the difference between the situation after and before the merge. This results in the following methods of computation for smoothness and compactness:

$$h_{smooth} = n_{Merge} \cdot \frac{l_{Merge}}{b_{Merge}} - \left( n_{Obj1} \cdot \frac{l_{Obj1}}{b_{Obj1}} + n_{Obj2} \cdot \frac{l_{Obj2}}{b_{Obj2}} \right)$$

$$h_{cmpct} = n_{Merge} \cdot \frac{l_{Merge}}{\sqrt{n_{Merge}}} - \left( n_{Obj1} \cdot \frac{l_{Obj1}}{\sqrt{n_{Obj1}}} + n_{Obj2} \cdot \frac{l_{Obj2}}{\sqrt{n_{Obj2}}} \right)$$

with  $n$  being the object size,  $l$  the object perimeter and  $b$  the perimeter of the bounding box.

### Segmentation of sub-objects for the purpose of line analysis

eCognition provides special features for the object oriented line analysis of image objects. For this purpose you will find, in addition, that the “normal” segmentation mode is a special modus for sub-object extraction using the multiresolution segmentation procedure. It operates with exclusive use of the criterion for maximizing compactness. The scale parameter – here ranging from 0.5 to 1 – determines the maximum relative border length of sub-objects to neighbors which are not sub-objects of the same superior object.

For the analysis of image objects such as in fig. 7 the specific image object level can be subsegmented. The results are compact sub-objects which guarantee a minimum and maximum border length to the outer environment. Operating from center point to center point of these sub-objects means that it is possible to easily analyze the length of a curved line, average thickness, curvature etc.

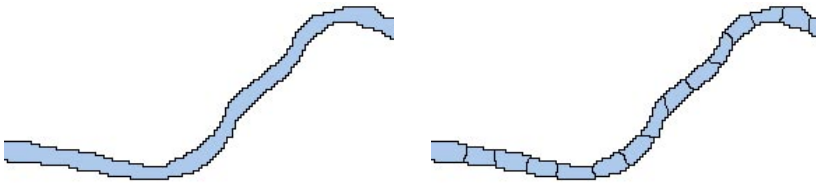


Fig. 7: linear structure subsegmented into compact objects

### The hierarchical network of image objects

The different techniques for segmentation in eCognition can be used to construct a hierarchical network of image objects which represents image information in different spatial resolutions simultaneously. The image objects are networked, so that each image object “knows” its context (neighborhood), its super-object and its sub-objects, see fig. 8. Thus, it is possible to define relations between objects, e.g., “Rel. Border to *Forest*,” and to utilize this kind of local context information.

Starting at the level of pixels the levels are consecutively numbered.

This hierarchical network is topologically definite, i.e., the border of a super-object is consistent with the borders of its sub-objects. The area represented by a specific image object is defined by the sum of its sub-objects’ areas. Technically this is carried into effect relatively simply, since all segmentation techniques used in eCognition are region merging algorithms. Each level is constructed based on its direct sub-objects, i.e., the

sub-objects are merged into larger image objects on the next level. Merging is limited by the borders of super-objects; adjacent image objects cannot be merged when they are sub-objects of different super-objects. In eCognition, image objects are defined as being spatially self-consistent.

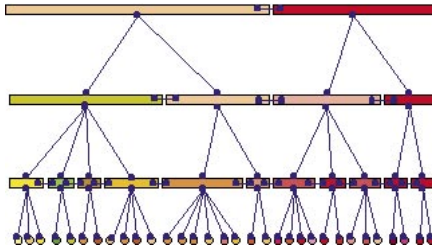


Fig. 8: hierarchical network of image objects in abstract illustration

The hierarchical network of image objects provides possibilities for innovative techniques:

- Structures of different scales can be represented simultaneously and thus classified in relation to each other.
- Different hierarchical levels can be segmented based on different data; an upper layer, for instance can be built based on thematic land register information, whereas a lower layer is segmented using remote sensing data. Classifying the upper level, each land register object can be analyzed based on the composition of its classified sub-objects. By means of this technique different data types can be analyzed in relation to each other.
- The shape of image objects can be corrected based on a regrouping of sub-objects.

**Note!** It is obvious that in this case the sequence in which the levels are segmented plays an important role. It makes a difference which level is constructed first. In this case, for instance, it would make sense to first build the cadastral level and then to create sub-objects. Again, adjacent sub-objects cannot be merged if they are not sub-objects of the same super-object.

A powerful technique is the analysis of image objects based on sub-objects. For this task you have the following possibilities:

- Texture analysis based on sub-objects, classifying attributes of all sub-objects of an image object on average. Attributes can for instance be contrast or shape.
- Line analysis based on sub-objects.

- Class-related features: relationships to classified sub-objects, such as the relative area of other image objects assigned to a certain class.

Another application of the hierarchical network of image objects is to classify image objects in relation to their respective super-object.

All segmentation procedures provided by eCognition operate on arbitrary levels in this hierarchical network. Since the level of pixels and the level of the whole image always exist by definition, each segmentation of a new level is a construction in between a lower and an upper level. To guarantee a definite hierarchy over the spatial shape of all objects the segmentation procedures follow two rules:

- Object borders must follow borders of objects on the next lower level.
- Segmentation is constrained by the border of the object on the next upper level.

## Handling of vector information

### Vectorization

eCognition supports a simultaneous raster / vector representation of image objects. After segmentation, vectorization functionality allows production of polygons for each image object. This vector information is produced in different resolutions for different purposes.

For rendering outlines of image objects independently of the zoom, eCognition produces polygons along the pixel raster (fig. 9: polygons 1) or slightly abstracted polygons (fig. 9: polygons 2). The latter polygons are referred to in the following as base polygons. They are created with respect to the topological structure of image objects and are used for exporting vector information, too. More abstracted vector information represents the shape of image objects independently of the topological structure (fig. 9: polygons 3) and is used for the computation of shape features. These polygons are referred to as shape polygons.

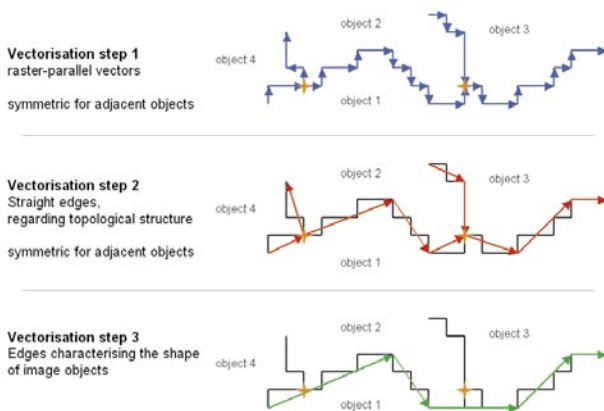


Fig.9: different polygon types for the vectorization of segments / image objects.

The computation of base polygons is done by means of a Douglas Peucker algorithm. The Douglas Peucker algorithm is one of the most common procedures for polygon extraction. It is a top-down approach, which starts with a given polygon line and divides it into smaller sections iteratively. Given the two end points of a polygon line – in eCognition typically these two starting points are topological points, see yellow marks in fig. 9 – the algorithm detects this specific point on the polygon line with the largest vertical distance to a line connecting the two end points, see fig. 10. At this detected point, the polygon line is cut into two shorter polygon lines, fig. 10 b. This procedure continues until the longest vertical distance is smaller than a given threshold, fig. 10 c. In other words: the threshold describes the strongest possible deviation of the polygon

from the underlying raster. In eCognition, this threshold can be defined in the dialog box “Create Polygons” and is measured in pixel units.

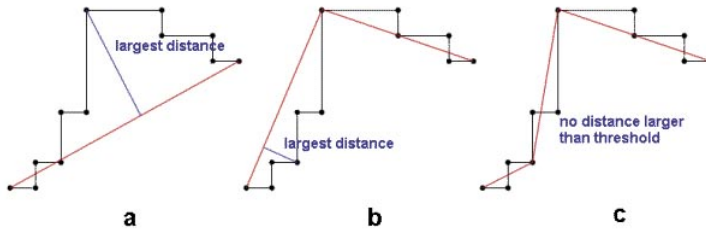


Fig. 10: Douglas-Peucker algorithm:

a) start configuration and detection of largest distance; b) new state after dividing into two sections; c) final result, no further division as no distance is larger than given threshold.

The Douglas-Peucker algorithm in its pure application suffers in some cases in producing relatively acute angles. Therefore, in eCognition, in order to improve the result, angles smaller than 45 degrees are detected in a second run. From the two particular vectors at such an angle, that one is subdivided which will result in the largest angles. This procedure continues in iterative steps until there are no angles smaller than 45 degrees.

For high thresholds, which produce a strong abstraction from the original raster, slivers and intersections within and between base polygons can arise. This can be especially disturbing when these base polygons are used for export. In order to avoid this effect, an additional, optional algorithm detects intersections and fractionises the affected vectors.

The shape polygons are created by means of a derivative of multiresolution segmentation (3), in this case not applied to image regions but to single vectors. In contrast to the Douglas-Peucker algorithm this procedure is a bottom-up approach. Starting with base polygons, the single vectors are subsequently merged, optimizing a homogeneity criterion. It is important to understand that the heterogeneity of single shape vectors is defined as deviation of the underlying base vectors. Thus, a threshold of 0 will always produce shape polygons identical with the underlying base polygons. The resulting shape therefore depends also on the threshold of the base polygons. A threshold bigger than 0 will result in a stronger abstraction than the base polygons.

Concretely, the deviation is computed as the maximum of the difference of length between shape vector and underlying base vectors and the sum of the lengths of the vertical parts of the underlying base vectors to the shape vector. Iteratively, the two adjacent vectors of a polygon which result in the smallest heterogeneity are merged. This continues until the predefined threshold is reached. As shown in figure 1, the resulting

shape polygons are independent of the topological structure and therefore specific for each single image object. A straight edge of a segment is represented as one vector, even if it contains a topological point (figure 1, yellow marks). Thus, fractally shaped parts of the boundary of an image object are represented in a characteristic way by a number of short vectors, whereas straight edges are represented by long edges. Based on these shape polygons a lot of different shape features can be computed.

It must be mentioned that vectorization cannot be applied to image objects which are built using plain neighborhood. Diagonal neighborhood can result in segments with mutual diagonal permeation. It is possible to represent such segments based on raster but not based on polygons.

### Skeletons

Based upon the objects' shape polygons, skeletons are created. Although the skeletons are only shown for individually selected objects they are created on demand for image objects for which polygons were created. By creating skeletons, the objects' shape can be described more accurate, since with skeletons it is possible to describe the inner structure of an object.

To obtain the skeletons, eCognition first performs a Delaunay triangulation of the objects' shape polygons. The skeletons then are created by identifying the mid points of the triangles and connecting them (see fig 11). Thereby the triangles' mid points are determined by connecting the midpoints of the triangles' sides.

To find skeleton branches, three types of triangles are created: branch-triangles (three-neighbor-triangle), connecting triangles (two-neighbor-triangles) and end-triangles (one-neighbor-triangles; see fig. 11). Branch triangles indicate branch-points of the skeleton, whereas two-neighbor-triangles indicate a connection point and end-triangles indicate end-points of the skeleton. To obtain the skeleton, the generated points become connected whereby the main line is represented by the longest possible connection of branch-points. Beginning with the main line, the connected lines then become ordered according to their types of connecting points (see fig. 11). The branch order is comparable to the streamorder of a river network. Respectively, each branch obtains an appropriate order value (see fig. 11): The main line always holds a value of 0 while the outmost branches have the highest values, depending on the objects' complexity (see fig. 11).

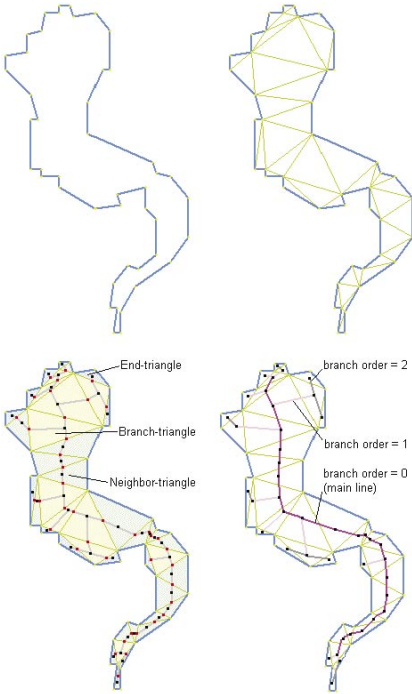


Fig. 11: Creation of skeletons based upon a Delaunay triangulation of image objects' shape polygons.

Beside a more accurate shape description, skeletons are also used for an automated object shape correction (automated object cut). Thereby, the degree of abstraction of the new created objects is determined by the branch order from where-on the object is cut. When cutting an object automatically, the cutline is always determined by the triangle structure:

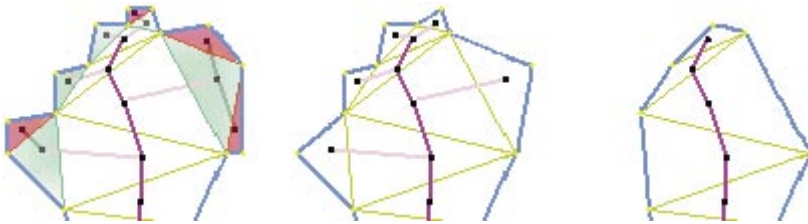


Fig. 12: Automatic object cut. Left: branch order 1 (red), branch order 2 (green). Middle: resulting object after cut by branch order greater-equal to 1. Right: resulting object after cut by branch order greater-equal to 0.

Regarding the results, automated object cut can be understood as pruning the object's skeleton from outside to inside.



### Vector format import and export

eCognition supports the import and export of thematic data in shape format. As eCognition is a region based analysis system, only polygons are considered for import and export, and not line or point information.

In order to make thematic information accessible for the raster based algorithms in eCognition, vector data need to be rasterized when imported. During the import routine for shape files, eCognition detects the resolution of the project that was defined when importing image raster data. Using this resolution, a thematic raster \*.tif file is produced with the same name as the original \*.shp file and the additional extension *\_r.\_id...* *r* stands for the resolution, and after *id* you will find a number which defines the chosen subset in a definite way. If you for instance import a shape file “thematic.shp” then you might receive a derived raster file named “thematic\_r15\_id312677.tif.” This file is saved on the hard disc in the same folder as the shape file. From now eCognition will handle the imported thematic information in the form of a thematic raster layer. The reloading of a saved project will always refer to the converted \*.tif file and not to the original vector data. Explicit information about resolution and lower left corner is necessary in order to avoid the converted \*.tif files’ being overwritten while using the same vector files in different projects with different resolutions or different subsets.

For the transition of a shape file into raster, eCognition uses a simple criterion in order to assign a pixel to the one or the other thematic class: it chooses the class with the maximum coverage of the pixel’s area. There is in principal no definite relation between vector and raster representations of shapes. For this reason, note the following particularities: It might happen that contiguous structures in the shape format are disjointed into several areas when converting them into raster format. This happens especially when structures represented by polygons are thinner than the size of a single pixel. When using the converted \*.tif files for segmentation in eCognition, the resulting image objects will be separated, too.

Due to the necessary raster conversion it might happen that the polygons in the exported shape files slightly differ from the polygons in imported vector data, even if you used exclusively the latter for segmentation. The exported polygons are produced based on first rasterizing the imported shape information and then transferring the resulting image objects again into vector format by means of eCognition’s vectorization procedure. Due to the subtle change of information in each of these transfers, a complete match cannot be guaranteed.

The export of image objects as polygons simply uses the base polygons produced throughout vectorization. Therefore, before exporting image objects in shape format, a vectorization must be performed using an appropriate threshold for abstraction. It is recommended to additionally apply the algorithm which avoids slivers.

## Fuzzy classification in eCognition

Fuzzy classification is a simple technique which basically translates feature values of arbitrary range into fuzzy values between 0 and 1, indicating the degree of membership to a specific class. Fuzzy classification was chosen for the analysis of image objects in eCognition because

- by translating feature values into fuzzy values, it standardizes features and allows the combination of features, even of very different range and dimension,
- it provides a transparent and adaptable feature description, especially compared to neural networks,
- it enables the formulation of complex feature descriptions by means of logical operations and hierarchical class descriptions.

Each class of a classification scheme formulated in eCognition contains a class description. Each class description consists of a set of fuzzy expressions allowing the evaluation of specific features and their logical operation. The output of the system is twofold: a fuzzy classification with detailed information of class mixture and reliability of class assignment, and a final crisp classification where each object is assigned to exactly one class (or none, if no assignment was possible).

A fuzzy rule can have one single condition or can consist of a combination of several conditions which have to be fulfilled for an object to be assigned to a class. In eCognition the conditions are defined by expressions which are inserted into the class descriptions. Expressions can be membership functions, similarities to classes, or a nearest neighbor. In the following the definition of single conditions and the logical combinations of several conditions are described.

### Single conditions

A single condition is defined by a one-dimensional membership function. For image object features, one-dimensional membership functions are defined by a graphical interface (see Functional Guide > Classification Basics). At this point, you can integrate all available knowledge about the relations between features and class assignment. The flexible fuzzy approach allows integration of knowledge close to human thinking. (see above [Concepts & Methods > Fuzzy classification systems](#)).

The simplest fuzzy rule you can create with eCognition, is to base your class assignment on only one condition, one single fuzzy feature. Take the following example:

The class *water* is defined by a low layer mean.

First, the membership function for the object feature “layer mean” has to be defined for the fuzzy set “low layer mean (LLM)”. Then you formulate a fuzzy rule representing your knowledge about the relation between the values of layer mean and the class assignment:

if layer mean (*object*)  $\in$  LLM, then land cover (*object*) = water

The first part in the rule represents the condition for the second part.

In eCognition this rule is realized very straightforwardly. Within the class hierarchy the class *water* is inserted and the object feature “layer mean” is selected. For this object feature the membership for fuzzy set *LLM* is defined using the graphical interface. Here, the fuzzy set *LLM* automatically corresponds to the land cover class *water*.

In the case of only one condition the image object’s membership to the class water equals the membership of the feature “layer mean” to the fuzzy set *LLM*.

$$\mu_{\text{water}}(\text{object}) = \mu_{\text{LLM}}(\text{layer mean}(\text{object}))$$

### Combinations of conditions

Usually the rules necessary to describe, for example, a land use class are more complicated than described above. Classes often consist of combinations of conditions connected by operators like “and,” “or” and “not.” The class *river* could for example be described as all water objects which have a high length/width ratio. The fuzzy set “high length/width ratio” *HL/W* is defined by a membership function for the object feature “length/width ratio”.

Thus, the above example could be extended by

if (layer mean(*object*)  $\in$  LLM) AND (length/width(*object*)  $\in$  HL/W), then land cover(*object*) = river

If the minimum operator is chosen, the image object’s membership to the class *river* equals the minimum value of the two individually calculated memberships:

$$\mu_{\text{river}}(\text{object}) = \min(\mu_{\text{LLM}}(\text{layer mean}(\text{object})), \mu_{\text{HL/W}}(\text{length/width}(\text{object})))$$

The output of these fuzzy rules can be the input to the next fuzzy rule. Thus a well-structured hierarchy is created. Even the simple example with a twofold condition

given above could be structured hierarchically. One rule defines “water” the second one specifies the subset “river” within the class *water*.

IF the first rule is fulfilled, THEN the second one is carried out.

- 1  $\mu_{\text{water}}(\text{object}) = \mu_{LLM}(\text{layer mean}(\text{object}))$
- 2  $\mu_{\text{river}}(\text{object}) = \min(\mu_{\text{water}}(\text{land cover}(\text{object})), \mu_{HL/W}(\text{length / width}(\text{object})))$

This hierarchy allows the easy definition of an additional class, *lake*, characterized by low length/width ratio (a fuzzy set *LL/W* is defined) A second rule is inserted in the lower hierarchy level.

The structure of the rule base shows very clearly that the classes *river* and *lake* are similar. They differ only in one condition. Using hierarchy makes this relation obvious and simplifies definitions. Both contribute to a consistent overall rule base.

- 1  $\mu_{\text{water}}(\text{object}) = \mu_{LLM}(\text{layer mean}(\text{object}))$
- 2.1  $\mu_{\text{river}}(\text{object}) = \min(\mu_{\text{water}}(\text{land cover}(\text{object})), \mu_{HL/W}(\text{length / width}(\text{object})))$
- 2.2  $\mu_{\text{lake}}(\text{object}) = \min(\mu_{\text{water}}(\text{land cover}(\text{object})), \mu_{LL/W}(\text{length / width}(\text{object})))$

If the fuzzy set *LL/W* is complementary to *HL/W*, this rule base would result in a classification of all water image objects either to land cover *river* or land cover *lake*. Rather than defining the membership for fuzzy set *LL/W* independently, the inversed similarity operator should be used to ensure a consistent rule base.

- 1  $\mu_{\text{water}}(\text{object}) = \mu_{LLM}(\text{layer mean}(\text{object}))$
- 2.1  $\mu_{\text{river}}(\text{object}) = \min(\mu_{\text{water}}(\text{land cover}(\text{object})), \mu_{HL/W}(\text{length / width}(\text{object})))$
- 2.2  $\mu_{\text{lake}}(\text{object}) = \min(\mu_{\text{water}}(\text{land cover}(\text{object})), \mu_{LL/W}(\text{length / width}(\text{object})))$

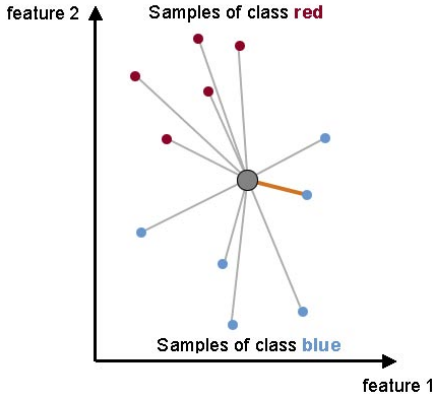
### Creation of conditions in multidimensional feature space by nearest neighbor classification

Using a combination of one-dimensional membership functions as described above, one can cover a multidimensional feature space. However, the form of the resulting multidimensional membership function is restricted. This restriction can lead to an insufficient class description and make classification impossible.

Thus, a direct creation of conditions in the multidimensional feature space is necessary. eCognition provides a fuzzy realization of the nearest neighbor classification to do this. The nearest neighbor (see [Functional guide > Classification basics](#)) is applied to selected object features and is trained by sample image objects. The fuzzy realization of the nearest neighbor approach – which is used in eCognition – automatically generates

multidimensional membership functions. They are suitable for covering relations in high-dimensional feature space. In comparison to pixel-based training, the object-based approach of the nearest neighbor requires fewer training samples: one sample object already covers many typical pixel samples and their variations.

The nearest neighbor classifies image objects in a given feature space and with given samples for the classes of concern.



The principle is simple: first, the software needs samples, typical representatives for each class. As eCognition is based on an object oriented approach to image analysis, specific image objects are addressed to be the samples. After a representative set of sample objects has been declared for each class, the algorithm searches for the closest sample object in the feature space for each image object. If an image object's closest sample object belongs to *Class A*, the object will be assigned to *Class A*.

Fig. 11: principle of nearest neighbor classification

However, all class assignments in eCognition are determined by assignment values in the range 0 (no assignment) to 1 (full assignment). The closer an image object is located in the feature space to a sample of class *A*, the higher the membership degree to this class. eCognition computes the distance as follows:

$$d = \sqrt{\sum_f \left( \frac{v_f^{(s)} - v_f^{(o)}}{\sigma_f} \right)^2}$$

$d$  Distance between sample object  $s$  and image object  $o$ .

$v_f^{(s)}$  Feature value of sample object for feature  $f$ .

$v_f^{(o)}$  Feature value of image object for feature  $f$ .

$\sigma_f$  Standard deviation of the feature values for feature  $f$ .

The distance in the feature space between a sample object and the image object to be classified is standardized by the standard deviation of all feature values. Thus, features of varying range can be combined in the feature space for classification. Due to the

standardization, a distance value of  $d = 1$  means that the distance equals the standard deviation of all feature values of the features defining the feature space.

Based on the distance  $d$  a multidimensional, exponential membership function  $z(d)$  is computed.

$$z(d) = e^{-k \cdot d^2}$$

The parameter  $k$  determines the decrease of  $z(d)$ . You can define this parameter with the variable function slope, whereby

$$k = \ln\left(\frac{1}{\text{function slope}}\right)$$

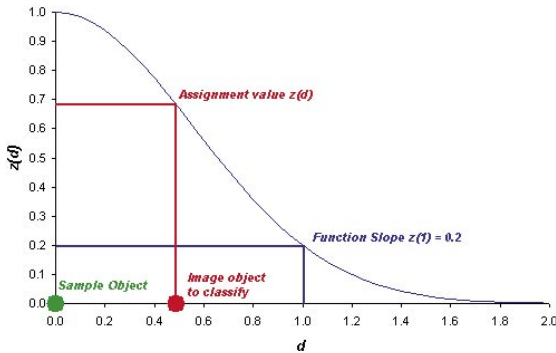
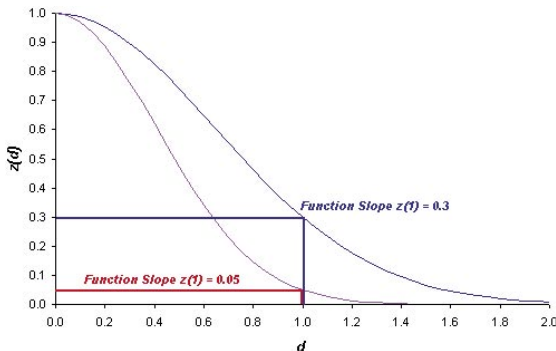


Fig. 12: nearest neighbor membership function; starting from the sample objects point in the features space, a membership function is computed depending on the function slope. Depending on the different distances between image object and sample object, different membership values result.

The function slope equals  $z(d)$  for  $d=1$ , see fig. 12. Thus, the function slope is the membership value of an image object to a class, if the closest sample object of that class has a distance to the image object which equals the standard deviation of the feature



values from the closest sample object. The default value for the function slope is 0.2. Fig. 13 demonstrates how the exponential function changes with different function slopes.

Fig. 13: nearest neighbor classification; different membership values for the same object for different function slopes.

The smaller the parameter function slope, the narrower the membership function. Image objects have to be closer to sample objects in the feature space to be classified. If the membership value is less than the minimum membership value (default setting 0.1), then the image object is not classified.

#### Differences between the application of membership functions and nearest neighbor

The examples below illustrate the differences and characteristics for class descriptions by combining one-dimensional membership functions and nearest neighbor:

#### Membership Functions

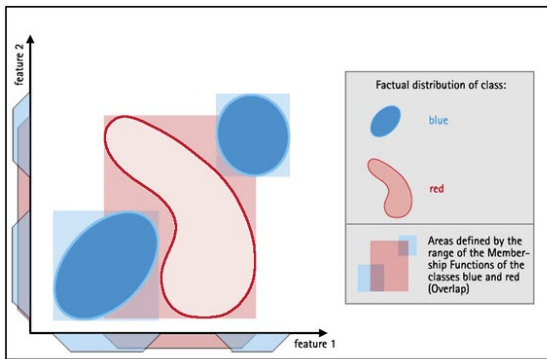




Fig. 14: overlap of class descriptions based on membership functions in a two-dimensional feature space

Interactive editing of membership functions – as it is possible in the one-dimensional case in eCognition – allows the formulation of knowledge and concepts. Rule base development to form multidimensional dependencies is very clear and an adaptation is easily possible. However, if one combines one-dimensional membership functions, the form of the resulting multidimensional membership functions is restricted and often does not match the necessary class description closely enough.

Therefore, if a class can be separated from other classes by just a few features or only one feature, we recommend the application of membership functions, otherwise the nearest neighbor is suggested.

The factual distribution of classes *blue* and *red* do not overlap. However, notice that the membership functions describing the class *blue* (blue trapezoids ) in fig. 14, feature 1) create an overlap with the membership function describing the class *red* (red trapezoid). The area of these overlaps in the feature space increases with its number of dimensions. When using two features to describe the classes *red* and *blue*, the areas defined by the range of the membership functions (red and blue rectangles) produce an enormous overlap. This overlap can be reduced if multidimensional mem- 

bership functions are directly designed, because the actual distribution of the class *red* can be approximated much better.

The use of a nearest neighbor (NN) to define these multidimensional membership functions is therefore advisable if you intend to use several object features for a class description. There are several reasons for the use of NN:

NN evaluates the correlation between object features favorably.

Overlaps in the feature space increase with its dimension and can be handled much easier with NN.

NN allows very fast and easy handling of the class hierarchy for the classification (without class-related features).

However, the feature space should also be kept as small as possible for a nearest neighbor classification, if you want to keep a reasonable number of sample objects for each class.

The two illustrations below show the procedure of a NN classification schematically. In a first step a few sample objects (samples) for each class are declared, see fig. 15.

### Nearest Neighbor I

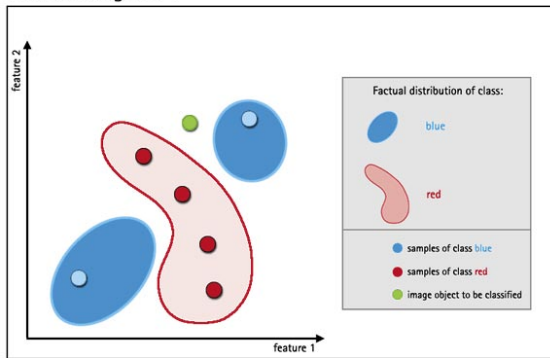


Fig. 15: initial training of nearest neighbor classification for the same class distributions as in fig. 14.

A first classification can be started afterwards. The algorithm searches the closest sample in the feature space for each image object. If an unclassified image object (●) is situated in the feature space next to a sample of class *blue*, it

will be assigned to this class, for instance.

Image objects that have not been assigned to any class and incorrectly classified objects can be chosen as additional samples to improve the classification result.



As you can see below, even objects situated on the edge of a class in an elongation can be integrated in the classification. Furthermore, separate classes, like the class *blue* in this example, can be classified straightforwardly.

### Nearest Neighbor II

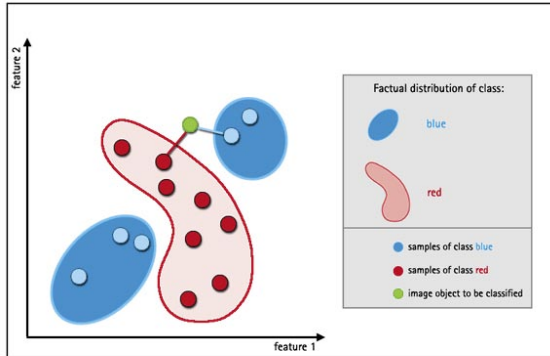


Fig. 16: refinement of nearest neighbor classification for the same class distributions as in fig. 14 and fig. 15.

### Combine conditions created with different approaches in a fuzzy rules base

Independent from the approach to define a condition and independent from the complexity of the condition, the fuzzy system will deliver one single value as the degree of fulfillment of the condition. This degree determines the result of the rule. Thus, it is clear that one can deliberately combine conditions in a rule base, independent of whether they are defined by a combination of one-dimensional membership functions or by nearest neighbor. This gives you great flexibility: For each condition and class description, the best fitting method can be chosen.

### The class hierarchy

The class hierarchy is the frame of eCognition's "language" for formulating the knowledge base for classifying image objects. It contains all classes of a classification scheme in a hierarchically structured form. The relations defined by the class hierarchy are twofold: the inheritance of class descriptions of child classes on the one hand, and semantic grouping of classes on the other. Each class is represented by a semantic group. The semantic objects can have different relationships to each other. Set aside is a third concept of a collection of classes into structure groups for the purpose of classification-based segmentation.

**Inheritance:** Class descriptions defined in parent classes are passed down to their child classes. A class can inherit descriptions from more than one parent class. Based on the same inherited feature descriptions, the inheritance hierarchy is a hierarchy of similarities.

**Purpose:** reduction of redundancy and complexity in the class descriptions.

**Groups:** Combination of classes to a class of superior semantic meaning. Beyond that, the groups hierarchy has a direct functional implication: each feature which addresses a class is automatically directed to this class and all its child classes in the groups hierarchy. A class can be part of more than one group. The group register displays the hierarchy of semantic meaning.

**Purpose:** combination of classes previously separated by the classification in a common semantic meaning.

**Structure:** Differs slightly from the other two hierarchies, although the structure can have parallels to the groups hierarchy. Different classes can be put together in structure groups as a basis for classification-based segmentation.

**Purpose:** fusion even of previously heterogeneous regions to single objects.

The inheritance hierarchy and the groups hierarchy essentially complement each other: while the inheritance hierarchy is used to subsequently separate and differentiate classes in the feature space, the groups hierarchy permits the meaningful grouping of the resulting classes. This structure of the class hierarchy is the basic reason for the abundance of semantic expression modes in eCognition.

### Advantages

- Formulation of complex semantics
- Structured semantics
- Reduction of complexity using class-related features
- Reduction of complexity by inheritance of features

### Inheritance

Inheritance is a common technique in object orientated modeling. More general parent objects pass on their properties to child objects. Inheritance is not only used for the sake of simplicity, but also ensures the synchronization of all child objects. Changes in a parent class do not need to be redone in each of the child classes since these inherit the changes automatically.

**Example:** The parent class *Forest* is described by spectral mean values. It passes on its class description to the child classes *Urban Forest* and *Free Forest*. *Urban Forest* and *Free Forest* now contain the same inherited class description, but at the same time they differentiate this description depending on the embedding in the *Urban* context, using for instance the feature “Relative border length to urban neighbor objects.”

If this class hierarchy is applied to another scene with a slightly different spectral signature, the class description of *Forest* can be adapted. All its child classes do not need to be adapted because they automatically inherit the modified class description. Multiple inheritance is supported. Circular inheritance, however, is impossible (*A* passes on to *B*, *B* passes on to *C*, *C* passes on to *A*).

### Groups

The groups hierarchy allows the grouping of even very different classes to a superior class of common semantic meaning. This functionality also facilitates the addressing of different classes together within a single class-related feature.

**Example:** The classes *Urban Forest*, *Urban Impervious* and *Urban Green* are child classes of the different classes *Forest*, *Impervious* and *Grasslands* in the inheritance hierarchy and inherit a part of their class descriptions from these classes.

However, in the groups hierarchy they are all child classes of the same parent class *Urban*, making *Urban Forest*, *Urban Impervious* and *Urban Green* part of a class of the same superior semantic meaning. Thus, the evaluation of the expression “relative area of *Urban* in the neighborhood” would refer to all classes with the parent class *Urban*, addressing *Urban Forest*, *Urban Impervious* and *Urban Green* at the same time. Classes can be child class of an arbitrary number of parent classes. Circular grouping, however, is impossible (*A* is child to *B*, *B* is child to *C*, *C* is child to *A*).

## The classification process

Classification is the process of connecting the classes in a class hierarchy with the image objects in a scene. After the process of classification, each image object is assigned to a certain (or no) class and thus connected with the class hierarchy. With the assignment of a class to an image object, the relations to other classes formulated in the specific class description are transferred to the image object. The result of the classification is a network of classified image objects with concrete attributes, concrete relations to each other and concrete relations to the classes in the class hierarchy. eCognition provides a dialog box for image object Information which provides comprehensive and detailed information about an object, e.g., attributes in all available features or detailed evaluation of each class for this object.

### Classification without class-related features

For classification without class-related features this process is deterministic and relatively simple. Each possible class is applied to each image object and the degree of fuzzy membership of the image object to the specific class is computed from the class description. The class with the highest membership will be assigned as the current classification to the image object, as long the membership value exceeds a predefined minimum value (for instance 10 %). Note that fuzzy membership values are not classification probabilities!

### Classification with class-related features

The use of class-related features is more complex. When an object changes its classification because of the classification of networked objects, the problem arises that the object itself might be a context feature for the evaluation of other objects. Therefore, classification must be an iterative process in cycles in which each object is classified over and over taking into account the changes in the classification of networked objects. The number of cycles can be specified for this purpose.

With context classification a new complexity arises: while classification without context is a deterministic process, context classification can become indeterministic and even unstable due to the possibility of circular dependencies between different classes. Classification becomes an optimizing problem in which convergence to a global best classification must be ensured.

This problem of unstable classification can basically be avoided by the sensible generation of class descriptions. Mutual or circular dependencies between classes should be avoided whenever possible. *Class A* should not be described by means of class-related

features which refer to *Class B* if *Class B* itself depends on *Class A* because of its class description. If this can be ensured classification might need more than one classification cycle, but it is not an optimizing problem.

### Simulated annealing

In certain, rather exceptional cases a class hierarchy which contains mutual or circular dependencies needs to be developed. For these cases, eCognition offers a specific optimization procedure for the classification process to handle potential instabilities: simulated annealing. It is developed to reduce and avoid the occurrence of local, mutual confirmation of wrong classifications.

The simulated annealing approach randomly changes the computed membership of an image object to a class, taking into account the membership values of the different classes. The extent to which random change takes place is determined by a so-called “temperature.” The higher the temperature chosen, the more decisions are done stochastically. The more the system is cooled down, the more deterministically the membership of each image object is computed. The whole simulated annealing classification starts with a cycle of chosen maximum temperature and cools down the temperature to zero over the number of chosen cycles, ending with totally deterministic classification decisions.

A deterministic classification is the standard case in eCognition: an image object is assigned to the class whose class description computes the highest membership value concerning the evaluation of the image object.

A stochastic decision allows an image object to be assigned to a class other than the one with the highest membership. The probability of this happening depends on the level of temperature. The specific class is determined by making a random choice based on the membership values of the image object to different classes. The higher the membership value, the higher the probability of assignment.

Thus, simulated annealing allows image objects to be temporarily classified in a sub-optimal way. Consequently, this can change the classification of networked image objects, which in turn can cause a change of the previously suboptimal classification to – in the best case – an optimal one.

Caution should be taken if the membership of an image object is identical for different classes. The assignment to one of these classes is then more or less a random choice. It might change for example, if you load a project and perform a new classification with settings identical to the current one. It might lead to a different result. Thus, membership of different classes with identical membership values are sometimes a cause for nonlinear effects in classification.

## Feature overview

eCognition provides a number of features which can be used by means of fuzzy logic to build class descriptions. This section gives an overview of these features.

### Object features

Object features are obtained by evaluating image objects themselves as well as their embedding in the image object hierarchy.

<b>Layer values</b>	These are features concerning the pixel channel values of an image object (spectral features).
<b>Shape</b>	With these features the shape of an image object can be described using the object itself or its sub-objects.
<b>Texture</b>	Texture features evaluate the texture of an image object either based on its sub-objects or on the gray level co-occurrence matrix (GLCM) resp. the grey level difference vector (GLDV) of the object's pixels after Haralick
<b>Hierarchy</b>	These features provide information about the embedding of an image object in the entire image object hierarchy.
<b>Thematic attributes</b>	These are attributes of the thematic layer objects. This feature is only available if such a thematic layer has been imported into the project.

### Class-related features

Class-related features refer to the classification of other image objects which are taken into account for the classification of the image object in question.

<b>Relations to neighbor objects</b>	These features refer to existing class assignments of image objects on the same level in the image object hierarchy
<b>Relations to sub-objects</b>	These features refer to existing class assignments of image objects on a lower level in the image object hierarchy

<b>Relations to super-objects</b>	These features refer to existing class assignments of image objects on a higher level in the image object hierarchy
<b>Membership to</b>	In some cases it is important to incorporate the membership value to different classes into one class. This function allows the explicit addressing of membership values to different classes
<b>Classified as</b>	The idea of this feature is to enable the user to refer to the classification of an object without regard to the membership value.
<b>Classification value of</b>	In some cases it is of importance to incorporate the membership value to different classes in one class. This function allows you to explicit address the membership values to different classes. As opposed to the feature “Membership to” it is possible to use all membership values to all classes without restrictions instead of the use of only the three highest values.

### Global Features

Global features refer to global project parameters. Since they are not usable to differentiate objects by class, they are actually only useful in combination with customized features (see Guided Tours > Defining arithmetic features and Guided Tour > Defining relational features).

<b>Global scene related</b>	These features refer to global statistical parameters of the scene.
<b>Global Class related</b>	These features refer to global statistical parameters for all objects of a class or a group.

### Terms

This section provides further components for building a class description.

<b>Standard nearest neighbor</b>	Applies the standard nearest neighbor to the class description.
<b>Nearest neighbor</b>	Applies the nearest neighbor to the class description.

**Similarity to classes** Use this feature to define a class description that is identical to another.

**Logical terms** Apply logical operators to the class description to define how class assignments to classes described by more than one feature have to be evaluated.

### Sub-object analysis

The hierarchical structure of image objects offers a new technique of form and texture analysis: sub-object analysis. By looking at the structure of a given image object's sub-objects, we can gain insight into the object's form and texture:

Consider a line-shaped object containing sufficiently small sub-objects. By taking the centers of each sub-object and adding the distance between the centers of neighboring sub-objects, you can get a good measure of the line length. Obtain a measurement of the object's twisting by looking at the angles between the lines connecting the neighboring sub-object centers. Multiresolution segmentation offers a special purpose algorithm for creating suitable sub-objects for this type of line analysis.

Texture features can be derived from the features of sub-objects in a similar manner, e.g., mean size of the sub-objects or the standard deviation of the sub-objects' sizes. An advanced method of sub-object texture analysis is to define several texture classes (e.g., *black dot*, *white line*) and classify the sub-objects using these classes. Afterwards, knowledge-based relations to sub-objects can be used to analyze the texture of the image object by evaluating the classification of the sub-objects (e.g., at least 5 % of the object's area consists of sub-objects of type *black dots* and not more than 20 % of *white lines*).

### Frequently used concepts

**Relative border length to neighbor objects** The length of the image object's common border with the selected neighboring objects divided by the total border length of the image object.

**Relative area of a group of objects' subgroup** The area of the subgroup is divided by the total area of the entire object group. This concept is similar to relative border but can also be applied to sub-objects and other sets of objects. The relative area computes the amount of a specific image object type in a larger set of image objects.

**Neighbor objects** All image objects adjacent to an image object on the same level.



<b>Extended neighborhood</b>	Consists of all neighbor objects and additional image objects with a distance less than a predefined threshold. The distance between two objects is defined as the distance between the object centers.
<b>Sub-objects</b>	All image objects located in the image objects of choice on the next lower level.
<b>Super-objects</b>	The image object on the next coarser level in which the object of concern is located.

## Features and terms for the fuzzy class description

### Object features

#### Layer values

##### Mean

Layer mean value  $\bar{c}_L$  calculated from the layer values  $c_{Li}$  of all  $n$  pixels forming an image object.

$$\bar{c}_L = \frac{1}{n} \cdot \sum_{i=1}^n c_{Li}$$

Feature value range: [0 ; depending on the bit depth of data], for 8 bit data the value range is [0 ; 255].

##### Brightness

Sum of the mean values of the layers containing spectral information divided by their quantity computed for an image object (mean value of the spectral mean values of an image object). To define which layers provide spectral information, use the dialog “Define Brightness” (menu item “Classification > Advanced Settings > Image Layers for Brightness...”).

$$b = \frac{1}{n_L} \cdot \sum_{i=1}^{n_L} \bar{c}_i$$

Feature value range: [0 ; depending on bit depth of data], for 8 bit data the value range is [0 ; 255].

##### Max Diff.

To calculate “Max Diff.” The minimum mean value belonging to an object is subtracted from its maximum value. To get the maximum and minimum value the means of all channels resp. layers belonging to an object are compared with each other. Subsequently the result is divided by the brightness.

Feature Value range: [0; depending on bit depth of data]

##### StdDev

Standard deviation calculated from the layer values of all  $n$  pixels forming an image object.

$$\sigma_L = \sqrt{\frac{1}{n-1} \cdot \sum_{i=1}^n (c_{Li} - \bar{c}_L)^2}$$

Feature value range: [0 ; depending on bit depth of data]

**Ratio**

The ratio of layer  $L$  is the layer  $L$  mean value of an image object divided by the sum of all spectral layer mean values. Again, only layers containing spectral information can be used to achieve reasonable results.

$$r_L = \frac{\bar{c}_{L, Object}}{\bar{c}_{L, SO}}$$

Feature value range: [0 ; 1]

**Min. pixel value**

Value of the image object's pixel with the lowest value.

Feature value range: [0; depending on the bit depth of data].

**Max. pixel value**

Value of the image object's pixel with the highest value.

Feature value range: [0; depending on the bit depth of data].

**to Neighbors****Mean diff. to neighbors**

For each neighboring object the layer mean difference is computed and weighted with regard to the length of the border between the objects (if they are direct neighbors, feature distance = 0) or the area covered by the neighbor objects (if neighborhood is defined within a certain perimeter (in pixels) around the image object in question, feature distance > 0).

The mean difference to direct neighbors is calculated as follows:

$$\Delta c_L = \frac{1}{l} \cdot \sum_{i=1}^n l_{Si} \cdot (\bar{c}_L - \bar{c}_{Li})$$

$l$  border length of the image object of concern

$l_{Si}$  border length shared with direct neighbor  $i$

$\bar{c}_L$  layer mean value of the image object of concern

$\bar{c}_{Li}$  layer mean value of neighbor  $i$

$n$  quantity of neighbors

If you defined neighbor objects as objects within a certain perimeter (see “feature distance” below), the mean difference is computed differently:

$A$  total area covered by neighbor objects

$A_i$  area covered by neighbor object  $i$

$\bar{c}_L$  layer mean value of the image object of concern

$\bar{c}_{Li}$  layer mean value of neighbor  $i$

$n$  quantity of neighbors

Feature value range: [– depending on bit depth of data; depending on bit depth of data]

#### Mean diff. to neighbors (abs)

The same definition as for “Mean diff. to neighbors,” with the difference that absolute values of the differences are averaged:

$$\Delta c_L = \frac{1}{l} \cdot \sum_{i=1}^n l_{Si} \cdot |\bar{c}_L - \bar{c}_{Li}|$$

Feature value range: [0 ; depending on bit depth of data]  $\Delta c_L = \frac{1}{A} \cdot \sum_{i=1}^n A_i \cdot |\bar{c}_L - \bar{c}_{Li}|$

#### Mean diff. to brighter neighbors

This feature is computed the same way as “Mean diff. to neighbors,” but only image objects with a layer mean value larger than the layer mean value of the object concerned are regarded.

Feature value range: [0 ; depending on bit depth of data]

**Mean diff. to darker neighbors**

This feature is computed the same way as “Mean diff. to neighbors,” but only image objects with a layer mean value less than the layer mean value of the object concerned are regarded.

Feature value range: [0 ; depending on bit depth of data]

$$\Delta C_L = \bar{C}_{L, Object} - \bar{C}_{L, Scene}$$

**Rel. border to brighter neighbors**

Ratio of shared border with image objects of a higher mean value in the selected layer and the total border of the image object concerned.

Feature value range: [0 ; 1]

**to Super-object**

$$\Delta \sigma_L = \sigma_{L, object} - \sigma_{L, so}$$

**Mean diff. to super-object**

Difference between layer  $L$  mean value of an image object and the layer  $L$  mean value of its super-object. You can determine in which level the super-object is selected by editing the feature distance.

Feature value range: [0 ; depending on bit depth of data]

**StdDev diff. to super-objects**

Difference between layer  $L$  StdDev value of an image object and the layer  $L$  StdDev of its super-object. You can determine in which level the super-object is selected by editing the feature distance.

$$r_{\sigma} = \frac{\sigma_{L, object}}{\sigma_{L, so}}$$

Feature value range: [0; depending on bit depth of data]

**Ratio to super-object**

Ratio of the layer  $L$  mean value of an image object and the layer  $L$  mean value of its super-object. You can determine in which level the super-object is selected by editing the feature distance.

Feature value range: [0 ; inf]

**StdDev ratio to super-objects**

Ratio of the layer  $L$  standard deviation of an image object and the layer  $L$  standard deviation of its super-object. You can determine in which level the super-object is selected by editing the feature distance.

Feature value range: [0; inf]

**to Scene****Mean diff. to scene**

Difference between layer  $L$  mean value of an image object and the layer  $L$  mean value of the whole scene.

$$\Delta C_L = \bar{C}_{L, Object} - \bar{C}_{L, Scene}$$

Feature value range: [0 ; depending on bit depth of data]

**Ratio to scene**

Ratio to scene of layer  $L$  is the layer  $L$  mean value of an image object divided by the layer  $L$  mean value of the whole scene.

$$r_L = \frac{\bar{C}_{L, object}}{\bar{C}_{L, scene}}$$

Feature value range: [0; inf]

**Shape**

Many of the form features provided by eCognition are based on the statistics of the spatial distribution of the pixels that form an image object. As a central tool to work with these statistics eCognition uses the covariance matrix:

$$S = \begin{pmatrix} Var(X) & Cov(XY) \\ Cov(XY) & Var(Y) \end{pmatrix}$$

$X$  =  $x$ -coordinates of all pixels forming the image object

$Y$  =  $y$ -coordinates of all pixels forming the image object

Another frequently used technique to derive information about the form of image objects (especially length and width) is the bounding box approximation. Such a bounding box can be calculated for each image object and its geometry can be used as a first clue of the image object itself.

The main information provided by the bounding box is its length  $a$ , its width  $b$ , its area  $a * b$  and its degree of filling  $f$ , which is the area  $A$  covered by the image object divided by the total area  $a * b$  of the bounding box.

Methods using bounding box approximation are well suited for image objects that are not curved. For lengthy or curved image objects, methods using sub-objects are superior since sub-objects provide the possibility of iterating through the super-object along a center line. By adding up the distances between the single sub-objects along the center line, you can get a very good approximation of the image object length. For more compact image objects, however, it is difficult to determine such a center line, which is the reason why, for these objects, the bounding box approximation is superior.

### Generic shape features

#### Area

In nongeoreferenced data the area of a single pixel is 1. Consequently, the area of an image object is the number of pixels forming it. If the image data is georeferenced, the area of an image object is the true area covered by one pixel times the number of pixels forming the image object.

Feature value range: [0 ; scene size]

#### Length/width

There are two ways to compute the length/width ratio of an image object.

1. The ratio length/width is identical to the ratio of the eigenvalues of the covariance matrix with the larger eigenvalue being the numerator of the fraction.

$$\gamma = \frac{l}{w} = \frac{eig_1(S)}{eig_2(S)}, \quad eig_1(S) > eig_2(S)$$

2. The ratio length/width can also be approximated using the bounding box.

$$\gamma = \frac{l}{w} = \frac{a^2 + ((1-f) \cdot b)^2}{A}$$

eCognition uses both methods for the calculation and takes the smaller of both results as the feature value.

Feature value range: [0 ; 1]

### Length

$$l = \sqrt{A \cdot \gamma}$$

The length can also be computed using the length-to-width ratio derived from a bounding box approximation. It is approximated as follows:

Another possibility which works better for curved image objects is to calculate the length of an image object based on its sub-objects.

Feature value range: [0 ; depending on shape of image object]

### Width

Also the width of an image object is approximated using the length-to-width ratio. In eCognition the width is approximated as follows:

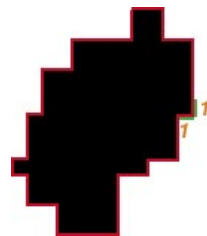
$$w = \sqrt{\frac{A}{\gamma}}$$

Again, for curved image objects the use of sub-objects for the calculation is the superior method.

Feature value range: [0 ; depending on shape of image object]

### Border length

The border length  $e$  of an image object is defined as the sum of edges of the image object that are shared with other image objects or are situated on the edge of the entire scene. In nongeoreferenced data the length of a pixel edge is 1.



Feature value range: [4 ; depending on shape of image object]



### Shape index

Mathematically the shape index is the border length  $e$  of the image object divided by four times the square root of its area  $A$ . Use the shape index  $s$  to describe the smoothness of the image object borders. The more fractal an image object appears, the higher its shape index.

$$s = \frac{e}{4 \cdot \sqrt{A}}$$

Feature value range: [1 ; depending on shape of image object]

### Density

The density  $d$  can be expressed by the area covered by the image object divided by its radius. eCognition uses the following implementation, where  $n$  is the number of pixels forming the image object and the radius is approximated using the covariance matrix:

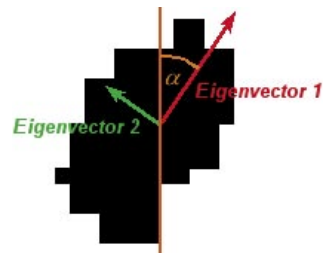
$$d = \frac{\sqrt{n}}{1 + \sqrt{\text{Var}(X) + \text{Var}(Y)}}$$

Use the density to describe the compactness of an image object. The ideal compact form on a pixel raster is the square. The more the form of an image object is like a square, the higher its density.

Feature value range: [0 ; depending on shape of image object]

### Main direction

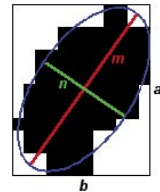
In eCognition, the main direction of an image object is the direction of the eigenvector belonging to the larger of the two eigenvalues derived from the covariance matrix of the spatial distribution of the image object.



Feature value range: [0 ; 180]

### Asymmetry

The lengthier an image object, the more asymmetric it is. For an image object, an ellipse is approximated which can be expressed by the ratio of the lengths of minor and major axes of this ellipse. The feature value increases with the asymmetry.



$$K = 1 - \frac{n}{m}$$

Feature value range: [0 ; 1]

### Compactness

In eCognition the compactness  $c$ , used as a feature, is calculated by the product of the length  $m$  and the width  $n$  of the corresponding Object and divided by the number of its inner pixels  $a$ .

$$c = \frac{n \cdot m}{a}$$

Feature value range: [0; inf]

### Elliptic Fit

As a first step in the calculation of the elliptic fit is the creation of an ellipse with the same area as the considered object. In the calculation of the ellipse also the proportion of the length to the width of the Object is regarded. After this step the area of the object outside the ellipse is compared with the area inside the ellipse that is not filled out with the object. While 0 means no fit, 1 stands for a complete fitting object.

Feature value range: [0; 1]

### Rectangular fit

A first step in the calculation of the rectangular fit is the creation of a rectangle with the same area as the considered object. In the calculation of the rectangle also the proportion of the length to the width of the object is regarded. After this step the area of the object outside the rectangle is compared with the area inside the rectangle, which is not filled out with the object. While 0 means no fit, 1 stands for a complete fitting object.

Feature value range: [0; 1]

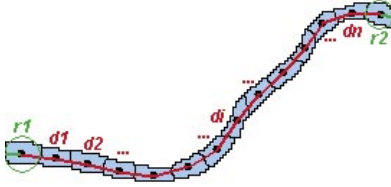
### Line features based on sub-objects

The information for classification of an object can also be derived from information provided by its sub-objects. A specific method is to produce compact sub-objects for the purpose of line analysis. You can find information about the creation of such sub-objects in the above sub-chapter [Concepts & Methods > Multiresolution Segmentation of Image Objects](#) and in [Functional Guide > Image Object Generation I: Multiresolution Segmentation](#). The basic idea is to represent the shape of an object by compact sub-objects and operate from center point to center point to get line information.

As mentioned above, this method is superior to bounding box approximation, if you want to extract features out of lengthy and curved image objects (e.g., image objects representing rivers or roads).

**Line so: length**

Of the image object of concern, the object center is known. Among all the sub-objects those two objects are detected which are situated furthest from this center point. From one end point to the other, the distances between the center points of adjacent sub-objects are added together (red lines). The radii of the end objects are also considered to complete the approximation (green).



$$l_{SO} = r_1 + r_2 + \sum_{i=1}^n d_i \quad \gamma_{SO} = \frac{l_{SO}}{w_{SO}} = \frac{l_{SO}^2}{A}$$

Feature value range: [1 ; depending on image object shape]

**Line so: width**

The image object width calculated on the basis of sub-objects is the area  $A$  (in pixels) of the image object divided by its length derived from sub-object analysis.

$$w_{SO} = \frac{A}{l_{SO}}$$

Feature value range: [1 ; depending on image object shape]

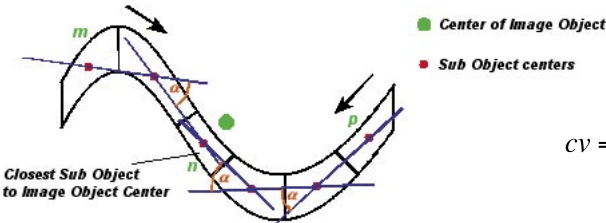
**Line so: length/width**

The length-to-width ratio based on sub-object analysis is the square length derived from sub-object analysis divided by the object area (in pixels).

Feature value range: [0 ; 1]

Line so: curvature/length

The curvature of an image object divided by its length. Both curvature and length are based on analysis of sub-objects. The curvature is the sum of all changes in direction (absolute values) when iterating through the sub-objects from both ends to the sub-object that is situated closest to the center of the image object of concern.



$$CV = \sum_{i=m}^n \alpha_i + \sum_{i=p}^n \alpha_i + \alpha_i$$

The curvature is calculated as follows:

Feature value range: [0 ; depending on image object shape]

Line so: stddev curvature

The standard deviation of all changes in direction ( $\alpha_i$ ) when iterating through the sub-objects from both ends to the sub-object situated closest to the center of the image object of concern. If an image object can be characterized by a high standard deviation of its curvature, this means that there are a large number of changes in direction when iterating through the sub-objects. On the other hand, an image object may appear curved, but if it follows a circular line, the standard deviation of its curvature will be small, since the changes in direction when iterating through its sub-objects are more or less constant.

Feature value range: [0 ; 180]

+

The polygon features provided by eCognition are based on the vectorization of the pixels that form an image object. The following figure shows a raster image object with its polygon object after vectorization:

The lines that are shown in red colors are the edges of the polygon object of the raster image object.



**Area (excluding inner polygons)**

Calculating the area of a polygon is based on Green's Theorem in a plane. Given points  $(x_i, y_i)$ ,  $i = 0, \dots, n$ , with  $x_0 = x_n$  and  $y_0 = y_n$ , the following formula can be used for rapidly calculating the area of a polygon in a plane:

$$Area = \frac{1}{2} \sum_{i=0}^{n-1} a_i \text{ where } a_i = x_i y_{i+1} - x_{i+1} y_i$$

This value does not include the areas of existing inner polygons.

**Area (including inner polygons)**

The same formula as for area (excluding inner polygon) is used to calculate this feature. The areas of the existing inner polygons in the selected polygon are taken into account for this feature value.

The above picture shows a polygon with one inner object.

**Perimeter**

The sum of the lengths of all edges which form the polygon is considered as the perimeter of the selected polygon.

**Compactness**

In eCognition compactness is defined as the ratio of the area of a polygon to the area of a circle with the same perimeter. The following formula is used to calculate the compactness of the selected polygon:

$$compactness = \frac{4 \cdot \pi \cdot Area}{Perimeter^2}$$

Feature value range: [0 ; 1 for a circle]

**Number of edges**

This feature value simply represents the number of edges which form the polygon.

### Stddev of length of edges

This feature value shows how the lengths of edges deviate from their mean value. The following formula for standard deviation is used to compute this value.

where  $X_i$ ,  $\bar{X}$  are length of edge  $i$  and mean value of all lengths respectively. The total number of edges has been shown as  $n$ .

$$stddev = \sqrt{\frac{\sum_{i=0}^n (X_i - \bar{X})^2}{n}}$$

### Average length of edges

The average length of all of edges in a polygon is another feature value which eCognition calculates using the following formula:

$$Average = \frac{\sum_{i=0}^n X_i}{n}$$

where  $X_i$  is length of edge  $i$  and  $n$  total number of edges.

### Length of longest edge

The value of this feature contains the length of the longest edge in the selected polygon.

### Number of inner objects

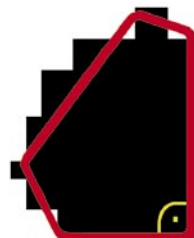
If the selected polygon includes some other polygons (image objects), the number of these objects is assigned to this feature value. The inner objects are completely surrounded by the outer polygon.

### Edges longer than...

This feature reports the number of edges that have lengths exceeding a threshold value. The user defines the threshold value.

### Rectangular angles with edges longer than...

This feature value gives the number of rectangular angles that have at least one side edge longer than a given user defined threshold. The following figure shows a polygon with one rectangular angle.



### Shape features based on skeletons

For the better understanding of the following descriptions the skeleton is divided in a main line and branches as above mentioned. Each mid-point of the triangles created by the Delaunay Triangulation is called a node. For more information see Concepts & Methods > Handling of Vector Information.

#### Length of main line (no cycles)

The length of the main line is calculated by the sum of all distances between its nodes. “No cycles” means that if an object contains an island polygon, the main line is calculated without regarding the island polygon. In this case the main line could cross the island polygon. Note that this is an internal calculation and could not be visualized like the skeletons regarding the island polygons.

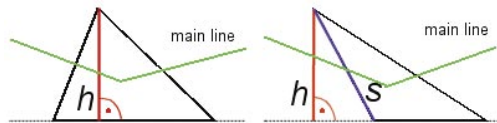
Feature value range: [0; depending on shape of objects]

#### Length of main line (regarding cycles)

The length of the main line is calculated by the sum of all distances between its nodes. “regarding cycles” means that if an object contains an island polygon, the main line is calculated regarding this island polygon. Consequently the main line describes a path around the island polygon. This way also the skeletons for visualization are calculated. Feature value range: [0; depending on shape of objects]

#### Width (only main line)

To calculate the width of the objects the average height  $h$  of all triangles crossed by the main line is calculated. An exception are triangles in which the height  $h$  does not cross one of the sides of the corresponding triangle. In this case the nearest side  $s$  is used to define the height.



Feature value range: [0; depending on shape of objects]

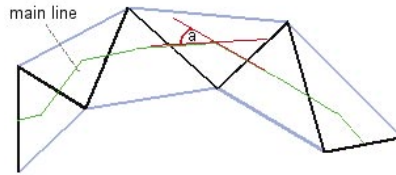
#### Length/width (only main line)

In the feature “Length/width (only main line)” the length of an object is divided by its width.

Feature value range: [0; depending on shape of objects]

### Curvature/length (only main line)

The feature “Curvature/length (only main line)” is calculated by the ratio of the curvature of the object and its length. The curvature is the sum of all changes in direction of the main line. Changes in direction are expressed by the acute angle  $\alpha$  in which sections of the main line, built by the connection between the nodes, cross each other.



Feature value range: [0; depending on shape of objects]

### Stddev curvature (only main line)

The standard deviation of the curvature is the result of the standard deviation of the changes in direction of the main line. Changes in direction are expressed by the acute angle in which sections of the mainline, built by the connection between the nodes, cross each other.

Feature value range: [0; depending on shape of the objects]

### Degree of skeleton branching

The degree of skeleton branching describes the highest order of branching in the corresponding object.

Feature value range: [0; depending on shape of objects]

### Number of segments

Number of segments is the number of all segments of the main line and the branches.

Feature value range: [0; depending on shape of objects]

### Avrg. area represented by nodes

“Avrg. area represented by nodes” calculates the average area of all triangles created by the Delaunay Triangulation.

Feature value range: [0; depending on shape of objects]



**Stddev of area represented by nodes**

“Stddev of area represented by nodes” calculates the standard deviation of all triangles created by the Delaunay Triangulation.

Feature value range: [0; depending on shape of the objects]

**Maximum branch length**

Maximum branch length calculates the length of the longest branch. The length of a branch is measured from the intersect point of the branch and the main line to the end of the branch.

feature value range: [0; depending on shape of objects]

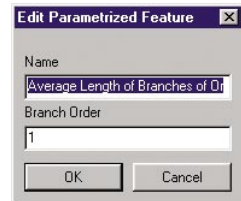
**Average branch length**

“Average branch length” calculates the average length of all branches of the corresponding object.

Feature value range: [0; depending on shape of objects]

**Average length of branches of order [1]**

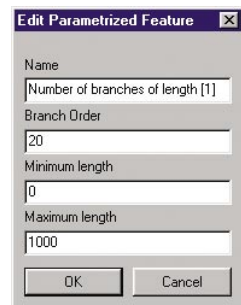
“Average length of branches of order” calculates the average length of branches of a selected order. The length of the branch of the selected order is measured from the intersect point of the whole branch and the main line to the end of the branch. The order can be manually defined. With a right click on the feature a pop up menu opens where you have to select “Edit Feature”. In the dialog it is possible to select the order of the branches to select. For more information regarding parametrized features see User Interface > Edit Parametrized Feature.



Feature value range: [0; depending on shape of objects]

**Number of branches of length [1]**

“Number of branches of length” calculates the number of branches of a special length up to a selected order. At this all ends of branches are counted up to the selected order. Since it is a parametrized feature it is possible to select the branch

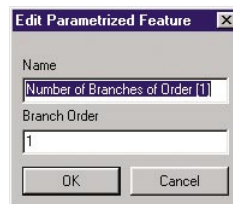


order and the length in a special range manually. To do so, select “Edit Feature” from the pop up menu you can open with a right click. For more information regarding parametrized features see User Interface > Edit Parametrized Features.

Feature value range [0; depending on shape of objects]

#### Number of branches of order [1]

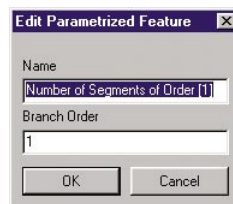
“Number of branches of order” calculates the number of branches of a predefined order. Define the branch order in the dialog “Edit Parametrized Features”. To open this dialog select “Edit Feature” from the pop up menu that is opened with a right click on the corresponding feature. For more information regarding parametrized features see User Interface > Edit Parametrized Features.



Feature value range [0; depending on shape of objects]

#### Number of segments of order [1]

“Number of segments of order” calculates the number of line segments of branches with a selected order. Note, that only segments are counted that do not belong to a lower order. Define the branch order in the dialog “Edit Parametrized Features”. To open this dialog select “Edit Features” from the pop up menu that is opened with a right click on the corresponding feature. For more information regarding parametrized features see User Interface > Edit Parametrized Features.



Feature value range [0; depending on shape of objects]

### Position

All the following features refer to the position of an image object relative to the entire scene. These features are of special interest when working with georeferenced data as image object can be described by their geographic position.

#### X-center

X-position of the image object center (center of gravity, mean value of all X-coordinates).

Feature value range: [0 ; number of columns of raster layers] or depending on coordinates.

**Y-center**

Y-position of the image object center (center of gravity, mean value of all Y-coordinates).

Feature value range: [0 ; number of rows of raster layers] or depending on coordinates

**X-min**

Minimum X-position of the image object (derived from bounding box).

Feature value range: [0 ; number of columns of raster layers] or depending on coordinates

**Y-min**

Minimum Y-position of the image object (derived from bounding box).

Feature value range: [0 ; number of rows of raster layers] or depending on coordinates

**X-max**

Maximum X-position of the image object (derived from bounding box).

Feature value range: [0 ; number of columns of raster layers] or depending on coordinates

**Y-max**

Maximum Y-position of the image object (derived from bounding box).

Feature value range: [0 ; number of rows of raster layers] or depending on coordinates

**distance to image border**

Distance to the nearest border of the image.

Feature value range: [0; half of the number of columns or rows (depending on the higher value) of raster layers or depending on coordinates]

**x distance to image right border**

Distance to the right border of the image.

Feature value range: [0; number of columns of raster layers or depending on coordinates]

**y distance to image top border**

Distance to the top border of the image.

Feature value range: [0; number of rows of raster layers or depending on coordinates]

**x distance to image right border**

Distance to the right border of the image.

Feature value range: [0; number of columns of raster layers or depending on coordinates]

**y distance to image bottom border**

Distance to the bottom border of the image.

Feature value range: [0; number of rows of raster layers or depending on coordinates]

**Distance to line**

Distance to a line, which could be manually defined by the enter of two points that are a part of this line. Note, that the line has neither a start nor an end. Click with the right mouse button on the feature, select “Edit Feature...” and adapt the coordinates to your analysis.

Feature value range: [0; squareroot of  $(rows^2 + columns^2)$  or depending on the coordinates]

**Edit Parametrized Feature** [X]

Name: Distance to line [1]

First Coordinate (X): 0

First Coordinate (Y): 0

Second Coordinate (X): 500

Second Coordinate (Y): 500

OK Cancel

**to super-object**

Use the following features to describe an image object by its form relations to one of its super-objects (if there are any). Which super-object is to be referred to is defined

by editing the feature distance ( $n$ ). Especially when working with thematic layers these features might be of great interest.

$$e_{to\ Super\ Object} = \frac{e_{in}}{e}$$

#### Rel. inner border to super-object ( $n$ )

This feature is computed by dividing the sum of the border shared with other image objects that have the same super-object by the total border of the image object. If the relative inner border to the super-object is 1, the image object of concern is not situated on the border of its super-object. Use this feature to describe how much of an image object is situated at the edge of its super-object.



Feature value range: [0 ; 1]

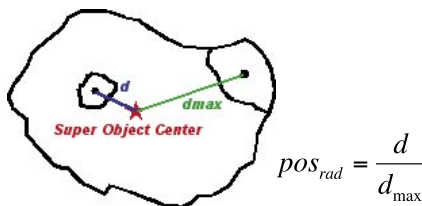
#### Rel. area of super-object ( $n$ )

The feature is computed by dividing the area of the image object of concern by the area covered by its super-object. If the feature value is 1, then the image object is identical to its super-object. Use this feature to describe an image object by the amount of area it covers of its super-object.

Feature value range: [0 ; 1]

#### Rel. rad. position to super-object ( $n$ )

The feature value is calculated by dividing the distance from the center of the image object of concern to the center of its super-object by the distance of the center of the most distant image object which has the same super-object.



Use this feature to describe an image object by its position relative to the center of its super-object.

Feature value range: [0 ; 1]

### Texture

All features concerning texture are based on sub-object analysis. This means you must have a level of sub-objects to be able to use them. Which sub-object level to use can be defined by editing the feature distance ( $n$ ). The texture features are divided in two groups: texture concerning the spectral information of the sub-objects and texture concerning the form of the sub-objects.

#### Layer value texture based on so

These features refer to the spectral information provided by the image layers.

#### Mean of so: stddev

Standard deviation of the different layer mean values of the sub-objects. At first this feature might appear very similar to the simple standard deviation computed from the single pixel values (layer values), but it might be more meaningful since (a reasonable segmentation assumed) the standard deviation here is computed over homogeneous and meaningful areas. The smaller the sub-objects, the more the feature value approaches the standard deviation calculated from single pixels.

Feature value range: [0 ; depending on bit depth of data]

#### Avrg. mean diff. to neighbors of so

The contrast inside an image object expressed by the average mean difference of all its sub-objects for a specific layer. This feature has a certain spatial reference, as a local contrast inside the area covered by the image object is described. For each single sub-object the layer  $L$  mean difference (absolute values) to adjacent sub-objects of the same super-object is calculated. The feature value is the mean value of the layer  $L$  mean differences.

Feature value range: [0 ; depending on bit depth of data]

**Form texture based on so**

The following features refer to the form of the sub-objects. The premise to use these features properly is an accurate segmentation of the image, because the sub-objects should be as meaningful as possible.

**Area of so: mean (n)**

Mean value of the areas of the sub-objects.

Feature value range: [0 ; scene size]

**Area of so: stddev (n)**

Standard deviation of the areas of the sub-objects.

Feature value range: [0 ; depending on image object shape]

**Density of so: mean (n)**

Mean value calculated from the densities of the sub-objects.

Feature value range: [0 ; depending on image object shape]

**Density of so: stddev (n)**

Standard deviation calculated from the densities of the sub-objects.

Feature value range: [0 ; depending on image object shape]

**Asymmetry of so: mean (n)**

Mean value of the asymmetries of the sub-objects.

Feature value range: [0 ; depending on image object shape]

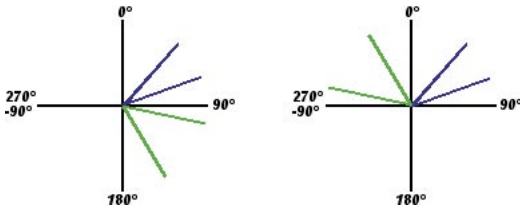
**Asymmetry of so: stddev (n)**

Standard deviation of the asymmetries of the sub-objects.

Feature value range: [0 ; depending on image object shape]

**Direction of so: mean [n]**

Mean value of the directions of the sub-objects. In the computation, the directions are weighted with the asymmetry of the respective sub-objects (the more asymmetric an image object, the more significant its main direction). Before computing the actual feature value, the algorithm compares the variance of all sub-object main directions with the variance of the sub-object main directions, where all directions between  $90^\circ$  and  $180^\circ$  are inverted (direction  $-180^\circ$ ). The set of sub-object main directions which has the lower variance is selected for the calculation of the main direction mean value weighted by the sub-object asymmetries.



Feature value range: [0 ; 180]

**Direction of so: stddev [n]**

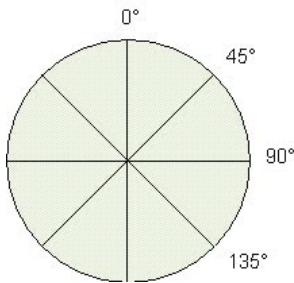
Standard deviation of the directions of the sub-objects. Again, the sub-object main directions are weighted by the asymmetries of the respective sub-objects. The set of sub-object main directions of which the standard deviation is calculated is determined in the same way as explained above (Direction of SO: Mean).

Feature value range: [0 ; 90]



### Texture after Haralick - a general overview

The grey level co-occurrence matrix (GLCM) is a tabulation of how often different combinations of pixel grey levels occur in an image. A different co-occurrence matrix exists for each spatial relationship. To receive directional invariance all 4 directions (0°, 45°, 90°, 135°) are summed before texture calculation. An angle of 0° represents the vertical direction, an angle of 90° the horizontal direction. In eCognition texture after Haralick is calculated for all pixels of an image object. To reduce border effects, pixels bordering the image object directly (surrounding pixels with a distance of one) are additionally taken into account. The directions to calculate texture after Haralick in eCognition are:



Every GLCM is normalized according to the following operation:  
where:

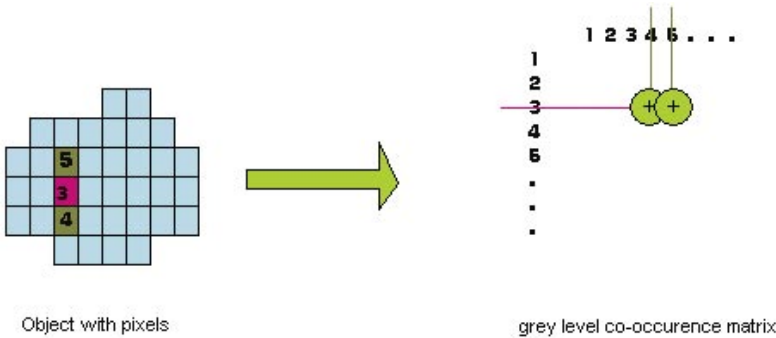
$$P_{i,j} = \frac{V_{i,j}}{\sum_{i,j=0}^{N-1} V_{i,j}}$$

- $i$  is the row number and  $j$  is the column number
- $V_{i,j}$  is the value in the cell  $i,j$  of the matrix
- $P_{i,j}$  is the normalized value in the cell  $i,j$
- $N$  is the number of rows or columns

The normalized GLCM is symmetrical. The diagonal elements represent pixel pairs with no grey level difference. Cells, which are one cell away from the diagonal, represent pixel pairs with a difference of only one grey level. Similarly, values in cells, which are two pixels away from the diagonal, show how many pixels have a 2 grey levels and so forth. The more distant to the diagonal, the greater the difference between the pixels' grey levels is. Summing-up the values of these parallel diagonals, gives the probability for each pixel to be 0, 1, 2 or 3 etc. different to its neighbour pixels.

Another approach to measure texture is to use a grey-level difference vector (GLDV) instead of the GLCM. The GLDV is the sum of the diagonals of the GLCM. It counts the occurrence of references to the neighbour pixels' absolute differences. In eCognition the GLCM and GLDV are calculated based on the pixels of an object. They are computed for each input channel. Within each "Texture after Haralick" feature you have the choice of either one of the above directions or of all directions.

The calculation of "Texture after Haralick" is independent of the image data's bit-depth. The dynamic range is interpolated to 8 bit before evaluating the co-occurrence. However, if 8 bit data is used directly the results will be most reliable. When using data of higher dynamic than 8 bit, the mean and standard deviation of the values is calculated. Assuming a Gaussian distribution of the values, more than 95% is in-between the interval:



The interval is subdivided into 255 equal sub-intervals to obtain an 8 bit representation.

$$\bar{x} - 3 \cdot \sigma < x < \bar{x} + 3 \cdot \sigma$$

### The calculation of the features

In the following for each "Texture after Haralick" feature its general calculation is described. The usable features in eCognition are sorted by their direction of concern: "All directions", "Direction 0°", "Direction 45°", "Direction 90°" and "Direction 135°". Further, each feature is calculated based upon the gray values of one selectable layer.

### GLCM: Homogeneity

If the image is locally homogenous, the value is high if GLCM concentrates along the diagonal. "Homogeneity"

$$\sum_{i,j=0}^{N-1} \frac{P_{i,j}}{1 + (i - j)^2}$$

weights the values by the inverse of the “Contrast” weight with weights, decreasing exponentially according to their distance to the diagonal.

#### GLCM: Contrast

“Contrast” is the opposite of “Homogeneity”. It is a measure of the amount of local variation in the Image. It increases exponentially as  $(i-j)$  increases.

$$\sum_{i,j=0}^{N-1} P_{i,j} (i-j)^2$$

#### GLCM: Dissimilarity

Similar to “Contrast”, but increases linearly. High if the local region has a high contrast.

$$\sum_{i,j=0}^{N-1} P_{i,j} |i-j|$$

#### GLCM: Entropy

The value for “Entropy” is high, if the elements of GLCM are distributed equally. It is low if the elements are close to either 0 or 1. Since  $\ln(0)$  is undefined, it is assumed that  $0 * \ln(0) = 0$ .

$$\sum_{i,j=0}^{N-1} P_{i,j} (-\ln P_{i,j})$$

#### GLCM: Angular Second Moment

$$\sum_{i,j=0}^{N-1} P_{i,j}^2$$

#### GLCM: Mean

The “GLCM Mean” is the average expressed in terms of the GLCM. The pixel value is not weighted by its frequency of occurrence itself, but by the frequency of its occurrence in combination with a certain neighbour pixel value.

$$\mu'_{i,j} = \frac{\sum_{i,j=0}^{N-1} P_{i,j}}{N^2}$$

**GLCM: Standard Deviation**

$$\sigma_{i,j}^2 = \sum_{i,j=0}^{N-1} P_{i,j} (i, j - \mu_{i,j})^2$$

Standard Deviation:

$$\sigma = \sqrt{\sigma_{i,j}^2}$$

“GLCM Standard Deviation” uses the GLCM, therefore it deals specifically with the combinations of reference and neighbour pixels. Thus, it is not the same as the simple standard deviation of grey levels in the original image. Calculating the “Standard Deviation” using i or j gives the same result, since the GLCM is symmetrical. “Standard Deviation” is a measure of the dispersion of values around the mean. It is similar to contrast or dissimilarity.

**GLCM: Correlation**

Measures the linear dependency of grey levels of neighbouring pixels.

$$\sum_{i,j=0}^{N-1} P_{i,j} \left[ \frac{(i - \mu_i)(j - \mu_j)}{\sqrt{(\sigma_i^2)(\sigma_j^2)}} \right]$$

**GLDV: Angular Second Moment**

High if some elements are large and the remaining ones are small. Similar to “GLCM Angular Second Moment”: it measures the local homogeneity.

$$\sum_{k=0}^{N-1} V_k^2$$

**GLDV: Entropy**

Since  $\ln(0)$  is undefined, it is assumed that  $0 * \ln(0) = 0$ :

The values are high if all elements have similar values. It is the opposite of “GLDV Angular Second Moment”.

$$\sum_{k=0}^{N-1} V_k (-\ln V_k)$$

**GLDV: Mean**

The mean is mathematically equivalent to the “GLCM Dissimilarity” measure above. It is only left here for compatibility reasons.

$$\sum_{k=0}^{N-1} k(V_k)$$

**GLDV: Contrast**

$$\sum_{k=0}^{N-1} V_k k^2$$

It is mathematically equivalent to the “GLCM Contrast” measure above. It is only left here for compatibility reasons.

**Note!** The calculation of any “Texture after Haralick” feature is very CPU demanding because of the calculation of the GLCM.

**Implementation and references for Texture after Haralick:**

Haralick features were implemented in eCognition according to the following references:

- R. M. Haralick, K. Shanmugan and I. Dinstein, “Textural Features for Image Classification”, IEEE Tr. on Systems, Man and Cybernetics, Vol SMC-3, No. 6, November 1973, pp. 610-621.
- R. M. Haralick, “Statistical and Structural Approaches to Texture”, Proceedings of the IEEE, Vol. 67, No. 5, May 1979, pp. 786-804.
- R. W. Conner and C. A. Harlow, “A Theoretical Comparison of Texture Algorithms”, IEEE Tr. on Pattern Analysis and Machine Intelligence, Vol PAMI-2, No. 3, May 1980

**Feature distance**

In some features, a distance can be edited in order to define the feature specifically.

**Hierarchical distance between objects on different levels in the image object hierarchy**

Starting from the current level the number in brackets indicates the hierarchical distance of feature levels containing the respective objects (sub-objects or super-objects). The default setting is one level (1) but you can modify this number in two ways: before inserting the class in the “Insert Expression” dialog by clicking on the expression with the right mouse button, or afterwards in the “Membership Function” dialog by clicking on the button “Feature Distance.” A dialog appears where you can insert the number of feature levels you want to refer to. But pay attention, for this feature attribute the distance of feature levels will be changed in all applications.

### Distance between objects on the same level in the image object hierarchy

If you want to analyze neighborhood relations between image objects on the same level in the image object hierarchy, the feature distance expresses the spatial distance (in pixels) between the image objects. The default value is 0, i.e., only neighbors that have a mutual border are regarded. The value can be edited the same way as described above.

### Hierarchy

All the following features refer to the embedding of an image object in the entire image object hierarchy.

#### Level

The number of the image object level an image object is situated in. You will need this feature if you perform classification on different image object levels to define which class description is valid for which level.

Feature value range: [1 ; number of image object levels]

#### Num higher levels

The number of image object levels situated above the image object level the object of concern is situated in. This is identical to the number of super-objects an image object may have.

Feature value range: [1 ; number of image object levels -1]

#### Num sublevels

The number of image object levels situated below the image object level the object of concern is situated in.

Feature value range: [1 ; number of image object levels -1]

#### Num neighbors

The number of the direct neighbors of an image object (i.e., neighbors with which it has a common border) on the same level in the image object hierarchy.

Feature value range: [0 ; number of pixels of entire scene]

#### Num subobjects

The number of sub-objects of an image object on the next lower level in the image object hierarchy.

Feature value range: [0 ; number of pixels of entire scene]

### Thematic attributes

Thematic attributes can only be used if a thematic layer has been imported into the project. If this is the case, all thematic attributes in numeric form that are contained in the attribute table of the thematic layer can be used as features in the same manner as you would use any other feature provided by eCognition.

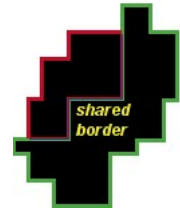
### Class-related features

#### Relations to neighbor objects

Use the following features to describe an image object by the classification of other image objects on the same level in the image object hierarchy.

##### Rel. border to class

The ratio of the border of an object shared with neighboring objects assigned to a defined class to the total border length. If the relative border of an object to objects of a certain class is 1, the image object is totally embedded in these image objects.



Feature value range: [0 ; 1]

##### Border to class

The absolute border of an image object shared with neighboring objects of a defined classification. If you use georeferenced data, the feature value is the real border to image objects of a defined class; otherwise it is the number of pixel edges shared with the adjacent image objects, as by default the pixel edge-length is 1.

Feature value range: [0 ; number of pixels of scene]

##### Rel. area of class

Area covered by image objects assigned to a defined class in a certain perimeter (in pixels) around the image object concerned divided by the total area of image objects inside this perimeter. The radius defining the perimeter can be determined by editing the feature distance.

Feature value range: [0 ; 1]

**Existence of class**

Existence of an image object assigned to a defined class in a certain perimeter (in pixels) around the image object concerned. If an image object of the defined classification is found within the perimeter, the feature value is 1 (= true), otherwise it would be 0 (= false). The radius defining the perimeter can be determined by editing the feature distance.

Feature value range: [0 (false) ; 1 (true)]

**Distance to class**

The distance (in pixels) of the image object concerned to the closest image objects assigned to a defined class.

Feature value range: [0 ; depending on scene size]

**Layer mean diff. to class**

The mean difference of the layer  $L$  mean value of the image object concerned to the layer  $L$  mean value of all image objects assigned to a defined class.

Feature value range: [0 ; depending on bit depth of data]

**Relations to sub-objects**

These features refer to existing class assignments of image objects on a lower level in the image object hierarchy. Which of the lower levels to refer to can be determined by editing the feature distance.

**Rel. area of class**

The area covered by sub-objects assigned to a defined class divided by the total area of the image object concerned.

Feature value range: [0 ; 1]

**Existence of class**

Checks if there is at least one sub-object assigned to a defined class. If there is one, the feature value is 1 (= true), otherwise the feature value is 0 (= false).

Feature value range: [0 (false) ; 1 (true)]



**Area of class**

The absolute area covered by sub-objects assigned to a defined class. If your data are georeferenced, the feature value represents the real area (see Form > Area).

Feature value range: [0 ; scene size]

**Number of class**

The number of sub-objects assigned to a defined class.

Feature value range: [0 ; number of all image objects]

**Relations to super-objects**

This feature refers to existing class assignments of image objects on a higher level in the image object hierarchy.

**Existence of class**

Checks if the super-object is assigned to a defined class. If this is true, the feature value is 1, otherwise 0.

Feature value range: 0 (false) ; 1 (true)

**Similarity to classes**

Similarities work in a way similar to class inheritance. Adding a similarity to a class description basically does the same as inheriting from this class. But since similarities are part of the class description, they can be used with much more flexibility than can inheritance. This is especially true when they are combined inside logical terms.

One very interesting method is the use of inverted similarities as a kind of negated inheritance: consider a class *Bright* defined by large channel mean values. You can define a class *Dark* by inserting a similarity to bright and inverting it, which yields the logical connection *Dark* is not *Bright*.

**Membership to**

In some cases it is of importance to incorporate the membership value to different classes in one class. This function allows you to explicitly address the membership values to different classes.

Feature value range: [0 ; 1]

**Classified as**

The idea of this feature is to enable the user to refer to the classification of an object without regard to the membership value.

Feature value range: 0 (false) ; 1 (true)

**Global features****Global scene related****Number of pixels**

Number of pixels in the pixel layer of the image.

**Number of objects**

Number of objects of any class on all levels of the scene. Includes unclassified objects.

**Number of layers**

Number of layers which are imported in the scene.

**Resolution**

Resolution of the image as set in the metadata of the image. 1 if no resolution is set.

**Global layer mean of scene**

Mean value for the selected layer.

**Global layer stddev of scene**

Standard Deviation for the selected layer.

**Global Class related****Global number of objects of**

The absolute number of all objects of the selected class on all levels.

**Global area of**

The absolute area of all objects of the selected class on all levels.

**Global layer mean of**

The mean of all objects of the selected class on the selected layer.

**Global layer stddev of**

The standard deviation of all objects of the selected class for the selected layer.

### Customized features

eCognition comes with two possibilities to create new features based on the existing ones: arithmetic and relational customized features. New customized features can be named and also saved separately.

#### Customized features: arithmetic

A relatively simple method to create new features is to combine existing ones based on arithmetic operators. This method is similar to that of spectral arithmetics known from pixel based procedures such as vegetation indices. However, in eCognition all arbitrary feature and not only tone can be combined, which opens a whole bandwidth of interesting possibilities.

#### Customized features: relational

A more sophisticated method is the creation of relational customized features. One specific feature can be chosen from the feature list in eCognition. The basic principle of this method is that for each image object, different relations concerning this chosen feature can be computed between this particular object and a specified selection of objects or sub-objects in its neighborhood. The specific relations which can be chosen are described in the following.

**To surrounding objects:** Relations to surrounding objects target either the directly adjacent neighbor objects or, when a distance is defined, all objects within a certain radius.

#### mean value

Calculates the mean value of the feature values of an image object and the objects in a specified surrounding. Note that for averaging, the feature values are weighted by the size of the respective objects.

#### std dev [value]

Calculates the standard deviation of the features values of an image object and the objects in a specified surrounding.

**mean difference**

Calculates the mean difference between the feature value of an object and the feature values of the objects in a specified surrounding. Note that for averaging, the feature values of the objects in the surrounding are weighted by the respective size.

**mean abs. difference**

Calculates the mean absolute difference between the feature value of an object and the feature values of the objects in a specified surrounding. Note that for averaging, the absolute difference to each object in the surrounding is weighted by the respective size.

**min**

Returns the minimum value of the feature values of an image object and its neighbors of a selected class.

**max**

Returns the maximum value of the feature values of an image object and its neighbors of a selected class.

**mean difference to higher values**

Calculates the mean difference between the feature value of an object and the feature values of the objects in a specified surrounding which have higher values than the object itself. Note that for averaging, the feature values of the objects in the surrounding are weighted by the respective size.

**mean difference to lower values**

Calculates the mean difference between the feature value of an object and the feature values of the objects in a specified surrounding which have lower values than the object itself. Note that for averaging, the feature values of the objects in the surrounding are weighted by the respective size.

**rate of higher value area**

Calculates the portion of the area of the objects in a specified surrounding which have higher values for the specified feature than the object itself to the area of all objects in the specified surrounding.

**rate of lower value area**

Calculates the portion of the area of the objects in a specified surrounding which have lower values for the specified feature than the object itself to the area of all objects in the specified surrounding.

**rate of higher values**

Calculates the portion of the feature value difference times the area of the objects which have higher values for the specified feature than the object itself to the feature value difference times the area of all objects in the specified surrounding.

**rate of lower values**

Calculates the portion of the feature value difference times the area of the objects which have lower values for the specified feature than the object itself to the feature value difference times the area of all objects in the specified surrounding.

$a_N \sim$  = area of the neighbor object or rel. border length between this neighbor object and the object to be classified if  $\text{dist} = 0$ .

$\sum_N$  = sum over all neighbors

**To sub-objects:** relations to sub-objects target either the objects of the subsequent level or, when a distance is defined, the objects of a level which is the defined number of levels lower. The following relations to sub-objects can be evaluated:

**average value (avrg.value)**

Calculates the mean value of the feature for the sub-objects on the specified sublevel. Note that for averaging the feature values are weighted by the size of the respective sub-objects.

**standard deviation (std.dev)**

Calculates standard deviation of the feature concerning the sub-objects on the specified sublevel. This relational feature allows computation of a value for the structural heterogeneity of the given feature without consideration of spatial distribution.

**average absolute difference to neighbors (avrg.abs.diff to neighbors)**

Calculates the mean value of the contrast (mean absolute difference for a specified feature) of each sub-object on the specified sub-level to its direct neighbor objects. Note that for averaging the contrast are weighted by the size of the respective sub-objects.

This relational feature allows computation of a value for the structural heterogeneity of the given feature with consideration of spatial distribution.

## TERMS

### Standard nearest neighbor

The standard nearest neighbor is a nearest neighbor classifier that uses an identical feature space throughout the whole eCognition project. It is edited in a special dialog which can be opened via the menu item “Classification > Nearest Neighbor > Edit Standard NN Feature Space....” If any changes are made to the standard nearest neighbor feature space, these changes will be applied to all classes whose description contain the standard nearest neighbor.

### Logical terms

Logical terms are used to combine fuzzy expressions, such as membership functions, nearest neighbor classifiers, similarities, and logical terms with standard fuzzy logic operators.

eCognition provides the following operators:

<b>and (min)</b>	Fuzzy logical “and”-operator using the minimum function.
<b>and (*)</b>	Product of feature values.
<b>or (max)</b>	Fuzzy logical “or”-operator using the maximum function.
<b>mean (arithm.)</b>	Arithmetic mean of the assignment values.
<b>mean (geo.)</b>	Geometric mean of the assignment values.

Example: Consider four membership values of 100 % each and one of 0 %. The “and”-operator yields the minimum value, i.e., 0 %, whereas the “or”-operator yields the maximum value, i.e., 100 %. The arithmetic mean yields the average value, in this case 80 %.

## Classification-based segmentation and correction of image objects

Since eCognition's multiresolution segmentation is a knowledge-free method for the generation of image objects which is exclusively determined by a homogeneity criterion consisting of color and form homogeneity, the resulting image objects can only be regarded as object primitives. In addition, multiresolution segmentation yields image objects of a similar size, while different structures in an image are embedded in different scales of resolution. Consequently, one image object level with its characteristic resolution cannot represent all structures in the image. This can only be achieved using more than one image object level.

The tools described in this chapter are of great importance, because they allow the combination of shape information of different scales. Information of an image object level with a fine resolution can thus be used to correct a level with a coarser resolution.

To understand an image and to obtain information of a fine resolution level, in many cases a coarser resolution is necessary. This problem is analogous to human perception, where the discerning eye must be at a certain distance to realize what kind of structure is displayed in an image. Once you have obtained an overview, the detailed information can be analyzed. Image objects on a coarser level, e.g., urban and rural structures, might undergo border optimization to correct their shape.

To handle these problems, eCognition provides a second method of image object generation which is not based on a homogeneity criterion, but uses a knowledge base instead. The knowledge base taken is a classification of the image object primitives yielded by multiresolution segmentation.

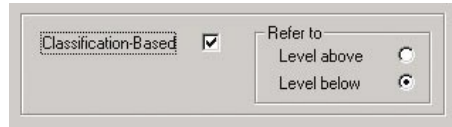
The formulation of rules for the so-called classification-based segmentation is carried out by editing structure groups. A structure group is a collection of classes representing the same structure in an image and can consist of classes defined for different levels in the image object hierarchy.

eCognition distinguishes different types of classification-based image object generation:



### Classification-based multiresolution segmentation

In some cases it might be necessary to perform a multiresolution segmentation only on certain objects of interest. This means the criteria of homogeneity are only applied to distinct areas, covered by already classified objects.



### Classification-based fusion

The principle of classification-based fusion is simple: all adjacent image objects that represent identical structures, or are parts of identical structures, are merged into one new image object. This way, a number of

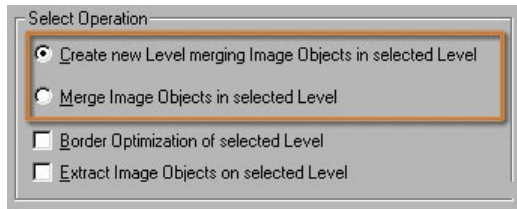


image objects forming an urban area can be merged into one image object representing an entire urban area. A structure is defined by organizing all classes which semantically form this structure in a structure group. You can either create a new image object level in which the objects are merged, or merge them in the same level. In the latter case, former image objects will be lost.




-  not classified or classes of no structure group
-  Classes of structure group 1
-  Classes of structure group 2

Image object level before classification-based fusion:



The same image object level after classification-based fusion:



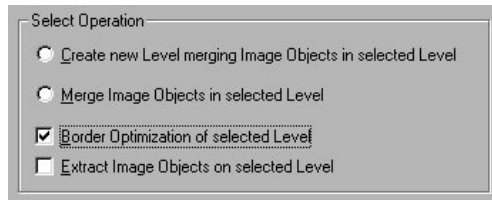
Examples of classification-based fusion can be found in [Functional Guide > Image Object Generation II: Classification-based Segmentation / Refinement](#) and [Guided Tours > Analysis of the Degree of Urban Impervious Area](#).

### Classification-based shape correction of image objects depending on sub-objects

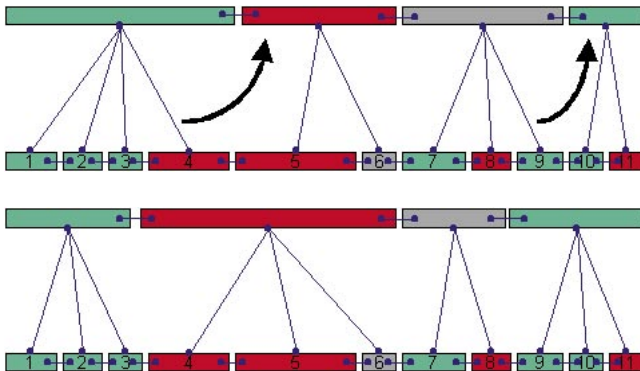
The main purpose of the following two tools is the shape correction of image objects based on sub-objects. To perform this procedure, in addition to the classified level of image objects which should be refined, a level of classified sub-objects is needed. Again, the organization of classes to structure groups is the knowledge base for those operations. The basic principle is that a sub-object which was assigned to a class in another structure group other than the class of its super-object is treated as heterogeneous or as “not belonging” to this super-object. The result is a regrouping of this sub-object. Two different techniques are available for this purpose.

### Classification-based border optimization

Sub-objects situated at the border of their respective super-object and member of another structure group will be re-grouped to become sub-object of a neighbor super-object, if this neighbor super-object is of the same structure group as the sub-object.



The result of such a border optimization appears above. The image regions represented by the affected super-objects alter: whereas the former super-object loses the area represented by the re-grouped



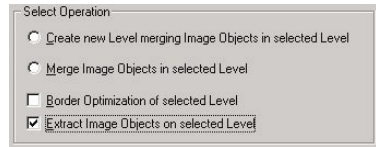
sub-object, the new super-object gains this area. Border optimization yields an image object level with shape corrected objects.

An example for border optimization can be found in [Guided Tours > High resolution Aerial Scan](#).

### Classification-based image object extraction

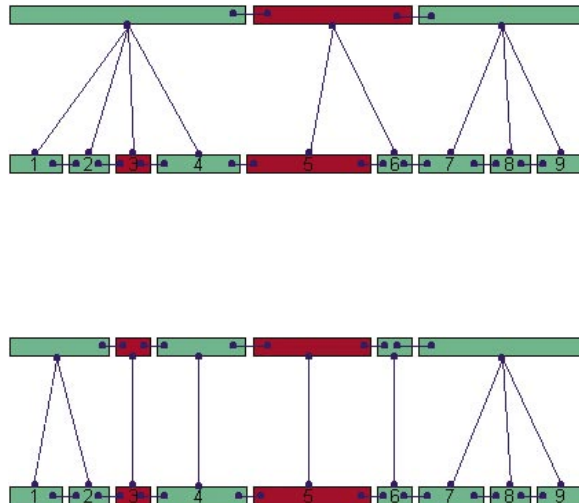
In contrast to border optimization, the extraction of sub-objects handles all sub-objects, even image objects situated geographically within a super-object.

If a sub-object is assigned to a different structure group than its super-object, the hierarchical connection between these two objects will be deallocated. The super-object loses the area represented by the sub-object. Instead, a new image object identical to the sub-object will be produced and inserted in the network as its new super-object.



An additional application of this procedure is to project image objects embedded in different levels of the hierarchical network of image objects onto the same image object level.

An example for this can be found in [Functional Guide > Image Object Generation II: Classification-based Segmentation/Refinement](#).



## Multisource data fusion

### Bridging remote sensing and GIS

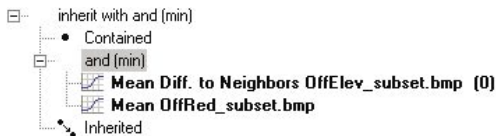
Multisource data fusion is supported by eCognition through different techniques. In this context, data fusion means the simultaneous utilization of image data of various origin, the only precondition being georeferenced data, i.e., the size of the image and of the pixels in the image layers and thematic layers are the same. In this case, the different information channels can be brought into a reasonable relation to each other. This chapter gives an overview of the possibilities of multisource data fusion.

### Arbitrary layer weights for the segmentation of images

eCognition enables you to determine the influence of any imported image layer on the process of image object generation: for each layer you can determine how much of the information provided by it should be used in the segmentation process ([Functional Guide > Image Object Generation I: Multiresolution Segmentation](#)) by editing its weight. This means that you can influence the generation of image objects in a way that best suits the posed problem. If, for instance, your project contains an image layer representing an elevation model, this elevation information can be incorporated into the segmentation process together with information of other image layers containing spectral information.

### Simultaneous utilization of the information provided by different image layers for classification

When classifying a segmented image, you can use information from different image layers simultaneously. Imagine working on an aerial photograph along with an elevation model. Roofs can be classified using their spectral value derived from an image layer containing spectral information along with their elevation value relative to their neighborhood derived from an image layer containing the elevation model.

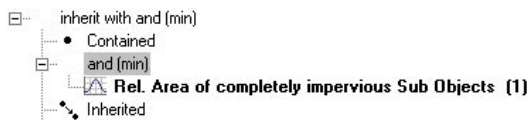


### Use one layer for segmentation, a different one for classification

Often image layers are weighted with a 0 during the segmentation process, which means that none of the information provided is used for the generation of image object primitives. However, the information provided can be utilized without restraint for classification. Consider a project containing an elevation model with a lower resolution than the image layers providing the spectral information. Due to the lower resolution, the elevation model will probably not be considered in the segmentation process. The information provided by it, however, can be utilized for the classification.

### Relate different scale-dependent structures

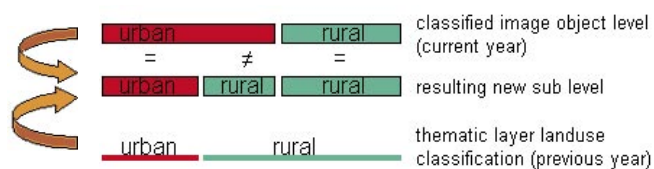
Image object levels of different resolutions can be created by weighting image layers differently. This way different scale-dependent structures can be represented on different levels in the image object hierarchy. Since each image object level can utilize the information from any other image object level, these structures can be brought into relation to each other. Consider a segmentation based on an imported thematic layer representing a land register as an example. The information from a sublevel with a finer segmentation can be utilized for the classification of such a thematic layer, while the segmentation of this sub-level is built on information provided by other image layers.



### Use structural information from a coarser image object level to limit the borders of sub-objects throughout the segmentation

If a sub-level is created for an existing image object level, all new sub-objects will be situated within the borders of the super-objects. This fact can be used for applications like change detection. If you intend to compare a classified image object level of the current year with an already existing thematic land use classification (previous year), you can use the borders of both levels for the segmentation of a new sublevel. This new level is segmented as a sublevel of the current level, with segmentation depending only on the thematic layer (by weighting all image layers with 0). Because the borders of the super-objects from the current classified level have to be considered in the segmentation, the new sub-level is divided as shown below. In doing so, changes between the thematic layer information and the current image object level are extracted as separate

image objects. Further classification of these “change” objects can decide how to handle them.



Information for the new segmentation

## Classification evaluation in eCognition

Evaluating the quality of a classification result is of high importance in remote sensing since it gives evidence of how well the generated or used classifier is capable of extracting the desired objects from the image. Commonly, as a first evaluation, simple visual inspection can be used to evaluate the plausibility of the classification results. Nevertheless, this is just a subjective method and thus hardly to be quantified or even capable of being expressed in comparable values. Moreover, it is necessary to obtain information about the classification stability and how capable the classes are to extract the desired image information. Beside the classical methods of accuracy assessment, special methods based upon fuzzy concepts can be used.

### Common accuracy measures

In order to measure the quality of a classifier and to compare and evaluate classifications with respect to their suitability for specific applications, accuracy measures are used. Mostly they are derived on the basis of a comparison of the classification in question with another classification. This latter classification is often obtained with different methods, e.g., by on-site ground measurements, and is considered reliable and true, which is why sometimes the term “ground truth” is used. We will here use the term “reference classification” to emphasize that it is essentially also a classification, the reliability of which must be assured and cannot be taken for granted. Special care has, e.g., to be taken if the reference classification is obtained with data taken at a different time than was the data for the classification to be evaluated. Moreover, it must be assured that the reference classification and the classification in question carry comparable information. This means that they have the same classes, or at least classes which can be assigned to each other’s classes by merging, and that they both have the same grid, i.e., speaking in digital terms, that the pixels have the same location and spatial extent on the ground. In the sequel we take all this for granted (assuming that if necessary some pre-processing of classification and reference classification was performed) and we assume especially that both classification and reference classification assign the same classes in a crisp way, i.e., that a pixel is assigned to exactly one class. And we shall assume that the pixels of the reference classification are a subset of those of the classification.

We can then derive a so-called confusion table by counting how many of the pixels classified as *class i* in the classification are of *class k* in the reference classification. We denote this number by  $a_{ik}$  and write it in row  $i$  and column  $k$  of the table.

This table, sometimes referred to as *confusion matrix* or *error matrix*, contains all the information about the relation between classification and reference classification. However, it is often useful to derive from it some characteristic numbers which simplify the accuracy assessment of the classification (Congalton, 1991).

	Reference classification					
Classification		Class 1	Class 2	...	Class N	
	Class 1	$a_{11}$	$a_{12}$		$a_{1N}$	$\sum_{k=1}^N a_{1k}$
	Class 2	$a_{21}$	$a_{22}$		$a_{2N}$	$\sum_{k=1}^N a_{2k}$
	...	...	...		...	...
	Class N	$a_{N1}$	$a_{N2}$		$a_{NN}$	$\sum_{k=1}^N a_{Nk}$
		$\sum_{k=1}^N a_{k1}$	$\sum_{k=1}^N a_{k2}$		$\sum_{k=1}^N a_{kN}$	$n = \sum_{i,k=1}^N a_{ik}$

A first such number is *overall accuracy*  $OA$ . It is the proportion of all reference pixels which are classified correctly (in the sense that the class assignment of the classification and of the reference classification agree). It can be computed from the confusion table by

$$OA = \frac{\sum_{k=1}^N a_{kk}}{\sum_{i,k=1}^N a_{ik}} = \frac{1}{n} \sum_{k=1}^N a_{kk}$$

where  $n$  is the number of all reference pixels. So  $OA$  is the sum of the diagonal entries of the confusion table divided by the number of all reference pixels. Overall accuracy is a very coarse measure. It gives no information about what classes are classified with good accuracy. In fact, a classification with poor overall accuracy may find one certain class with high accuracy, although it confuses all others, and thus be of interest for certain applications. Therefore, other measures are useful.

One such measure, *producer's accuracy*  $PA(class_i)$ , estimates the probability that a pixel which is of *class*  $i$  in the reference classification is correctly classified. It is thus for each *class*  $i$  the proportion of pixels where classification and reference classification agree in *class*  $i$  and the reference pixels are classified as this class. As the total number of the pixels of *class*  $i$  in the reference classification is obtained as the sum of column  $i$  in the confusion table, we have

$$PA(class_i) = \frac{a_{ii}}{\sum_{t=1}^N a_{ti}}$$



Producer's accuracy is actually a measure for the producer of a classification, which tells him how well the classification agrees with the reference classification. It gives, however, no information about how well the classification predicts a class, i.e., it gives no information about the probability that a pixel classified as *class i* is actually of *class i*. This is the primary interest of a user of a classification and an estimate of this probability is thus called *user's accuracy*  $UA(class_i)$ . We estimate this probability by the proportion of pixels where classification and reference classification agree in *class i* and the number of all reference pixels classified as *class i* by the classification. Now the total number of pixels which are classified as *class i* is obtained by the sum of row *i* of the confusion table, so that we have

$$UA(class_i) = \frac{a_{ii}}{\sum_{k=1}^N a_{ik}}$$

Let us give a small example to clarify the differences between *PA* and *UA*. Consider the following confusion table.

Classification	Reference classification				
		Class 1	Class 2	Class N	
Class 1		50	0	0	50
Class 2		40	100	60	200
Class N		10	0	40	50
		100	100	100	$n = 300$

We obtain here as producer's accuracy for class 1  $PA(class_1) = 50/100 = 0.5$ , meaning that only 50 % of the class 1 pixels of the reference classification are found by the classification. On the other hand, the value  $UA(class_1) = 50/50 = 1.0$  shows that the user can rely completely on the classification here: whenever a pixel is classified to class 1, this is correct.

The situation is quite different when we consider class 2. Here we get  $PA(class_2) = 100/100 = 1.0$ , i.e., the classification classifies all class 2 pixels of the reference classification correctly. But this time, as  $UA(class_2) = 100/200 = 0.5$ , the user can be only 50 % sure that a pixel classified as class 2 is in fact class 2. Such a pixel is in half of all cases confused with other classes.

For class 3 we find  $PA(class_3) = 40/100 = 0.4$  and  $UA(class_3) = 40/50 = 0.8$ . So although again producer's accuracy is quite low, a user can rely on a pixel classified as class 3 to 80 %.

The overall accuracy in this example,  $OA = (50 + 100 + 40)/300 = 0.63$ , is rather low. It gives no information to the producer or user of the classification as to which classes are well detected, although some of them can be of interest in certain applications (e.g., class 1 can be of interest in the design of a classifier and classes 2 and 3 can be of interest in the use of the classification).

In order to compensate for the different interests of users and producers of a classification, certain combinations of producer's and user's accuracy are used. A first such choice is the *product*

$$PA \cdot UA(class_i) = PA(class_i) \cdot UA(class_i)$$

or the *minimum*

$$PA \wedge UA(class_i) = \min(PA(class_i), UA(class_i))$$

of the two. They give values close to one if both accuracies are high and are close to zero if any of the two is low. In the example above we get for classes 1, 2 and 3 respectively the values for  $PA \cdot UA$  0.5, 0.5 and 0.32, and for  $PA \wedge UA$  we find 0.5, 0.5 and 0.4.

Other possibilities are the measure of Hellden [3,4]

$$HA(class_i) = \frac{2a_{ii}}{\sum_{k=1}^N a_{ik} + \sum_{l=1}^N a_{il}}$$

which is the *harmonic mean* of  $PA(class_i)$  and  $UA(class_i)$ , i.e.,

$$HA(class_i) = \frac{2}{\frac{1}{PA(class_i)} + \frac{1}{UA(class_i)}} = \frac{PA(class_i) \cdot UA(class_i)}{(PA(class_i) + UA(class_i))/2}$$

or the measure of Short [4,5]

$$SA(class_i) = \frac{a_{ii}}{\sum_{k=1}^N a_{ik} + \sum_{l=1}^N a_{il} - a_{ii}}$$

For Short's measure we find

$$SA(class_i) = \frac{1}{\frac{1}{PA(class_i)} + \frac{1}{UA(class_i)} - 1}$$

$$= \frac{PA(class_i) \cdot UA(class_i)}{PA(class_i) + UA(class_i) - PA(class_i) \cdot UA(class_i)}$$

The interpretation of this accuracy measure is not so easy. Let us only mention that it is the quotient of the product t-norm  $t(\alpha, \beta) = \alpha \cdot \beta$  and the associated t-conorm  $s(\alpha, \beta) = 1 - t(1 - \alpha, 1 - \beta)$  for  $\alpha = PA(class_i)$  and  $\beta = UA(class_i)$ .

In our above example, the values of  $HA$  for classes 1, 2 and 3 are 0.67, 0.67 and 0.53. The corresponding values for  $SA$  are 0.5, 0.5 and 0.36. Each of these measures shows that the classification for classes 1 and 2 is better than the one for class 3.

$$\left. \begin{array}{l} SA(class_i) \\ PA \cdot UA(class_i) \end{array} \right\} \leq PA \wedge UA(class_i) \leq HA(class_i)$$

which says that in a certain sense, Short's accuracy measure or the product of producer's and user's accuracies is the most pessimistic, and Hellden's accuracy measure is the most optimistic. The use of producer's accuracy or user's accuracy alone should be treated with caution, as both answer completely different questions.

There is another accuracy measure which has attained wider interest in the classification community, namely Cohen's kappa coefficient [1,4]. It follows, however, a different idea. Whereas overall accuracy,  $OA$ , checks how many of all pixels are classified correctly, assuming that the reference classification is true, here it is assumed that both classification and reference classification are independent class assignments of equal reliability; how well they agree is what is measured. The big advantage of the kappa coefficient over overall accuracy is that kappa takes chance agreement into account and corrects for it. Chance agreement means here the probability that classification and reference classification agree by mere chance. Assuming statistical independence we obtain for this probability the estimation

$$\begin{aligned} P_c &= \sum_{k=1}^N P(\text{classification classifies class}_k \text{ and reference classification classifies class}_k) = \\ &= \sum_{k=1}^N P(\text{classification classifies class}_k) \cdot P(\text{reference classification classifies class}_k) = \\ &= \sum_{k=1}^N \frac{\sum_{i=1}^N a_{ki}}{n} \cdot \frac{\sum_{i=1}^N a_{ik}}{n} = \frac{1}{n^2} \sum_{k=1}^N \left( \sum_{i=1}^N a_{ki} \cdot \sum_{i=1}^N a_{ik} \right) \end{aligned}$$

With this the kappa coefficient is defined as

$$\kappa = \frac{P_o - P_c}{1 - P_c}$$

where  $P_o = \frac{1}{n} \sum_{i=1}^N a_{ii}$  is the proportion of observed agreement and  $P_c$  as derived above is the proportion of chance agreement. Note that the kappa coefficient may have negative values (whereas the other accuracy measures treated so far range between 0 and 1) and that  $\kappa = 1$  means perfect agreement between classification and reference classification. There is also a kappa coefficient for the classes (Rosenfield & Fitzpatrick-Lins, 1986) defined by

$$\kappa(class_i) = \dots \frac{a_{ii} \left( \sum_k a_{ik} \right) - \sum_k a_{ik} \sum_k a_{ki}}{\sum_k a_{ki} \sum_k a_{ik} - \sum_k a_{ik} \sum_k a_{ki}}$$

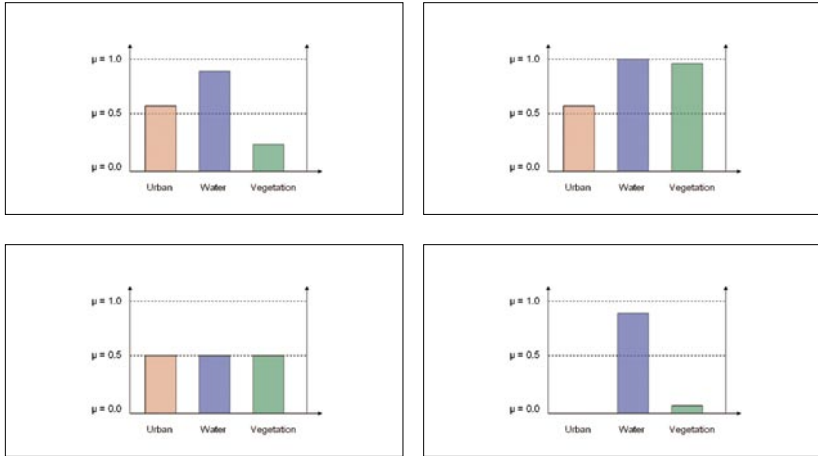
which can be used to assess the agreement of the two classifications for each class. The kappa coefficients are often criticized for the assumption of statistical independence and should therefore be treated with care.

#### Advanced classification evaluation based on fuzzy classification

When using fuzzy classification methods, objects can belong to several classes but with different degrees of membership, which is the case when class descriptions overlap. Thus, to evaluate the reliability or stability of classes it is necessary to survey the different degrees of membership of the classified objects. Objects whose feature values are within these overlapping ranges can be seen as ambiguous objects, since they fulfill the criteria of more than one class. Although fuzzy concepts make it possible to describe these ambiguities, the main aim of each classification should be to define classes as unambiguously as possible. Note: obtaining ambiguous objects does not mean that the objects are misclassified; it rather means that there is no class to which these objects belong to explicitly. Hence, regarding the objects' statistics of the degrees of membership to the classes helps to evaluate the quality of the classes. In other words: the less ambiguous the objects, the more usable the classification results. Likewise, one can say: the more distinctly the classes differentiate the objects, the more clearly they express the image's content.

To quantify a class's quality regarding the statistics of its objects' degrees of membership is an appropriate method: the more objects having a membership degree of 1 to just this class, the better the class is, and vice versa. In addition, the statistics and some parameters such as minimum, maximum, standard deviation and mean of the several degrees of membership can give more evidence.

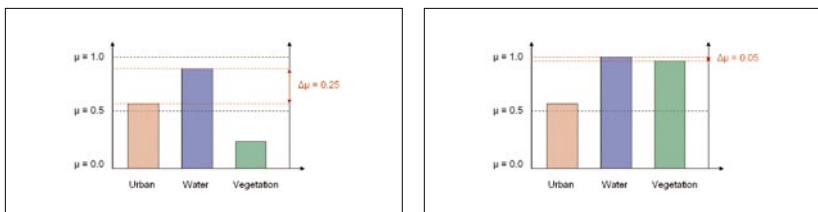
The figures on the next page show the degrees of membership ( $\mu$ ) of four objects classified as water, whereby the membership threshold to become classified is at  $\mu=0.5$ . All examples show ambiguous objects since there is no object classified with a degree of membership  $\mu=1.0$  to water only.

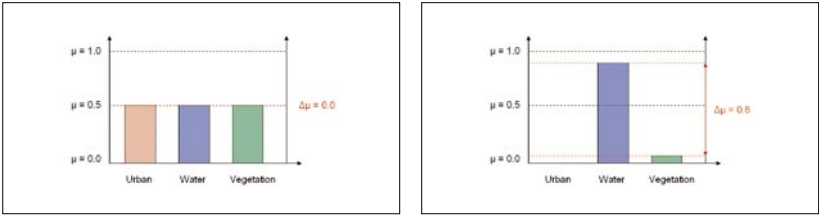


Assuming the class *water* consists only of these four objects, the statistics of the objects'  $\mu$  would look like:

No. of objects	Minimum	Standard deviation	Maximum	Mean
4	0.5	0.2	1.0	0.8

The table can be interpreted as follows: at least one object fulfills the class description completely, but there are some which do not. There is also at least one object which meets the class criteria poorly. But in general the objects have a membership degree  $\mu = 0.8 \pm 0.2$ . This can be interpreted as “most objects of the class meet the class’s criteria sufficiently.” Comparing the diagrams with the table it is obvious that the  $\mu$ -statistics for water only are not sufficient. Moreover, is it sufficient to evaluate the  $\mu$  statistics relatively? Especially a comparison between the best and second best  $\mu$ -values gives more evidence about the capability to separate the objects unambiguously. A simple operator which expresses the relativity between two values is their difference. The higher the  $\mu$ ’s difference, the more unambiguously an object belongs to its class.





For our class *water* the differences are:

object 1:  $0.8 - 0.55 = 0.25$ ; object 2:  $1.0 - 0.95 = 0.05$ ;  
object 3:  $0.5 - 0.5 = 0.0$ ; object 4:  $0.9 - 0.1 = 0.8$ ;

Regarding the statistics of the differences between the best and second best,  $\mu$  looks like:

No. of objects	Minimum	Standard deviation	Maximum	Mean
4	0.00	0.37	0.8	0.27

This table can be interpreted as: there is at least one object which belongs to another class with the same degree of membership as to *water* (minimum = 0.00). There is no object of the class *water* which does not belong to another class at the same time. If this were the case, a maximum of 1.0 would occur. In general the objects of water can only be poorly separated from other classes (mean =  $0.27 \pm 0.37$ ).

What are the consequences of this reasoning? Regarding the class descriptions (not only that of water) obviously their membership functions do overlap in the value ranges. To solve the problem either the membership functions must be adjusted to avoid overlapping value ranges or other features must be found which are more capable of distinguishing water from other classes. Beside, it is possible that the data itself only gives poor information for formulating clearly discriminating membership functions. For the last case, this means the class descriptions are well chosen and thus their membership functions describe the vagueness of the desired classes when using illegible image data.

The above examples showed one basic advantage of fuzzy classification methods: due to handling with degrees of membership instead of binary membership values, it is possible to evaluate the classifier's capability to extract the desired object classes from an image. Additionally, it is possible to detect unstable, unreliable classes. Nevertheless, it is necessary to compare the obtained classification results with the real world using

methods of accuracy assessment as described in the chapter above. Establishing fuzzy accuracy measures is a current research topic.

## Aspects of human perception

### Some facts and theory

Human perception does not simply register. It is a highly active and integrated process, which unfolds in many steps between the direct sensual inputs on the one hand, and categories and our knowledge of the world on the other.

An interesting perspective on this process is given by constructivism, one of the most influential streams in modern epistemology, which has a strong impact on the field of artificial intelligence. It describes each perception as a new, active construction of an imagination about what is outside, invoked by the confrontation of sensual inputs with our concepts and knowledge about the world. We see, for instance, only the front of a tree, but internally we add everything to it that is connected to our idea or experience of a tree: that it has a back also, that it is compact, consists of wood, that there can be some more trees around.... Evaluation of experiences is not done on the basis of the sensual inputs, but on basis of this self-constructed image.

These inner “images” include “image objects” with manifold relationships to each other, to different kinds of contexts (e.g., functional, spatial or temporal), and to our knowledge basis. Introspectively it is clear that for each situation there is a multiscale representation of events, of spatial or temporal cognition. All the time we are switching between scales, focusing sometimes more on details, sometimes more on the larger context.

Modern brain research has verified the fact that in many cases the inner representation of cognition, experiences, or perceptive contents is object oriented. The representation of perceptive contents in an object oriented and abstract way is the basis of thinking and also of language.

Beginning with primitive, more “technical” processing such as lateral inhibition – a contrast-increasing interplay between the neurons directly behind the retina – a process of increasing abstraction and increasing incorporation of semantic knowledge starts. It goes from construction of cognitive objects over primitive assignment of meaning to more and more elaborated, detailed, affective, and also conscious functionality. These sequences are not stiff; they include loops of local confirmation and improvement. So-called intermodal processing, furthermore, allows integrating inputs from different senses concerning the same cognitive event. A fairly complex reconciliation supports a higher degree of accuracy concerning knowledge about the world and the actual situation around us.

Modern cognitive psychology furthermore has worked out the vital role of affects and emotions in human perception. Each cognitive event is evaluated with an affective connotation. This affect plays an important role concerning the way of further information processing and concerning memory request. In changing life situations, emotions are filters which guide the focus of attention concerning what of the overabundance of sensual, cognitive inputs is of current importance and what is not.

All the functions and rules in the different steps of this process can be called categories in the sense of the categories “a priori” of Immanuel Kant, the German philosopher. More than 200 years ago he described that each perception needs a category “a priori,” a functionality which already exists before the perceptive step and which allows meaningful extraction of information. But where are categories coming from? Evolutionary epistemology has taught us that categories are learned through an evolutionary process in living interaction with the environment. In a similar way to anatomical features adapting to the environment, categories are adapting to the need for meaningful information gathering and decision making in changing environments. Our categories are the product of the experiences of all of our ancestors as well as of our own personal experiences. Thus, they are partly transmitted genetically and partly they grow with each step in life.

Throughout millions of years evolution has created what we today call human perception, and what we experience consciously or subconsciously in daily life. It is clear that the capabilities of the human brain and especially the semantic capabilities and the interplay between representation of a scene and perception are of astonishing elaborateness and flexibility. They will still remain unmatched for a long time.

#### ... and eCognition?

Of course the process of digital image analysis in eCognition is very, very different and in no way as complex as human visual perception. The underlying processes differ in principal. However, on a meta-level some parallel aspects are visible.

As the brain does, eCognition processes image information in an object oriented way.

The result of image analysis with eCognition comes in the form of a hierarchical network of image objects with concrete attributes, concrete classification and different kinds of concrete relations to each other and to the knowledge basis. This corresponds to the approach of constructivism: the network is the actively constructed imagination of the software about the image, a structure which resulted from the confrontation of image data with rules of processing and a knowledge basis. Evaluation / classification is done on the basis of previously self-constructed object primitives. Image data itself are not changed.



More complex applications of eCognition typically go step by step from simple extraction of image object primitives to more and more abstracted and knowledge based procedures.

Image information is represented by the hierarchical network at different spatial resolutions simultaneously, allowing mutual confirmation and the analysis of mutual relations.

Procedures for local image analysis strategies depending on the classification of image objects are possible, similar to loops of local improvement. Local confirmation is already included in the sense that, for instance, the attributes of an object, the composition of sub-objects, the relations to its neighborhood, and also to its context must all fulfill the feature description of the same class.

Multisource data fusion supported by eCognition has parallels to multimodal information processing by the brain. Data of very different sources are synchronized and brought into a meaningful relationship.

The resulting network of image objects is a perfect starting point for further object oriented simulation. Is thinking not something like simulation in the brain?

And how does the system learn? Nearest neighbor training and classification is already an example for how easy categories can be built in eCognition. Beyond that, many efforts will be undertaken in order to develop further methods which support the user in the development of robust and transferable knowledge bases.

### Acknowledgements

We thank Gunther Jäger for his kind contributions describing classification accuracy measures.

## References

### Fuzzy systems

Bandemer, H., Gottwald, S.: Fuzzy Sets, Fuzzy Logic, Fuzzy Methods with Applications, Wiley Press, New York, 1995

Benz, U.: Supervised Fuzzy Analysis of Single- and Multichannel SAR Data. IEEE Transactions on Geoscience and Remote Sensing, Vol. 37, No. 2, S. 1023-1037, 1999

Bezdek, J. C., Pal, S. K.: Fuzzy Models For Pattern Recognition, Methods That Search for Structures in Data, IEEE Press, 1992

Civanlar, R., Trussell, H.: Constructing Membership Functions Using Statistical Data, Fuzzy Sets and Systems 18, 1986, p. 1-13

Curlander, J., Kober, W.: Rule Base System for Thematic Classification in Synthetic Aperture Radar Imagery, IEEE 1992, p. 854-856

Driankov, D., Hellendoorn, H., Reinfrank, M.: An Introduction to Fuzzy Control, Springer Verlag, Heidelberg, 1993

Fogler, R., Koch, M., Moya, M., Hush, D.: Feature Discovery on Segmented Objects in SAR Imagery Using Self-Organizing Neural Networks, Conf. proceedings 930445, p. 12, 1993.

Haverkamp, D., Tsatsoulis, C.: The Use of Expert Systems in combination with Active and Passive Microwave Data to Classify Sea Ice, IEEE 1992, p. 1625-1627

Jäger, G., Benz, U.: Measures of Classification Accuracy based on Fuzzy Similarity, to be printed in IEEE Transactions on Geoscience and Remote Sensing

Karayiannis, N. B.: Generalized fuzzy k-means algorithms and their application in image compression, Aerosense ,95, Orlando, pp. 206-217, April 1995

Kosko, B.: Neural Networks and Fuzzy Systems, a Dynamical Systems Approach to Machine Intelligence, Prentice Hall, Englewood Cliffs, NJ 07632, 1992

Kosko, B.: Neural Networks for Signal Processing, Prentice Hall, Englewood Cliffs, NJ 07632, 1992

Kruse, R., Gebhardt, J., Klawonn, F.: Fuzzy-Systeme, Teubner Verlag, Stuttgart, 1993

Pierce, E., Ulaby, F. T., Sarabandi, K., Dobson, M. C.: Knowledge-Based Classification of Polarimetric SAR Images, IEEE Transactions on Geoscience and Remote Sensing, Vol. 32, No. 5, pp. 1081-1086, September 1994

Tsatsoulis, C., Expert systems in Remote Sensing applications, IEEE Geoscience and Remote Sensing, Society Newsletter, June 1993, p. 7-15

#### **Accuracy assessment**

J. Cohen: A coefficient of agreement for nominal scales, Edu. Psychol. Meas., vol. 20, pp. 37-46, 1960

R.G. Congalton: A review of assessing the accuracy of classifications of remotely sensed data, Rem. Sens. Environ., vol. 37, pp. 35-46, 1991

U. Hellden: A test of landsat-2 imagery and digital data for thematic mapping illustrated by an environmental study in northern Kenya, Lund Univ. Nat. Geog. Inst., Lund, Sweden, Techn. Rep. 47, 1980

G.H. Rosenfield and K. Fitzpatrick-Lins: A coefficient of agreement as a measure of thematic classification accuracy, Photogramm. Eng. Remote Sensing, vol. 52, pp. 223-227, 1986

N.M. Short: The Landsat tutorial workbook – Basics of satellite remote sensing, NASA ref. pub. 1078, Greenbelt 1982

#### **Segmentation**

Baatz, M. & A. Schäpe: Object-Oriented and Multi-Scale Image Analysis in Semantic Networks. In: Proc. of the 2nd International Symposium on Operationalization of Remote Sensing August 16th-20th 1999. Enschede. ITC. 1999

Baatz, M. & A. Schäpe: Multiresolution segmentation – an optimization approach for high quality multi-scale image segmentation. In Strobl, Blaschke & Greisebener (Edts): Angewandte Geographische Informationsverarbeitung XI. Beiträge zum AGIT-Symposium Salzburg 1999. Karlsruhe. Herbert Wichmann Verlag. 2000

Mao, J & A. Jain: Texture classification and segmentation using multiresolution simultaneous autoregressive models. In: Pattern Recognition, Vol. 25, S. 173-188. 1992

Hofmann, T.; Puzicha, J. & J. Buhmann: Unsupervised texture segmentation in a deterministic annealing framework. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 20, Nr. 8, S. 803-818. 1998



# 5 FUNCTIONAL GUIDE

## How to use eCognition

This chapter shows you how to use eCognition in all its different functionalities in detail. Important steps for handling images are described, including time-tested strategies, and we highly recommend you to go through it carefully. Use the User Interface chapter along with this one, if you do not yet feel completely comfortable with eCognition's user interface.

<b>How to use eCognition</b> .....	<b>169</b>
<b>Overview</b> .....	<b>170</b>
<b>eCognition workflow</b> .....	<b>170</b>
<b>Importing and Exporting</b> .....	<b>172</b>
<b>Navigation and Visualization</b> .....	<b>189</b>
<b>Image Object Generation I: Multiresolution Segmentation</b> .....	<b>198</b>
<b>Working with Polygons</b> .....	<b>211</b>
<b>Information on Image Objects and Features</b> .....	<b>215</b>
<b>Classification Introduction</b> .....	<b>222</b>
<b>Classification Basics</b> .....	<b>230</b>
<b>Classification Advanced</b> .....	<b>253</b>
<b>Image Object Generation II: Classification-based Segmentation/Refinement</b> .....	<b>281</b>
<b>Accuracy Assessment and Statistics</b> .....	<b>291</b>
<b>Automation of Operations</b> .....	<b>299</b>
<b>Manual Editing of Image Objects</b> .....	<b>302</b>

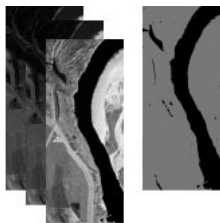
## Overview

The functional guide explains the different functionalities of eCognition in detail. The layout of the functional guide is meant to follow the workflow as it is encountered when analyzing an image with eCognition. Some of the tools are treated in more than one chapter because they are needed in more than one phase of the workflow.

We highly recommend that you go through the functional guide carefully while simultaneously consulting the user interface chapter along with this one, if you do not yet feel completely comfortable with eCognition's user interface.

## eCognition workflow

The workflow in eCognition follows a certain pattern. Prerequisite for a classification is the object hierarchy. The object hierarchy, whether it contains only one single level or a multitude of levels, provides the base for the classification. Then based on this object hierarchy the class hierarchy is developed. This development typically happens in iterative steps. A first set of classes is introduced and the image is classified. Then as additional classes are added and eventually class related features are used, the class hierarchy is improved and the classification result optimized.

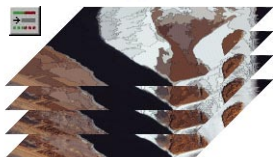


### Loading image and thematic data

The input data can have different resolutions and cover different areas, as long as geoinformation is available. The thematic layers can be ASCII raster or \*.shp vector format.

### Relevant chapters:

- 01 – Importing and Exporting
- 02 – Navigation and Visualization

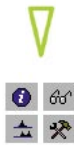


### Image object generation I:

#### Multiresolution Segmentation

Multiresolution segmentation extracts image objects, which are then classified. In many cases one level of image objects already satisfies the requirements for the classification. However, by iterative segmentation cycles an object hierarchy can be constructed which allows access to objects of different scales at the same time.

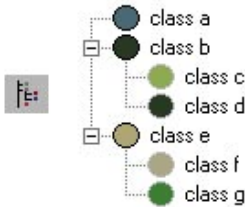


**Relevant chapters:**

- 03 – Image Object Generation I: Multiresolution Segmentation
- 04 – Working with Polygons

**Information on image objects and features**

eCognition offers several tools which help find useful features and ways to separate classes.

**Relevant chapter:**

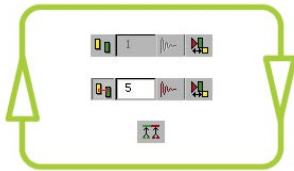
- 05 – Information on Image Objects and Features

**Creating a class hierarchy**

Based on the user's knowledge and aided by eCognition's tools, the knowledge base which is used to classify the image objects is constructed.

**Relevant chapters:**

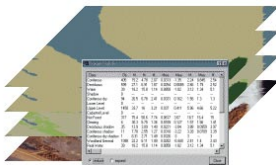
- 06 – Classification Introduction
- 07 – Classification Basics

**Classification & refinement**

In iterative steps the image is classified. Initial classification results can be refined by an extension of the class hierarchy and by restructuring the object hierarchy based on the first classification results.

**Relevant chapters:**

- 08 – Classification Advanced
- 09 – Information on Classification
- 10 – Image Object Generation II: Classification-based Segmentation/Refinement

**Export of information**

The classification results can be exported as graphic information in vector or raster format. Additionally statistical information can be created.

**Relevant chapters:**


- 11 – Accuracy Assessment and Statistics
- 01 – Importing and Exporting

## Importing and Exporting

This chapter is subdivided into the following parts:

- Creating new eCognition projects
- Adding additional layers to a running eCognition project
- Creating aliases for your layers
- Loading and saving eCognition projects
- Importing and exporting eCognition files
- Exporting objects, classification results and views

### Creating new eCognition projects

To import image or thematic layers open the “Import Image Layers...” dialog by either clicking the  icon or choosing “Project > New...” from the menu bar.

eCognition is capable of importing raster and vector data, whereby vector data is converted by eCognition into raster data to be imported. The software distinguishes two basic types of data:

- image layers
- thematic layers

While image layers contain continuous information, the information of thematic layers is discrete. The two types of layers have to be treated differently in both segmentation and classification. Thematic layers can be imported in addition to image layers.

Importing and exporting thematic layers also represents eCognition's interface to GIS.

**Note!** It is possible to import additional image and thematic layers into a running project (see below “Adding additional layers to a running eCognition project”).



## Importing image layers

### Supported image formats

By clicking “Insert” **2** you can insert image layers.

Using the GDAL library by Frank Warmerdam, a translator library for raster geospatial data formats and additional plug-in formats, eCognition supports the import of a variety of raster file formats including the most common ones.

Supported image formats:

- Arc/Info Binary Grid
- \*.asc ESRI ASCII GRID File
- \*.tif Tagged Image File (Geocoded)
- \*.ecw ER Mapper Compressed Raster
- \*.gif Compuserve GIF <sup>TM</sup>
- \*.bmp Windows or OS/2 Bitmap files
- \*.jpg JPEG JFIF
- \*.jp2 JPEG 2000
- \*.img Erdas Imagine Images
- \*.pix PCI DSK image format

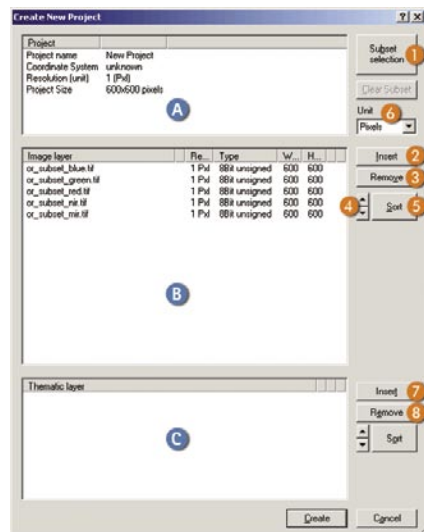
**Note!** If either PCI Geomatica© or Geomatica Geogateway© is installed on your computer, additional import formats are supported. The respective software only has to be installed, it is not necessary to start the software.

### Dialog “Create Project”

In the upper window **A** of the “Create Project” dialog you see the general project header information, including the following items:

- project name
- coordinate system
- resolution
- geocoding
- project size (or subset size, if defined)

The window in the middle of the dialog **B** displays the loaded image layers along with their properties.



The lower section of the dialog **C** displays thematic layers together with their attribute tables.

By clicking “Subset selection” **1** a subset can be defined (see next chapter “Select a subset”).

By clicking “Insert” (e.g., button **2** or **7**) or “Remove” (e.g., button **3** or **8**) you can insert new layers or remove selected layers from the project.

To sort layers, select the one or more layers that you want to sort and move them up or down by using the small arrows. **4**

To sort layers alphabetically, select the layers to be sorted and press the “Sort” button.

**5**

Select the desired units for your project with the drop-down menu „Unit“ **6**.

### Handling different resolutions

You can insert image or thematic layers with different resolutions into eCognition. They do not have to have the same number of columns and rows. To combine image layers of different resolutions, the images with the lower resolution “meaning larger pixel size” are resampled to the size of the smallest pixel size. If the layers have exactly the same extent and geographical position, then geocoding is not necessary for the resampling of images. If no measurement unit is given in the image data itself, the unit is automatically assigned to pixels. If you want to change the unit for your project you can assign it appropriately. You can choose from all commonly known sensible units - metric and imperial.

If no measurement unit is given in the image data itself, the unit is automatically assigned to pixels **6**. If you want to change the unit for your project you can assign it appropriately. You can choose from all commonly known sensible units - metric and imperial.



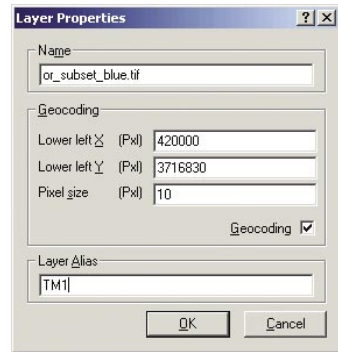
larger resolution - small pixel size



lower resolution - image is resampled to be imported into eCognition

### Geocoding and different geographical coverages

The geocoding information from the inserted files is automatically detected by the software. If the information is not included in the image file but is nevertheless available, you can open a dialog for each layer to insert the geocoding information by double-click or right-click on the appropriate layer. In the following dialog “Layer Properties” the name of the layer and the x and y coordinates of the lower left corner as well as the pixel size can be entered. Furthermore, you can decide whether you want to ignore the geocoding of a specified layer by activating the check box: No Geocoding. In this dialog you can also change the layer’s name and add an alias for the layer.



If you change the unit of your project and also the resolution, you will realize that also the coordinates of the project change accordingly.

See also [User Interface > Layer Properties](#).



If the loaded image files are georeferenced to one single coordinate system, image and thematic layers with a different geographical coverage and/or size (and resolution) can also be inserted, as displayed in the image below.

Images with different geographical location and size can be handled by eCognition if they belong to the same coordinate system.

### Select a subset

After inserting an image file, click the button “Subset selection” in the dialog “Create Project” to select a subset.

The following dialog comes up:



The subset can be determined by a rectangle drawn with the mouse or by inserting corner coordinate values. The corner coordinate values can be defined in the section „Subset“ ❶ in the fields “Minimum X/Y” (lower left corner) and “Maximum X/Y” (upper right corner). Assuming your data is georeferenced, activate checkbox ❷ “Use geocoding for subset” and the corner coordinates can be selected based on the respective geographical coordinate system. The field “Scene Size” ❸ shows the resulting subset extent based on the chosen subset rectangle, which is colored in red.

If you have inserted different layers the field “Active image layer,” ❹ allows you to visualize one image layer at a time. The location of the images that are not selected

as active image layer are indicated by a white rectangle if the image layers differ in their geographical location and/or size.

To save the subsets as separate files, check the checkbox “Store subset in own file(s)” ❺.

See also [User Interface > Subset Selection](#).

### Importing thematic layers

In principal all import formats supported for image layers are also supported for thematic layers; additionally the shape file format of the type polygon is supported.

To import a thematic layer press the “Insert” ❷ button in the lower section of the “Create Project” dialog, choose the thematic layer first and afterwards the attribute table from the menu that pops up. To insert additional thematic layers, repeat the procedure. The selection can be seen in window ❸.

Supported formats for thematic layers:

- Arc/Info Binary Grid
- \*.shp Shape files (shape type polygon)
- \*.asc ESRI ASCII GRID File
- \*.tif Tagged Image File (Geocoded)
- \*.ecw ER Mapper Compressed Raster
- \*.gif Compuserve GIF <sup>TM</sup>
- \*.bmp Windows or OS/2 Bitmap files
- \*.jpg JPEG JFIF
- \*.jp2 JPEG 2000
- \*.img Erdas Imagine Images
- \*.pix PCI DSK image format

**Note!** It is only possible to load shape files of the shape type polygon. Other shape file types like point or line are not supported.

Vector data needs to be rasterized for import into eCognition. A thematic raster \*.tif file is created and saved on the hard disc in the same folder as the shape file. For details see [Concepts and Methods > Vector format import and export](#). The folder where the shape file is stored cannot be a read-only file. Therefore, you cannot import a shape file from, e.g., a CD-ROM drive.

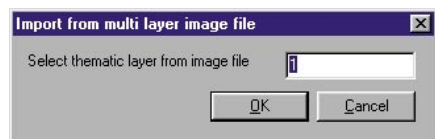
For polygon shape files the attribute table is selected automatically by choosing the appropriate \*.dbf file. For all other formats the respective attribute table must be specially indicated. Therefore the dialog “Load attribute table” comes up with the supported file formats:

- \*.txt ASCII text files
- \*.dbf Dbase files
- \*.csv Comma separated values

When loading a thematic layer from a multilayer image file (e.g., \*.img stack file), the appropriate layer that corresponds with the thematic information is requested in the dialog “Import from multilayer image” file: select thematic layer from image file. Additionally, the attribute table with the appropriate thematic information must be loaded.

To sort thematic layers proceed as described above in “Importing image layers.”

To finalize the import click “Create.”



**Note!** Thematic layers cannot have aliases.

## Adding additional layers to a running eCognition project

### Add image layer to an existing project

To add an image layer to a running eCognition project choose from the menu “Project > Add Image layer.”

The following image formats are supported (same as image layers):

- Arc/Info Binary Grid
- \*.asc ESRI ASCII GRID File
- \*.tif Tagged Image File (Geocoded)
- \*.ecw ER Mapper Compressed Raster
- \*.gif Compuserve GIF™
- \*.bmp Windows or OS/2 Bitmap files
- \*.jpg JPEG JFIF
- \*.jp2 JPEG 2000
- \*.img Erdas Imagine Images
- \*.pix PCI DSK image format

### Add thematic layer to an existing project

To add a thematic layer to a running eCognition project choose “Project > Add Thematic layer” from the menu.

Supported thematic file formats (same as thematic layers):

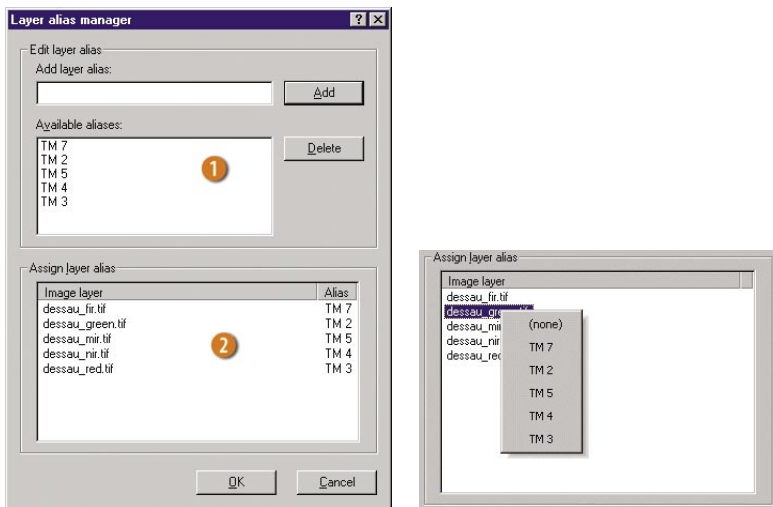
- Arc/Info Binary Grid
- \*.shp Shape files (shape type polygon)
- \*.asc ESRI ASCII GRID File
- \*.tif Tagged Image File (Geocoded)
- \*.ecw ER Mapper Compressed Raster
- \*.gif Compuserve GIF™
- \*.bmp Windows or OS/2 Bitmap files
- \*.jpg JPEG JFIF
- \*.jp2 JPEG 2000
- \*.img Erdas Imagine Images
- \*.pix PCI DSK image format

See also “Importing thematic layers,” especially section “Load attribute table.”

**Note!** If a thematic layer is not used for segmentation, it consequently is not possible to use the information given by this thematic layer for classification purposes.

## Assigning layer aliases

The basic function of layer aliases is to make the whole workflow in eCognition more transparent and independent from the initially used input data. Thus, when assigning and using layer aliases, all steps of your analysis with eCognition become more transferable, since all layer-sensitive operations and features can optionally refer to the layer aliases or to the layers themselves. Hence, protocols, class hierarchies and customized features become more independent from their situation of creation and are in consequence more flexible to use. To assign layer aliases you have two possibilities: either when creating the project (see above) or by selecting the appropriate dialog: „Project > Assign layer alias ...“.



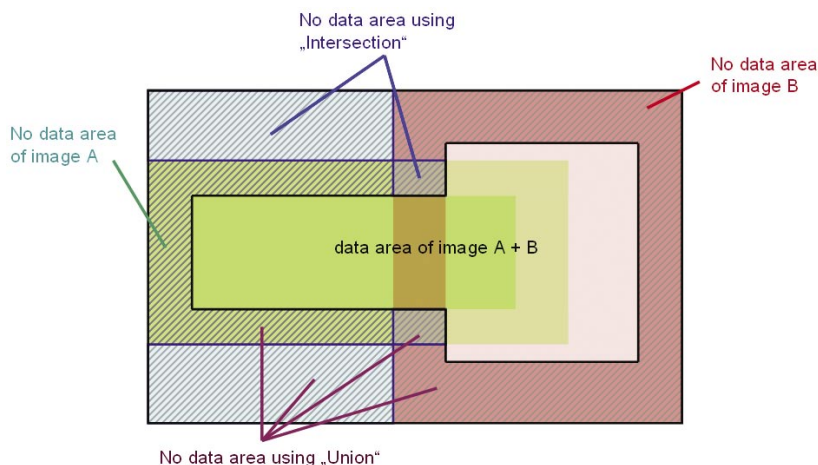
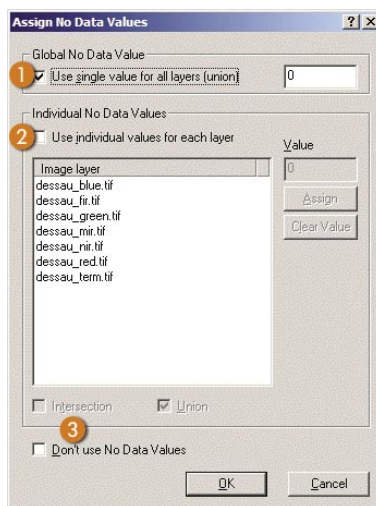
To enter and add an alias name choose the appropriate field **1** and button. “Delete” removes the selected alias name from the list. To assign an alias to a layer, right-click on the appropriate layer in the lower field **2** and select an alias name. Note! each alias can be assigned only to one layer and vice-versa.

## Defining no data values

In order to prevent eCognition from too much unnecessary preprocessing time, you can define image pixel values or combination of values, which shall be treated as no data. These areas will then not taken into account for any further analysis (segmentation, object statistics etc.). Especially georeferenced satellite images or mosaics usually have

such areas. To define the no data values choose from “Project > Assign No Data Value ...”. The following dialog comes up:

Check ❶ if you want to apply a single value for all used image layers. If you have a certain value combination (e.g. value 0 in the first layer and -255 in another layer) you can check ❷. Select the layer(s) of concern, enter for each layer a no data value and click “Assign”. For the case you have a project with overlapping images you can choose between “Intersection” and “Union” ❸. “Intersection” will only take into account those no data areas, all images have in common (overlapping no data areas). “Union” takes into account the no data areas of all individual layers for the whole project (see also the following figure). To delete a single assignment click “Clear Value”. Switching on “Don’t use No Data Values” will keep your assignments, but your project will be created without assigned no data values, i.e. the project will be created as usual.



**Note!** if there are any pixel within the data area which have the same gray value(s) as you have defined here, they will also be treated as no data.





## Loading and saving eCognition projects

When eCognition is started you can either load a saved project or create a new one. An eCognition project stores the loaded image information, view settings, classification results, and any other kind of operation undertaken so far. This enables you to save a current project in any status during the workflow process.

**Note!** As there is no undo button, it is recommended that you save a project prior to any operations that could lead to the unwanted loss of previous information, such as the deleting of object layers, classification-based fusion or new segmentation of existing layers.

eCognition project files have the extension \*.dpr. View settings are saved in a separate file with the extension \*.dps. If you want to move an eCognition project file and keep the view settings remember to move the settings file as well.

To load an eCognition Project, press  or choose “Project > Open...” from the menu bar and select the project you want to load. To save an eCognition Project, press  or choose “Project > Save” from the menu bar to save the project with its current name. To save the project under a new name, choose “Project > Save as...”

## Importing and exporting eCognition files

### Loading and saving class hierarchies

The class hierarchy is the knowledge base for classification defined by the user. To be able to use different classification schemes on one data set or the same classification scheme on different data sets, the class hierarchies can be saved. For the nearest neighbor classifier, the features to span the feature space together with their values are saved – not the sample objects. This enables you to transfer a class hierarchy working with the nearest neighbor classifier to different data sets. If you want to be able to use the sample objects of a saved class hierarchy in a different project, you first have to save the sample objects in a TTA mask.

eCognition's class hierarchy files have the extension \*.dkb.

To save a class hierarchy, choose “Classification > Save Class Hierarchy...” from the menu bar or right-click in the dialog “Class Hierarchy” and select “Save Class Hierarchy....”

To load a class hierarchy, choose “Classification > Load Class Hierarchy...” from the menu bar or right-click in the dialog “Class Hierarchy” and select “Load Class Hierarchy....”

### Loading and saving protocols

eCognition allows the recording of protocols. These protocols can be used to run a user-defined analysis on more than one project. When creating new classification applications it is also very useful to be able to reconstruct all operations performed during a project. See also “Automation of Operations.”

eCognition protocol files have the extension \*.dpt.

To load a protocol choose “Protocol > Load Protocol...” from the menu bar or right-click in the dialog “Protocol Editor” and choose “Load Protocol....”

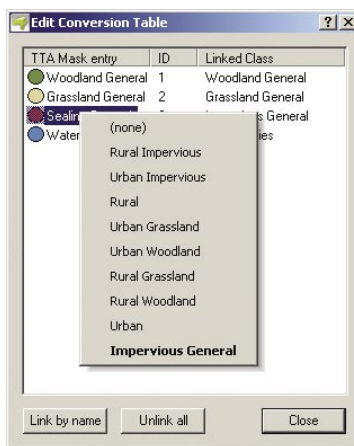
To save a protocol choose “Protocol > Save Protocol...” from the menu bar or right-click in the dialog “Protocol Editor” and choose “Stop Recording” and afterwards “Save Protocol....”

### Loading and saving TTA masks

In eCognition, you are able to load what is called a TTA mask. TTA stands for training and test area. A TTA mask can be used to create sample objects (training areas in a pixel-based software approach) for supervised classification using the nearest neighbor algorithm. It can also be used to define test areas for an accuracy assessment of the classification results.

When a TTA mask is loaded, you are asked whether you want to create classes from the conversion table of the TTA mask or not. This is necessitated by the two different applications of the TTA mask. If you want to use the mask for a nearest neighbor classification, it is reasonable to select yes, because then you do not have to define these classes. In this case, the samples are automatically linked to their classes.

If you select no, which is sensible when using the TTA mask for the evaluation of a classification, the mask classes are not linked automatically to the classes of your class hierarchy. This can be done manually afterwards if you choose “Samples > Edit Conversion Table...” from the menu bar.



See also [User Interface > Conversion Table](#).

To load a TTA mask choose “Samples > Load TTA Mask...” from the menu bar.

Supported formats to load a TTA mask file (same as thematic layers):

- Arc/Info Binary Grid
- \*.shp Shape files (shape type polygon)
- \*.asc ESRI ASCII GRID File
- \*.tif Tagged Image File (Geocoded)
- \*.ecw ER Mapper Compressed Raster
- \*.gif Compuserve GIF <sup>TM</sup>
- \*.bmp Windows or OS/2 Bitmap files
- \*.jpg JPEG JFIF
- \*.jp2 JPEG 2000
- \*.img Erdas Imagine Images
- \*.pix PCI DSK image format

To save a TTA Mask choose “Samples > Save TTA Mask...” from the menu bar.

Supported file formats:

- \*.asc ESRI ASCII GRID File
- \*.img Erdas Imagine Images
- \*.tif Tagged Image File (default)

Supported file formats (PCI Geomatica© or Geomatica Geogateway© is installed):

- \*.asc ESRI ASCII GRID File
- \*.img Erdas Imagine Images
- \*.pix PCI DSK image format
- \*.tif Tagged Image File

Supported formats for the attribute table see “Load attribute table.”

## Exporting objects, classification results and views

eCognition allows the export of results in three different ways.

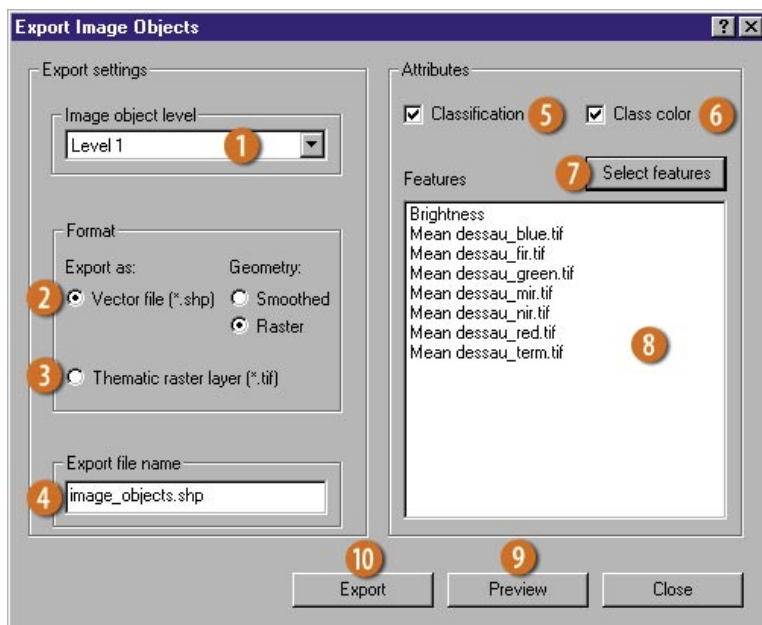
- Export image objects
- Export object shape
- Export classification
- Export current view

**Note!** The georeferencing information as provided when creating a project will be exported along with the classification results and additional information if you choose “Export image objects” or “Export classification.”

### Export image objects as thematic layer

Choose “Export > Image Objects...” from the menu bar to export image objects as a thematic layer.

In this case image objects can be exported together with their attributes and their classification. Each object has a unique object ID and the information is stored in an attached attribute table that is linked with the image layer. The georeferencing information as provided when creating a project will be exported as well.



In the “Image object level” drop-down menu select the image object level to export. <sup>1</sup> Then choose in the section “Format” whether you want to export an image object level as vector file <sup>2</sup> (\*.shp) or as a thematic raster layer <sup>3</sup> (\*.tif with attribute list \*.csv). For the geometry of the vector file, check either “Smoothed” or “Raster” to define the type of exported polygons. Raster geometry exactly follows the image pixels, whereas

smoothed geometry avoids single pixel steps. See also [Concepts & Methods > Vectorisation](#).

The default name for both file types is *ImageObjects* and can be modified in the field “Export file name” 4.

There are two possible locations for saving exported files:

- a) if a new project is created and not yet saved, the exported files are saved to the folder where the image data lies;
- b) if the project is saved (recommended), the exported files are saved in the folder where the project has been saved.

The directory where the exported files are saved cannot be chosen. The reason is that this would lead to conflicts in supporting the operation *export* in an eCognition protocol (See also [Functional Guide > Automation of Operations](#)).

The attributes “Classification” 5 and “Class color” 6 are added by default as an attribute to the export table but can also be deactivated. For “Classification” the class name, ID and membership value of the best, second best and third best classes are stated in the attribute table. For “Class color” three columns containing the RGB values are added to the attribute tables.

Click the button “Select features” 7 to add or remove features from the attribute list. The selected features are displayed in the left area of the window that pops up, the available features in the right area. They can be added or removed from one side to the other by double-clicking.

In the field “Features” 8 the selected features to be exported are listed.

A preview of the attribute table that is exported can be requested by choosing the “Preview” button 9.

Choose “Export” 10 to save the geometry and attribute information to the disk.

### Export object shapes

Choose „Export > Object Shapes...” from the menu bar to export polygons, lines or points of selected classes.

Also in this case image objects can be exported together with their attributes and classification. Each object has a unique object ID and the information is stored in an

attached attribute table, which is linked with the exported objects. The georeferencing information as provided when creating a project is exported as well.

There are two main differences to the “Export Image Objects” dialog:

- While in the “Export Image Objects” dialog it is only possible to export the whole classification of an image object level, here it is possible to export only the objects of single classes selected by the user.
- The second difference is that the export is not confined to the export of polygons based on the image objects. It is possible to select among three different shape formats. Points are the result of the calculation of the center of the main line for each image object. Lines are based on the main line of the skeleton computed for each

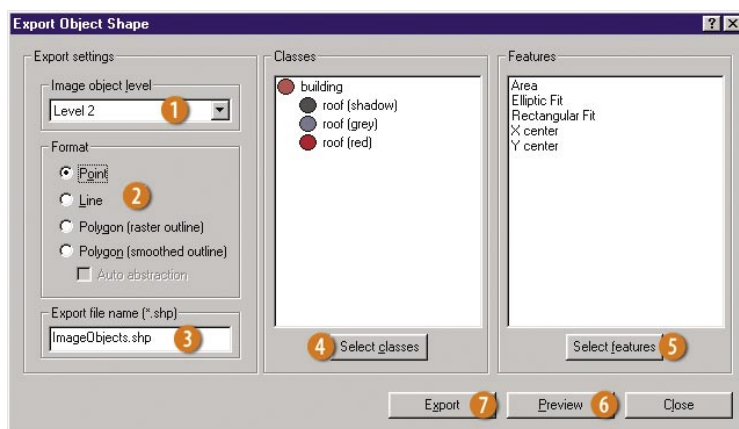


image object. Polygons describe the border of the image objects along the pixel raster (raster) or slightly abstracted border (smoothed).

- 1 Select the image object level where the classes to be exported are situated.
- 2 Choose the format, or whether to export polygons, points or lines. For the geometry of the polygons check either „Polygon (raster)“ or „Polygon (smoothed)“. As an additional feature you can choose for smoothed polygons an auto-abstraction. The purpose is a very high abstraction degree that should produce very smooth borders. For instance to get roofs with only four edges.
- 3 The default name *ImageObjects* could be changed here.
- 4 Use this button to add or remove classes to be exported. The selected classes are displayed in the right area of the window that pops up. The available classes are displayed in the left area. They can be added and removed by a click with the left

mouse button. With a right click it is possible to select a parent class including all child classes.

- 5 Use this button to add or remove features from the attribute list. The selected features are displayed in the right area of the window that pops up. The available features are displayed in the left area.
- 6 A preview of the attribute table that is exported can be requested by choosing the „Preview“ button.
- 7 Choose „Export“ to save the geometry and the attribute information to the disk.

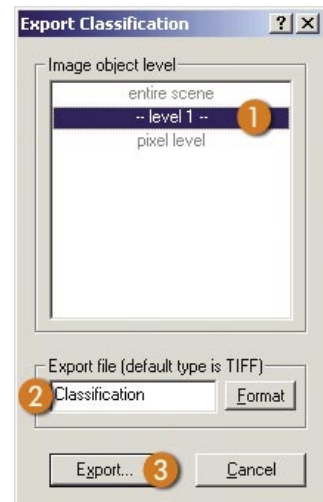
**Note!** The class-names or class IDs are not exported automatically. Thus, if you want to export shapes for more than one class and you want to distinguish the exported features by class, you should also export the feature “classified as”.

### Export classification

To export only the image classification, select “Export > Classification...” from the menu bar.

A labeled raster image with pixel values describing the class IDs is exported, as opposed to individual object IDs. The attached attribute table contains the class ID, color coding and class name by default. Furthermore, the georeferencing information as provided when creating a project will be exported as well. However, based on this export type adjacent image objects belonging to the same class cannot be distinguished anymore.

Select the image object level for export in the field “Image object level” ① and choose file name in the field “Export file” ② where the default name is *Classification*.



Supported file formats:

- \*.asc ESRI ASCII GRID File
- \*.img Erdas Imagine Images
- \*.tif Tagged Image File (default)

Supported file formats (PCI Geomatica© or Geomatica Geogateway© is installed):

- \*.asc ESRI ASCII GRID File


- \*.img Erdas Imagine Images
- \*.pix PCI DSK image format
- \*.tif Tagged Image File
- \*.bmp Windows or OS/2 Bitmap files

The attribute table is exported automatically as \*.csv file.

There are two possible locations for saving exported files:

- a) if a new project is created and not yet saved, the exported files are saved to the folder where the image data lies;
- b) if the project is saved (recommended), the exported files are saved in the folder where the project has been saved.

The directory where the exported files are saved cannot be chosen. The reason is that this would lead to conflicts in supporting the operation *export* in an eCognition protocol (See also [Functional Guide > Automation of Operations](#)).

Choose “Export”  to export the classification as a 16 bit image with attached attribute table.

### Export current view

To export your current view, choose “Export > Export View as File...” from the menu bar. This export type does not include additional information such as georeferencing, features or class assignments. Thus, exporting the current view is an easy way to create screenshots.

Supported file formats:

- \*.asc ESRI ASCII GRID File
- \*.img Erdas Imagine Images
- \*.tif Tagged Image File

Supported file formats (PCI Geomatica© or Geomatica Geogateway© is installed):

- \*.asc ESRI ASCII GRID File
- \*.img Erdas Imagine Images
- \*.pix PCI DSK image format
- \*.tif Tagged Image File
- \*.bmp Windows or OS/2 Bitmap files



## Navigation and Visualization

This chapter is divided into the following parts:

- Navigating within a scene
- Navigating within the image object hierarchy
- Navigating within the groups hierarchy
- Navigating between image layers
- View settings
- Edit highlight colors
- Multiwindow functionality


### Navigating within the scene

To navigate within a scene, you can use the different zoom and panning functions. The zoom function enlarges an image on the display. When an image is zoomed, it can be panned (scrolled) using the pan window (hint: panning or scrolling may also be called roaming in other software packages). Panning and zooming have no effect on the image files. To access the different cursors necessary for panning, zooming or selecting image objects, either right-click the view and select the respective cursor from the pop-up menu, or select the appropriate cursor from the „Zoom Functions“ toolbar.

### Zooming


For zooming there are four possibilities:

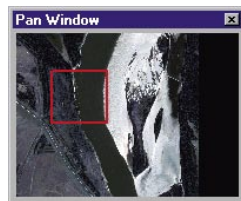
<b>Zoom In Center</b>	allows you to zoom in to the center of the view - default key is „+“ (not on numerical keys)
<b>Zoom Out Center</b>	allows you to zoom out from the center of the view - default key is „-“ (not on numerical keys)
<b>Zoom in</b>	allows you to zoom in to one point
<b>Zoom out</b>	allows you to zoom out
<b>Area zoom</b>	zooms to a defined area. Define the rectangle while holding down the left mouse button.
<b>Zoom 100 %</b>	displays the image so that one image pixel represents the size of one screen pixel.


The function “Zoom Scene to Window” is used to adjust the view to the window size. It can be applied either by using the  button or by choosing “View > Zoom Scene to Window” in the menu bar.

**Note!** Zooming works in steps of 100%. As a consequence the difference between the 100% and the 200% step might sometimes seem a bit large. However, this was done to increase performance, which was considered more crucial than smooth zooming.

## Panning

For panning, eCognition uses the pan window and the panning cursor. The panning cursor  allows you to pan a view by clicking and dragging the view. The cursor can be changed to the panning cursor by selecting it from the popup menu or the zoom toolbar or by selecting “View > Cursor mode > Panning...” from the menu.



The pan window is used to easily navigate in data sets. It always displays an overview of the whole scene while showing the outlines of the selection. Move the selection by clicking and dragging it. To open the pan window use the respective button  or open it in the “Toolbars & Dialogs” menu.

**Note!** To facilitate navigation, use the multiwindow function. By using different views with different settings which are frequently used, the time normally needed for changing the settings can be saved.



## Linking windows

If more than one window is opened, the zooming and panning functionalities can be linked. This way changes in the zoom factor in one window will be automatically performed in the linked window as well. The same is applied to the panning. To link windows, either select “Window > Link all Windows” or “Window > Link active Window” from the menu. When “Link active Window” is selected, the cursor changes to “Link.” Click on the window which you want to link with the active window and the link is performed.

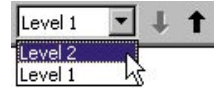
To unlink windows either select “Unlink all Windows” or “Unlink active Window” from the menu. The first selection will remove all links. The second one removes only the links of the active window.

## Navigating within the image object hierarchy

eCognition allows the construction of a hierarchical network of image objects of different spatial resolutions. It is often necessary to navigate through the different levels of



the image object hierarchy in order to be able to maintain information on the different levels and to perform classifications. You can either switch through the levels one by one, using the   buttons from the tool bar, or you can directly select the desired level in the drop-down menu of the toolbar.

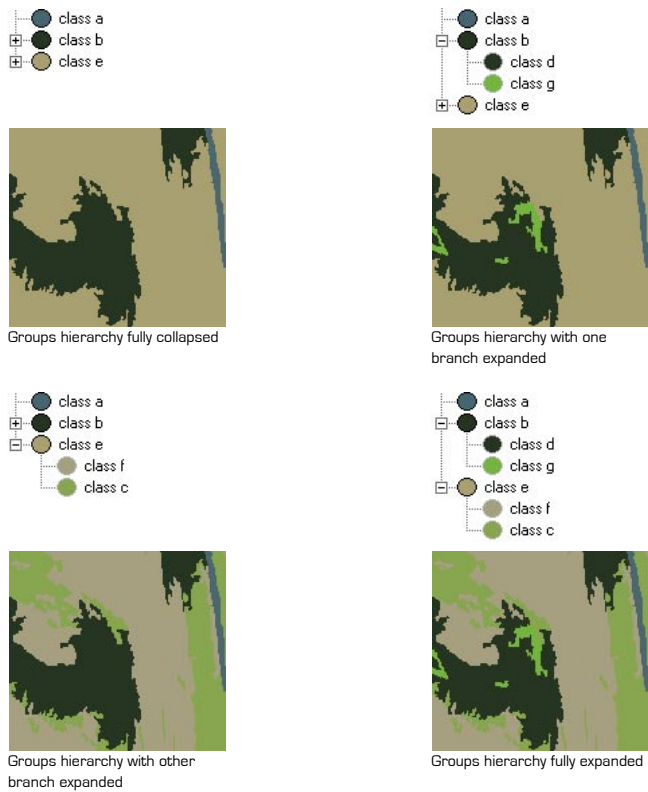
The current level of the image object hierarchy is displayed in the status bar as the first figure in the level field. The second figure, separated by a slash, represents the number of levels contained in the Image Object Hierarchy.




### Navigating within the groups hierarchy

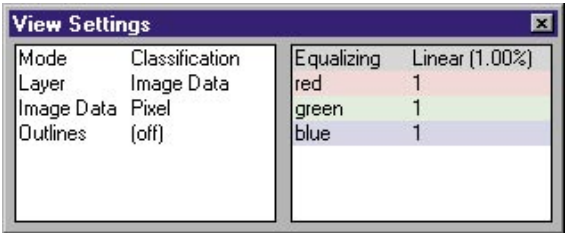
The groups hierarchy is used to organize classes in eCognition. Classes can be grouped as to their semantic relationships. Each reference which is directed to a group is directed automatically to all classes that are part of this group. Note that a class can be part of more than one group.

To be able to visualize these groupings, eCognition allows the display of image objects according to their group memberships. In the lowest level of the groups hierarchy, the image objects are displayed in the color of the class they are assigned to. When moving up in the groups hierarchy, the image objects are displayed in the color of the next higher class. In the “Parent Class for Display” window of the class description, you can determine whether the objects of one class are to be displayed in the color of their next higher class or whether they are to maintain the color of the class they are assigned to. To navigate within the groups hierarchy, move up or down using the   buttons from the tool bar. In addition, single branches of the groups hierarchy can be visualized separately by opening or closing them.

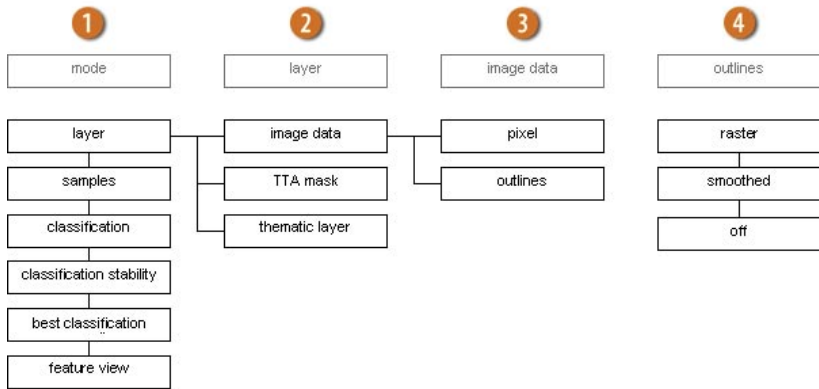


View settings

In the view settings the display features of image objects or image layers are determined. There are two main interfaces for adjusting view settings: The tool bar and the “View Settings” menu. To open the “View Settings” dialog, choose “View > Settings...” from the menu bar or click the  button from the tool bar.



In the “View Settings” dialog, several parameters can be adjusted:



### Mode 1

In the “Mode” box, several modes in which the object fill colors are presented can be chosen. Click over “Mode” to access the popup menu. Alternatively, use the buttons as indicated in the following list.

#### Layer



In the “Layer” mode, the objects are displayed without fill color. The layer information itself (the background, so to speak) is visible. The kind of layer information displayed is selected in the “Layer” settings. To apply this mode choose “Mode > Layer” or the respective icon in the toolbar. When using this setting, the increasing level of abstraction with increasing object size can be easily viewed.

#### Samples



In the “Samples” mode, the sample objects are displayed in their class color; the remaining objects are colored according to the “Layer” settings. To apply this mode, choose “Mode > Samples” or click in the tool bar.

#### Classification



To view the classification result, use the “Classification” mode or use the button from the tool bar. Unclassified objects are displayed according to the “Layer” settings or the settings of the “Highlight Color.”

#### Classification Stability

In the “Classification Stability” mode, the objects are colored according to the difference in membership values

of the best and second best class. The color ranges from green to red. The higher the difference in the two membership values, the greener the color. Small differences are colored red. See [Accuracy Assessment and Image Statistics](#).

**Best Classification Result**

In the “Best Classification Result” mode, the objects are colored according to their membership value. The color ranges from green to red. The higher the membership value, the greener the color. Low membership values are colored red. See [Accuracy Assessment and Image Statistics](#).

**Classification Membership**

To view the membership values of the final classification

**Feature View**



In addition to the mentioned view modes, the last five views applied during a project with the tool “Feature View” are stored and can be reapplied using the view settings.

**Layer 2**

The “Layer” settings determine what underlying information is displayed. Depending on availability, the “Layer” menu allows switching between a display of either the image data, a TTA mask, or thematic information. The default display shows the image data. To change the display, click on “Layer” and select among the available data sets.

**Image data 3**

When the “Layer” mode is set to “Image Data,” these settings can be used to switch between a display of the image pixels or a fill color, calculated from the mean value of the pixels of the object. When the pixel mode is selected, the object as such is only visible when selected. Use this mode in combination with the image outlines to get an overview of the quality of the segmentation results.

**Outlines / Polygons 4**

After polygons have been created, you are able to display them. There are four modes for the „Polygon“ settings:




**„off“  
raster**

In the “off” mode the polygon display is deactivated. The “raster” setting displays the exact border of two objects, following the raster space.

**smoothed**

The “smoothed” mode displays a generalized view of the borderline.

**Scale Parameter Analysis** “Scale Parameter Analysis” visualizes the results of the corresponding analysis. The polygons are colored according to the calculated results of the analysis.

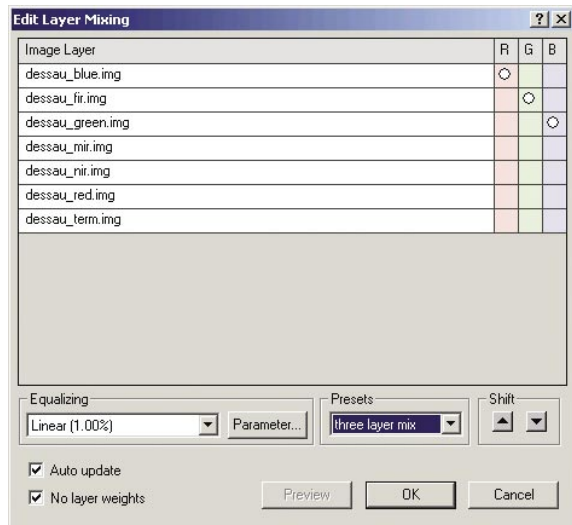
The polygon color can be changed in the „Edit Highlight Colors“ settings. To show or hide polygons you can also use the  button instead of the “View Settings” dialog. Since the polygons represent the borderline between two objects they cannot be displayed in a class color. To view outlines colored in the color of the classification, polygons have to be switched to “(off)” and the buttons  and  have to be active.

**Note!** Note that you have to create polygons before you are able to display polygons or skeletons and use them for features based on polygons/skeletons. By contrast, outlines can be displayed after the first segmentation without the creation of polygons.

### Layer mixing



The layer mixing dialog allows changes in the display of the image data. A multitude of different visualizations can be displayed with the help of this dialog. The data is not altered by changes in the layer mixing.

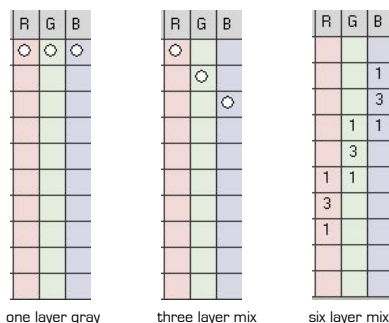
In the upper part of the dialog, the assignment of the image channels to the primary colors can be set individually. To assign a layer to a color, simply click the color the layer is to be displayed with. One layer can be displayed in several colors, and several layers can be displayed in the same color.



By default each layer is weighted similarly for each color. However, if a display with individual weights is desired, the “No channel weights” box can be deactivated. In this mode one layer can be given an individual weight for each color. To increase the

weight, left-click the color in which you want the layer to be displayed. To decrease the weight, right-click the color.

There are several predefined settings provided by eCognition. Depending on the number of available layers, a one layer mix, a three layer mix and a six layer mix are available. The one and three layer mixings can also be gathered by the  button and  button from the toolbar.



The „Shift“ function allows you to shift a defined layer mix up or down through the layer sequence.

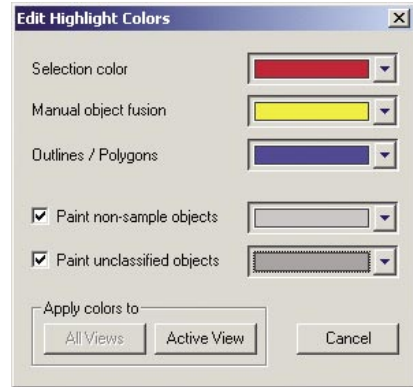
By default, any change performed in the layer mixing is automatically updated in the view. To deactivate the automatic update of the changes made in the layer mixing, deactivate the “Auto update” checkbox.

The “Equalizing” selection is used to apply an equalization stretch to the view. You can choose between linear equalization, standard deviation equalization and histogram equalization. For the linear equalization and the standard deviation equalization, the saturation value can be edited. The default saturation value is 1%. To edit this value you click the “Parameter ...” button. All stretches are only performed in the display device, i.e., the data file values are not changed.



## Edit highlight color

In the “Edit Highlight Colors” dialog, the highlight colors for image objects can be edited for different modes.



### Selection color

The most commonly used highlight color is the selection color which is used to display a selected object. It can be changed in the “Selection color” field.

### Manual object fusion

When the input mode is set to “Manual Fusion,” a different highlight color is used. This highlight color can be altered in the highlight color for “Manual object fusion.”

### Outlines/Polygons

The outlines and polygon color can be edited in “Polygons.”

### Paint nonsample objects

When the view mode is set to “Samples,” nonsample objects are displayed according to the “Layer” settings. To generally paint nonsample objects in a uniform color, activate the “Paint non-sample objects” checkbox and select a color.

### Paint unclassified objects

When the view mode is set to “Classification,” unclassified objects are displayed according to the “Layer” settings. To generally paint unclassified objects in a uniform color, activate the “Draw unclassified objects” checkbox and select a color.

## Multiwindow functionality


eCognition allows you to use several windows at the same time. To activate a new window, choose “Window > New Window” from the menu bar. Each window can have its own view and pan area settings. If an object is highlighted in one window, the shape of this object is highlighted in all other windows, regardless of the respective image object level. In windows that are set to pixel level, the selected object is not highlighted. See also “Linking windows” above.

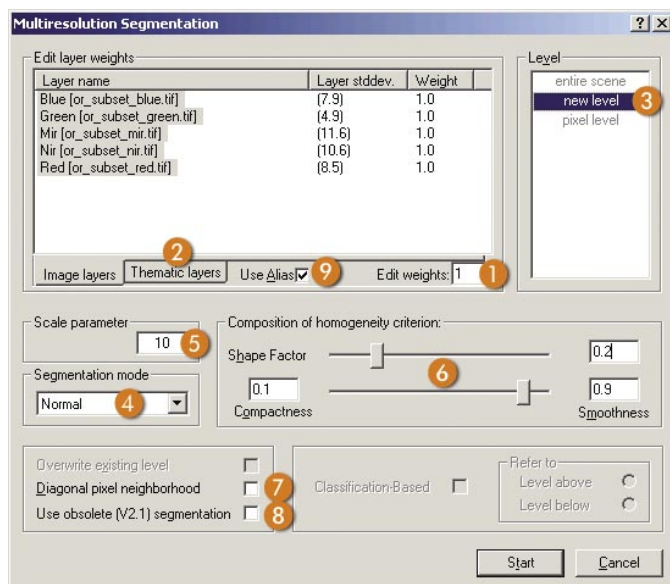
## Image Object Generation I: Multiresolution Segmentation

A basic procedure of eCognition is multiresolution segmentation for largely knowledge-free extraction of image object primitives. It was developed to produce image objects of different resolution and high quality. Multiresolution segmentation is essentially a heuristic optimization procedure, which locally minimizes the average heterogeneity of image objects for a given resolution over the whole scene.

### Multiresolution segmentation

Multiresolution segmentation is a method of generating image objects. It produces highly homogeneous segments in any chosen resolution, fitting your purpose. The resulting image segmentation can be universally applied to almost all data types. It is especially suited for high resolution data or highly textured data (e.g., radar). For a detailed description of the segmentation process see [Concepts & Methods > Multiresolution segmentation](#).

To open the “Multiresolution Segmentation” dialog either click  or choose “Segmentation > Multiresolution Segmentation...” from the menu bar.




To segment an image, several parameters have to be set:

- use alias
- layer weights
- image object level
- scale parameter
- segmentation mode
- composition of the homogeneity criterion
- type of neighborhood

The figure above shows the segmentation dialog in which the different parameters can be set. The displayed parameters are the default settings.

### Use Aliases

To make the segmentation process more flexible in order to re-use the segmentation method in other projects (e.g., when creating a protocol) you should use the layers' aliases instead of the image layers themselves. To switch on this mode click the appropriate radio button . By using aliases, the segmentation process becomes independent of the number of layers and their sequence because it then refers to the layers with the appropriate alias names. When re-using the segmentation process, only those image layers are taken into account in the new project which have layer aliases identical to the ones in the old project. See also Functional Guide > Automation of Operations.

**Note!:** For re-using purposes, in the new project at least the same number of image layers must be present as in the project wherein the segmentation was created.

### Layer weights

#### Image layers

Image layers can be assessed differently depending on their importance or suitability for the segmentation result. The higher the weight which is assigned to a layer the more of its information will be used during the segmentation process. Consequently, image layers that do not contain the information intended for representation by the image objects should be given little or no weight. When segmenting a LANDSAT scene, for example, the segmentation weight for the spatially coarser thermal layer should be set to 0 in order to avoid deterioration of the segmentation result by the blurred transient between image objects of this layer.

**Note!** Note that the sum of all chosen weights for image layers is internally normalized to 1.

The respective weight is displayed to the right of the layer name. To define the weight to be applied to the different image layers, select the layers you want to change by highlighting them and set the weight in the input field “Select and edit weights” ①. Along with the layer weight, the standard deviations of the layer values for each single layer over the entire scene are displayed to provide information about the layer dynamics.

### Thematic layers

Besides image layers, you can also use thematic layers for the segmentation. In contrast to image layers thematic layers contain discrete information. This means that related layer values can carry different information, which itself is defined in the attached attribute list. To be able to clearly define the affiliation of an object to a thematic class given by the thematic layer, it is not possible to create image objects which belong to different thematic classes. To ensure this, the borders separating different thematic classes are restrictive for further segmentation whenever a thematic layer is used. For this reason, thematic layers cannot be given different weights, but can merely be selected for use or not.

If a thematic layer is not used for segmentation, it consequently is not possible to use the information given by this thematic layer for classification purposes.

An assigned weight higher 1 means that the thematic layer is used; if the weight is set to 0 it is not used. Switch between the image layer and thematic layer view by clicking the register tab ②.

If you want to produce image objects based exclusively on thematic layer information, you have to switch the weights for all image layers to 0. You can segment an image also using more than one thematic layer. The results are image objects representing proper intersections between the layers.

### Level

In the “Level” box ③ you can choose the image object level which is to be created by the segmentation. You can create new levels above or below existing levels, in between two existing levels or you can overwrite levels. See also below “Constructing an image object hierarchy with more than one level.”

Since image objects are arranged hierarchically, it is not possible to create a level containing objects which are larger (i.e., which use a larger scale parameter) than objects on superior levels. Consequently, it is also not possible to build a level containing objects smaller than its sub-objects.

## Segmentation mode

In the „Segmentation mode“ control **4** you have a choice between “Normal”, “Sub obj. line analysis” and “Spectral Difference”

The normal mode is the default segmentation mode used for multiresolution segmentation, applicable in most cases. For this segmentation mode the scale parameter can have arbitrary values.

The mode for sub-object line analysis is used to create sub-objects for the calculation of line features based on sub-objects. For the creation of respective sub-objects, the shape of their super-objects is taken into consideration. For a detailed description see the respective section in “Concepts & Methods.” The scale parameter used in this mode ranges from 0.5 to 1. It determines the maximum border length of the resulting sub-objects relative to the border of their super-objects.

The mode for spectral difference segmentation is used to merge neighboring objects according to their difference in color. When in this mode, the value for “Scale Parameter” is used as the minimum spectral difference between the objects. If the difference is below this value, neighboring objects are merged.

## Scale parameter

The scale parameter is an abstract term which determines the maximum allowed heterogeneity for the resulting image objects. In heterogeneous data the resulting objects for a given scale parameter are smaller than in more homogeneous data. By modifying the value in the “Scale parameter” control **5** you can vary the size of image objects.

For the scale parameter of the segmentation modes „Sub obj. line analysis“ and „Spectral difference“ see above.

## Composition of the homogeneity criterion

The object homogeneity to which the scale parameter refers is defined in the “Composition of homogeneity criterion:” field **6**. In this context, homogeneity is used as a synonym for minimized heterogeneity. eCognition internally computes three criteria: Color, smoothness and compactness. These three criteria for heterogeneity can be applied in a mixed form. For most cases the color criterion is the most important one to create meaningful objects. However, a certain degree of shape homogeneity often improves the quality of object extraction. This is due to the fact that a certain compact-

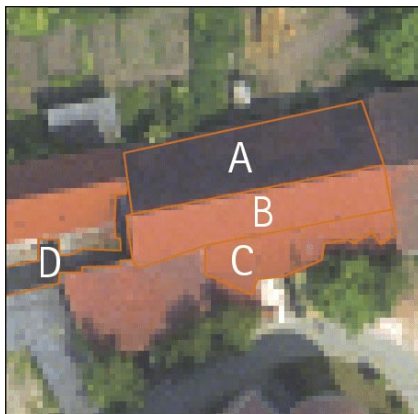
ness belongs to the conception of spatial objects. The shape criterion especially helps to avoid a fractal shaping of objects in strongly textured data (e.g., radar data).

In the “Color” field you can define to which percentage the spectral values of the image layers contribute to the entire homogeneity criterion, as opposed to the percentage of the shape homogeneity which is defined in the “Shape” field. Changing the weight for the color criterion to 1 will result in objects more optimized for spatial homogeneity. The color criterion cannot have a value less than 0.1, due to the obvious fact that without the spectral information of the image, the resulting objects would not be related to the spectral information at all.

In addition to spectral information, eCognition allows you to optimize object homogeneity with regard to the object shape. The shape criterion is composed of two parameters:

**Smoothness:** The smoothness criterion is used to optimize image objects with regard to smooth borders. To give an example, the smoothness criterion should be used when working on very heterogeneous data, such as radar images, to inhibit the objects from having frayed borders, while maintaining the ability to produce noncompact objects.

**Compactness:** The compactness criterion is used to optimize image objects with regard to compactness. This criterion should be used when different image objects which are rather compact, but separated from noncompact objects only by a relatively weak contrast, are to be extracted. This could be the case when segmenting urban areas, where rooftops (A/B), and adjacent roads or other structures (C/D) are of similar spectral values but of very different shape.



**Note!** It is important to notice that the two shape criteria are not antagonistic. This means that an object optimized for compactness can very well have smooth borders. Which criterion to favor depends on the actual task.


## Type of neighborhood

To use the diagonal pixel neighborhood mode as described in the chapter “Concepts & Methods,” check the respective box **7**. Diagonal pixel neighborhood should only be used if the structures of interest are of a scale near to the pixel size. A typical example


is the extraction of roads from coarse resolution data like LANDSAT TM. In all other cases, plane pixel neighborhood is the appropriate choice.

The decision of whether or not to use the diagonal pixel neighborhood mode has to be made before the first segmentation. If the first segmentation has been made with diagonal pixel neighborhood, all further segmentations have to use diagonal pixel neighborhood as well.

### Use obsolete (V2.1) segmentation

With release 3 a new segmentation method was introduced, which generates transferable segmentation results. In consequence the shape of objects created with this segmentation method differs slightly from that of objects created with eCognition 2.1 or lower. To keep class hierarchies, customized features and protocols created with old versions compatible, you can segment the image using the old segmentation method by checking this box 

### Deleting segmented levels

To delete segmented image object levels, select the menu item „Image Objects > Delete Level...“ or click the  button in the tool bar. Mark the pertinent levels to be deleted in the following dialog and click OK.

### How to produce image objects suited to your classification purpose

In order to be able to produce a satisfying classification result, the image objects have to be homogeneous with regard to the classes that have to be distinguished in the following classification. Therefore, avoid merging areas that can be assigned to different classes into one image object. Since it is often not possible to produce an image object level in which all image objects explicitly represent the classes to be extracted, it is recommended to use different object levels for the classification of structures of different scale.

The main principle for the segmentation is:

#### 1. Rule of thumb:

Always produce image objects of the biggest possible scale which still distinguishes different image regions (as large as possible and as fine as necessary). There is a tolerance concerning the scale of the image objects representing an area of a consistent classifica-

tion due to the equalization achieved by the classification. The separation of different regions is more important than the scale of image objects.

There is a further recommendation for the composition of the homogeneity criterion:

## **2. Rule of thumb:**

Use as much color criterion as possible while keeping the shape criterion as high as necessary to produce image objects of the best border smoothness and compactness. The reason for this rule is that a high degree of shape criterion works at the cost of spectral homogeneity. However, the spectral information is, at the end, the primary information contained in image data. Using too much shape criterion can therefore reduce the quality of segmentation results.

A well-suited approach when segmenting new image data is to simply run different segmentations with different parameters until the result is satisfying. If the data sets are too large for easy handling try to work with a representative subset to speed up the process. Once suitable segmentation parameters have been found, they can be applied to the whole data set.

## **Influence of bit depth of raster data**

Note that with increasing radiometric resolution (bit depth) of your image data the spectral similarity between adjacent pixels decreases (e.g., it is more probable to find two adjacent pixels with identical spectral values in 8 bit data than in 11 bit data). Consequently, you will have to choose a larger scale parameter for generating image objects on data with high radiometric resolution.

## **Working with image layers of different radiometric resolution**

If your project contains image layers of different bit depth, information from image layers with higher bit depth will have more influence on the segmentation result, since one scale parameter is applied to all image layers (see above). To equally provide information from image layers with higher bit depth, different weights have to be assigned to the image layers for the segmentation process: image layers with the lower radiometric resolution have to get higher weights.

## **Constructing an image object hierarchy with more than one level**

eCognition allows you to insert new image object levels both above and beneath existing ones. Since eCognition uses a pairwise merging algorithm, take into account that every segmentation uses the image objects of the next lower image object level as building



blocks which are subsequently merged to new segments. At the same time, the object borders of the next higher level are stringently obeyed. For this reason, it is not possible to build a level containing objects larger (i.e., using a larger scale parameter) than its super-objects. Consequently, it is also not possible to build a level containing objects smaller than its sub-objects.

**Note!** When creating the first image object level, the lower limit is represented by the pixels, the upper limit by the scene size.

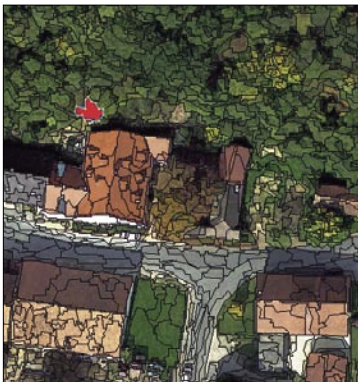
Given this information you can start creating an image object hierarchy. Segment the image in a way that you get image objects of different scales, which each represent image object primitives of your objects of interest.

### Classifying differently scaled objects

Image information is scale-dependent. It is not always possible to extract image objects with only one segmentation run that fits all classes of the classification scheme. An example would be when working on high resolution imagery with the task of classifying rooftop and forested areas at the same time:

In a first segmentation, rather small objects have been produced, which would be suitable for the classification of forested areas or treetops, respectively.

In another, coarser segmentation, objects better suited for the classification of the rooftops are shown. As you can observe, the forested areas are represented by a few objects which cover several treetops.



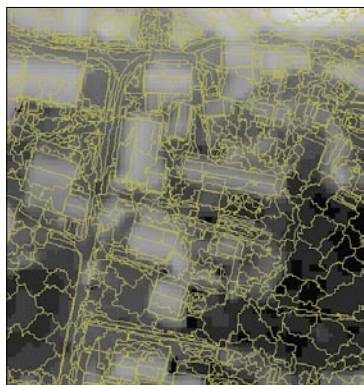
An advisable approach to solving this problem is to classify treetops on the finer resolved image object level and rooftops on the coarser one. In a subsequent refinement procedure, both image object scales can be projected onto one single image object level (see [Functional Guide > Image object generation II: Classification-based segmentation/refinement](#)).

### Using different data for segmentation and classification

When working with image data of different resolution combined in one project, it is recommended that the image objects be created based on the higher resolution image data, while using the coarser one merely as additional information for the classification.




In this example, a high-resolution aerial photograph was used for segmentation. Image objects quite well suited for the separation of houses, roads and types of land cover were created. As the separation of rooftops and roads always poses a problem in shadowed areas, a surface model was used to provide additional information.





As you can observe, the image information given by the surface model is not suited to the extraction of sensible image objects. The object borders displayed in this window are the same as in the window above, which were built using only the high-resolution spectral information. Nevertheless, the information contained in this image can be well used for discriminating elevated rooftops from low lying roads.

## Generating sub-objects for line analysis

eCognition delivers a powerful instrument for line analysis based on the analysis of appropriate sub-objects as described in “Concepts & Methods.” Those sub-objects have to be generated using the “Sub-objects line analysis” mode .

The difference between a sub-level built by using the multiresolution segmentation mode and one using the sub-objects line analysis mode is that the latter is especially implemented to produce objects suited to this feature. It takes the borders of the super-objects into consideration, producing compact sub-objects. Line analysis depending on sub-objects could be used with a conventionally created sub-level as well, but the results would be less satisfactory.


To generate a sub-level using the “Sub-objects line analysis” mode, choose the respective mode in the “Segmentation mode” control  and define the scale parameter . The scale parameter ranges from 0.5 to 1 (default setting: 0.75) for this kind of segmentation, describing the maximum relative border length of the sub-objects to the border of their super-object.

## Classification based multiresolution segmentation

You can perform selective multiresolution segmentation on already classified objects. By activating this mode, only objects within one structure group are affected by the segmentation parameters. Read more about classification based segmentation in Image Object Generation II > Classification-based Segmentation/Refinement.

## Scale parameter analysis

Obtaining optimal objects from multiresolution segmentation in most cases depends on a suitable choice of scale parameter. By analyzing the spectral relationships between neighboring objects, a potential scale parameter can be calculated which would lead to a merge of these objects, assuming that all other segmentation parameters do not change. Since the main aim of each segmentation is to create local homogeneity and to keep global heterogeneity, the statistics of all potential scale parameters can give a hint for a well suited scale parameter. To view the potential scale parameters for all neighboring objects you first have to calculate them.

Choose from “Segmentation > Scale Parameter Analysis ...” or click the  button. The following dialog will appear, which looks very similar to the “Multiresolution Segmentation” dialog:

**Scale Parameter Analysis**

Layer weights:

Layer name	Layer stddev.	Weight
or_subset_blue.tif	(7.9)	1.0
or_subset_green.tif	(4.9)	1.0
or_subset_mir.tif	(11.6)	1.0
or_subset_nir.tif	(10.6)	1.0
or_subset_red.tif	(8.5)	1.0

Select and edit weights: 1

Display settings:

Edit line width: 3

Minimum value: 0

Maximum value: 134

only applicable if polygons are available for selected level

Apply Reset

Level: Level 1

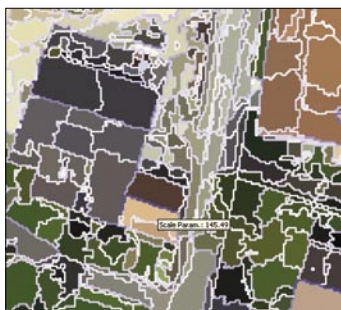
Segmentation mode: Normal

Composition of homogeneity criterion:

Criterion: Color 0.8, Shape 0.2, Smoothness 0.9, Compactness 0.1

Calculate Statistics Cancel

Press the „Calculate“ button to generate or update all potential scale parameters. You will realize that the color of the polygons and their size changes. By moving the mouse over one of the object borders you will see the potential scale parameter for this pair of objects in a tool tip:

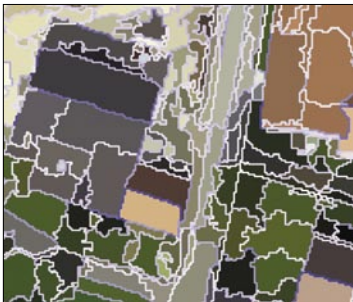
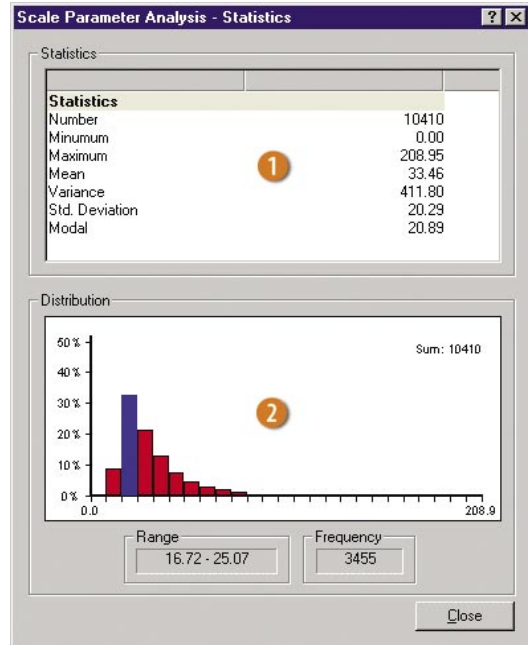


You can change the line width of object borders by adjusting it in the dialog ①. By default all possible values will be displayed, but you can also change the display range accordingly ②. To apply these adjustments immediately, press “Apply.” To select which segmentation level you want to analyze, make your choice in the “Level” box ③. All other parameters are analogous to the Multiresolution Segmentation dialog. Note! as soon as you change one of them, you always have to press “Calculate” ④ again.

To see the statistics over all potential scale parameters click the “Statistic” button ⑤.

In the upper part you get statistical information about the distribution of the potential scale parameters ①. In the lower part a histogram is shown wherein you can click on

each slot **2**. The respective slot will be marked in blue and you will get information about the range of the slot and the frequency of occurrences within this slot. For your statistical analysis the modal value is most important: In this example 30% of all neighborhoods have a value around 20.89 or between 16.72 and 25.07. Further, 10% and almost 0% have a value below the modal. If you choose a value around 21 for the scale parameter, all objects which are close to meeting this criterion will be merged in the next higher level above. In other words: you would maximize local homogeneity and simultaneously keep relevant global information by merging objects which hold more or less the same information in terms of color. In the figure below a segmentation with scale parameter 21 is shown:



You will realize that many of the bright borders have disappeared while the darker ones remain. The latter represent borders of high contrast whereas the others were not of high significance. This is comparable to eliminating contour lines from a DEM in areas with shallow slope and keeping them in areas with steep slope.

You can export the results of the Scale Parameter Analysis by selecting „Image Objects > Export Scale Parameter Analysis...“. The following dialog comes up:

**Scale Parameter Analysis**

Layer weights:

Layer name	Layer stddev.	Weight
dessau_blue.tif	(9.6)	1.0
dessau_green.tif	(6.4)	1.0
dessau_red.tif	(10.8)	1.0
dessau_nir.tif	(28.2)	1.0
dessau_mir.tif	(19.8)	1.0
dessau_term.tif	(9.1)	1.0
dessau_fir.tif	(14.2)	1.0

Select and edit weights: 1

Export

Edit file name

ScaleParam 1

Level

Level 1

Segmentation mode

Normal

Composition of homogeneity criterion:

Criterion

- Color 0.9
- Shape 0.1

- Smoothness 0.5
- Compactness 0.5

OK Cancel


In the “Export” section 1 you can enter a file name prefix under which all exported files will be saved. If you click OK the statistics results will be saved as a \*.csv and \*.dbf file. Also the histogram values will be exported as \*.csv file. Besides, the \*.csv files, an ESRI shape file \*.shp will be created in line format, which contains as attributes the potential scale parameters to merge neighboring objects. Note: all exported files will be saved in the same directory where your project data is located.

## Working with Polygons

One of the features of eCognition is the simultaneous raster and vector representation of image objects. Resulting from the polygonal representation of image objects, new possibilities are established for description and classification. In addition, polygons are needed for the visualization of image object outlines.

### Creating polygons

To create polygons a level of image objects with plain object neighborhood has to be available. For the creation of an image object level see Functional Guide > Image object generation I: Multiresolution Segmentation.

The process of polygon creation is launched by clicking  in the tool bar or selecting the menu item “Polygons > Create Polygons...”. A dialog box of great importance is opened, as the entire vector information, especially the degree of abstraction of the polygons, is based on the parameters which are specified via this interface.

Under “Threshold” in the “Base polygons” section the degree of abstraction for the base polygons can be specified. Activate “Remove slivers” to avoid any intersections of edges of adjacent polygons. Moreover, self-intersections of polygons will be prevented. Sliver removal becomes necessary with higher threshold values for base polygon generation. Sliver removal is not activated by default as the processing time to perform is high, especially for small thresholds where it is not needed anyway.




In the “Shape polygons” section the degree of abstraction for shape polygons which are independent of the topological structure can be edited in the “Threshold” box. No matter how high this value is chosen, any polygon will consist of at least three points. The threshold for shape polygons can be changed any time without the need to recalculate the base vectorization. For a description of the process of polygon creation see Chapter „Concepts & Methods. The process of polygon creation along with all its parameters can be recorded using the protocol editor.

**Note!** The generation of polygons is only possible if the segmentation was performed without diagonal neighborhood.

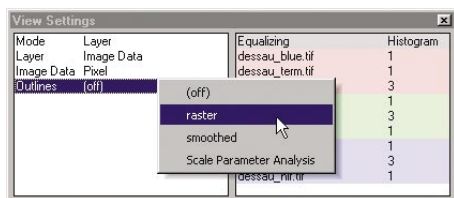


## Displaying polygons

As soon as polygons have been created, it is possible to display them. To activate the polygon display mode, click  in the tool bar. Furthermore, the polygons can be activated by using the “View Settings” control bar.

**Note!** A polygon view is only available for image object levels with polygons, while outlines can be shown after the first segmentation. Furthermore, if the polygons cannot be clearly distinguished due to a low zoom factor, they are automatically deactivated. Therefore, a higher zoom factor must be chosen in order to see the polygons.

By clicking on “Polygons” in the left list box, a context menu is opened with which the polygon display mode can be selected.

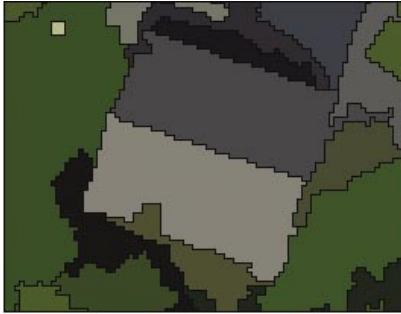
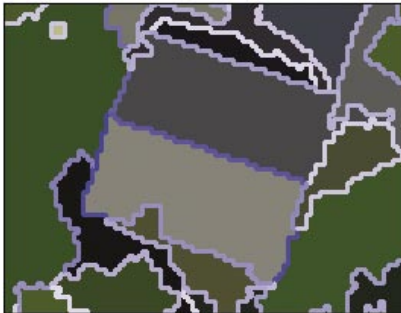


For the display of polygons, three different modes are offered:

- raster: In this mode the outlines are drawn along the pixel borders.
- smoothed: In this mode generalized polygons are drawn.
- Scale Parameter Analysis: In this mode the results of the scale parameter analysis can be shown in graduated colors.

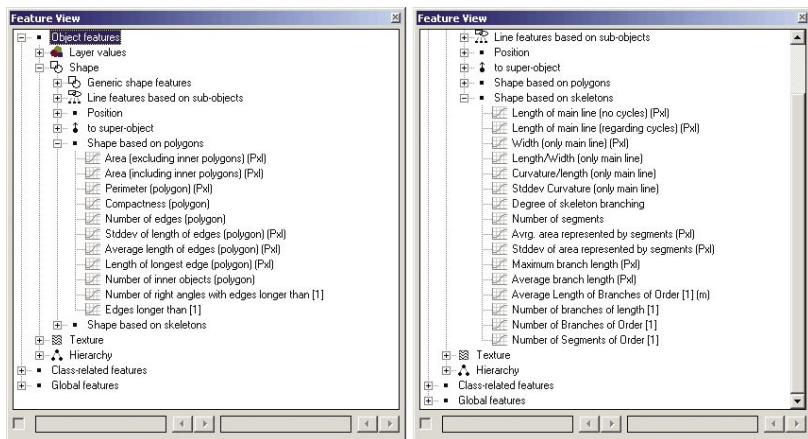
If the polygon view is activated, any time you select an image object it will be rendered along with its characterizing polygon. This polygon is more generalized than the polygons shown by the outlines and is independent of the topological structure of the image object level. Its purpose is to describe the selected image object by its shape.




*raster outline mode**smoothed outline mode**result of scale parameter analysis**selected object*

### Working with polygon and skeleton features

As mentioned above, polygons, especially generalized polygons without relation to topological structure, provide new and more detailed information for the characterization of image objects by their shape. The same is true for skeletons, which are generated automatically in conjunction with polygons. Consequently, as soon as polygons are created, a number of shape features based on polygons and skeletons are available. These features are used in the same way as other features. They can be found in the feature tree under “Object features > Shape > Shape based on polygons” or “Object features > Shape > Shape based on skeletons.”




## Displaying skeleton

As soon as polygons have been created, it is also possible to display a single object's skeleton. To activate the skeleton display mode, click  in the tool bar. Note! the object's skeleton is only displayed by selecting it.



## Deleting polygons

To delete polygons, click  in the tool bar or select the menu item "Polygons > Delete Polygons....".

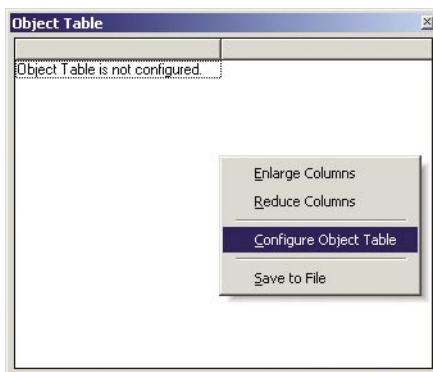
Note that the polygons and skeletons for the entire image object hierarchy will be deleted.



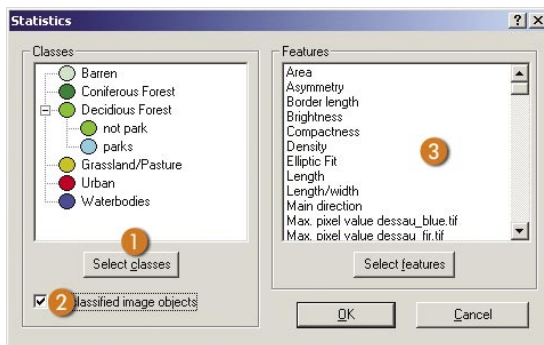
- Features
- Classification
- Class Evaluation

You can switch between “Features,” “Classification” and “Class Evaluation” by clicking the respective folder tabs.

Besides you can also use the Object Table to gain selective information on dedicated objects. When using the Object Table the first time, you are asked to configure it. To do so, you have to right-click in it and a configuration dialog comes up, called „Statistics“. Within this dialog you can determine what kind of objects together with what object features shall be displayed in the Object Table.



Click **1** to select the class of which objects shall be displayed in the Object Table. If you want unclassified objects to be displayed as well, check the respective check box **2**. On the right side you can choose the object features which shall be displayed in the table **3**.



After you have clicked OK you will see, that the Object Table now is filled up with your selected objects. In the very left column you will notice that each object has its own ID. If you click on a row in the table, you will also realize, that the table is linked with the view window(s), the Object Information dialog box(es), the Sample Editor and the Sample Selection Information dialog. This means if you click on an object in one of the view windows it is simultaneously highlighted in the object table and vice versa, except if it is not among the set of objects you have defined in the steps above.

## Features

In the “Features” window, you will get all the attributes of a selected image object. This function is essential for constructing knowledge bases. It displays the feature values of the selected image object, thus enabling you to determine which features are functioning as separating features for different classes. It should be used during the process of constructing a knowledge base to acquire information about the objects of different classes, so that functions suited to the classification can be selected.

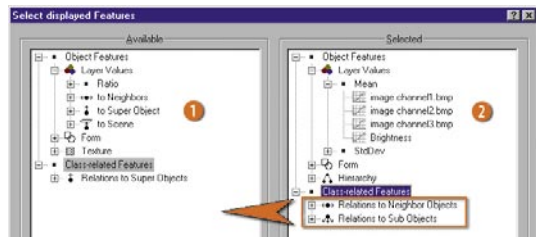
To facilitate this process, the image object information dialog has another function: If you double-click a feature, it will be displayed in this view the same way as if it had been selected in the feature view.

Since the number of features available in eCognition is quite large, not all features are displayed by default. To determine the features to be shown, open the “Select displayed Features” window by right-clicking on the “Image Object Information” dialog.

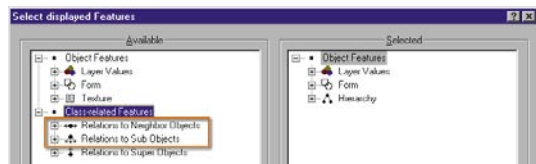


Feature	Val
<b>Layer values</b>	
Brightness	466.75
Mean red	349.98
Mean green	467.08
Mean blue	485.20
Mean ra	580.15
<b>Shape</b>	
Area	2530.00
Length	330.15
Width	70.80
Border length	1240.00
Area (polygon)	1589.50
Perimeter (polygon)	244.67
Compactness (polygon)	0.33
Number of edges (polygon)	37.00
Stdev of length of edges (polygon)	4.40
Average length of edges (polygon)	6.61
Length of longest edge (polygon)	21.10
<b>Relations to neighbor objects</b>	
Distance to class a Neighbor objects	76.27

You can now move features from “Available” ① to “Selected” ② and vice versa by double-clicking them. The features are grouped hierarchically and can be relocated as an entire group as well.



Another way to add or remove features is a right mouse-click in each dialog, where you can select features, for instance the „Feature View“ window or the „Insert Expression“ window. Select „Show Info“, and the feature appears in the „Image Objekt Information“ window.



## Classification

The classification part of the image object information gives information about the current classification as well as alternative class assignments, sorted by descending membership values.

The membership values resulting from the current classification can be seen under current classification. The results for other classes are given under alternative assignment.

## Class evaluation

This interface gives detailed information about the classification of the selected image object. The evaluation result of the selected class, the membership values of contained features and the features of parent classes (if existent) are displayed. In addition, the feature values which result in the membership value are given. For further information on this dialog, see [Functional Guide > Information on Classification](#).

## Analyzing and comparing image object attributes

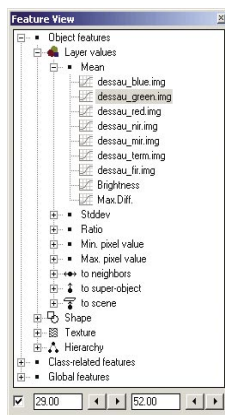
In this chapter, tools which can be used to visualize the object features for a group of objects, or all objects in a scene, are presented. As previously mentioned, it is of utmost importance to know which features are suitable for separating the classes to be specified. Therefore, it is recommended that extensive use be made of the tools provided, so as to reduce the features used for classification to the essential ones suitable for the separation of the different classes. The respective interfaces are:

- Feature view
- 2D feature space plot
- Sample editor

## Feature view

The feature view renders each image object in the current view according to its value for a selected feature. It allows you to visualize the properties of image objects in a graphical way and therefore provides an intuitive access to the peculiarity of a certain feature over all image objects in a scene. Note that not only spectral features can be visualized, but any feature usable for classification in eCognition. The feature view is a powerful tool to find features separating different classes of image objects.

Each object is displayed in a gray value according to the feature value that is selected for visualization. Objects displayed in red have not been defined for the evaluation of the chosen feature. If you view, for instance, the feature



“Rel. border to brighter neighbors” all objects without a brighter neighbor will be displayed in red.

To use the feature view, select “Tools > Feature View...” from the menu bar and double-click the feature to be visualized. If you move the mouse over an object, its feature value will be displayed.

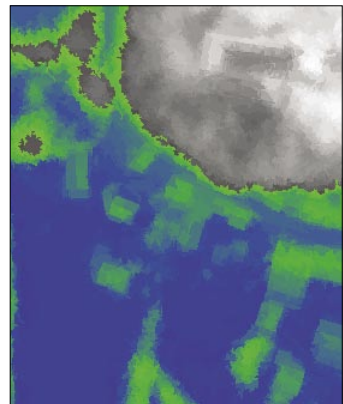
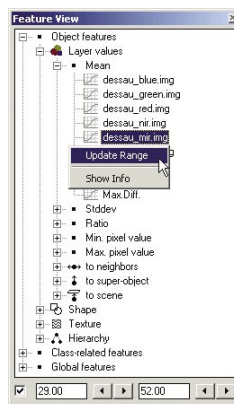
**Note!** When selecting features with a feature distance, the distance can be edited on right-click. This feature is then added to the list with the modified feature distance.

In this example, a feature view is displayed showing the ratio that the image objects have in the blue image layer. As you can see, this feature (ratio of the blue image layer) is suitable for the separation of water bodies and land mass.



Additionally, a feature range can be edited for the feature view. To limit the feature view to a certain range, select the checkbox at the bottom of the feature view window and edit the range. All image objects whose values are within the range are colored according to the adjusted range in a smooth transition from blue (low values) to green (high values). To update the range when you have selected a new feature use the right mouse button on the concerning feature and select „Update Range“. Otherwise the range of the recent feature is used.

**Note!** It is not necessary to open the “Feature View” dialog to visualize a feature. In each dialog you use to select features, like “Insert Expression” or “Select Displayed Feature”, you can select the feature you want to display with a right click and choose “Update Range”.



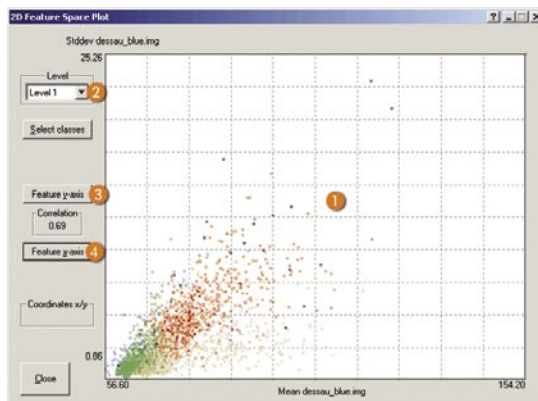
## 2D feature space plot

The 2D feature space plot is used to visualize the position all objects in a two-dimensional feature space. As with the feature view, not only spectral information can be displayed, but all features provided by eCognition.

This tool can be used to analyze the correlation of two features. If two features of a class description correlate highly, one of them can be deleted. The 2D feature space plot is also used for getting information about where an object or a group of objects is situated in the feature space.

To use the 2D feature space plot, select “Tools > 2D Feature Space Plot...” from the menu bar.

Unclassified image objects are displayed as small black crosses **1**, classified image objects as small crosses colored according to the class color. Objects assigned as samples are displayed as circles colored in the class color. The 2D feature space plot has numerous additional functions which help to obtain information about the classification of image objects (see [Functional Guide > Information on classification](#)).



In the “Level” box, **2** you can select the image object level used to create the 2D feature space plot.

To define the feature space, change the features assigned to the x- and y-axis by clicking the “Feature x-axis” **3** or “Feature y-axis” **4** buttons and selecting a feature.


## Sample editor

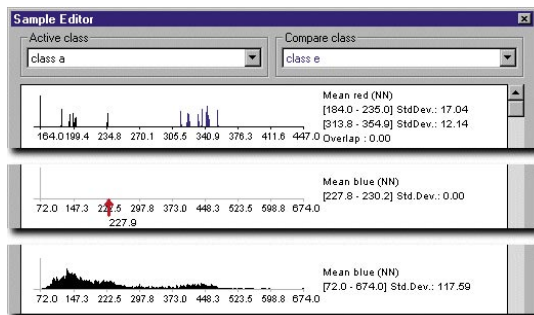
The sample editor is a graphical display for histograms of feature values. It shows, for a selected class, the histograms of the feature values of its samples for a selection of chosen features. The same can be done for all image objects of a certain level or of all levels in the image object hierarchy.



The sample editor is mainly used for comparing the attributes or histograms of image objects and samples for different classes. It is a very helpful tool in getting an overview of the feature distribution of image objects or of the samples of specific classes. The features of an image object can be compared to the total distribution of this feature over one or all image object levels. If you assign samples, features can also be compared to the samples of other classes.

Use this tool for assigning samples using nearest neighbor classification or for comparing an object to already existing samples in order to determine to which class an object belongs. It is also very useful for obtaining a quick overview of the different feature values of an object.

Open the sample editor by clicking the  button or select “Samples > Open Sample Editor...” in the menu bar.



To compare samples or layer histograms, select the classes or the levels that you want to compare in the “Active class” and “Compare class” boxes. Values of the active class are displayed in black in the diagram, the values of the compared class in blue. The value range and standard deviation of the samples are displayed to the right of the diagram.

By default, the sample editor window shows diagrams for only a selection of features. To modify this selection, right-click the sample editor and choose the features to be displayed by double-clicking them.

If you select an object, its feature value is highlighted for each feature with a red pointer. This enables you to compare different objects with regard to their feature values.

The sample editor offers a multitude of functionalities for classification as well, see [Functional Guide > Classification Basics](#).

## Classification Introduction

### Introduction

The production of image objects in eCognition is not a goal in itself. Image objects are rather basic information carriers for further classification. Compared to single pixels, image objects provide additional features aplenty and a significantly increased signal-to-noise ratio.

Thus, classification in eCognition always means classification of image objects. As described in “Concepts & Methods,” the whole classification is based on fuzzy systems. On the one hand, it enables you to come up easily and quickly with classification results based on the nearest neighbor algorithm. This alone may satisfy many needs. Beyond that, you will find a powerful and elaborate tool box, which introduces concepts and knowledge, sophisticated class descriptions and formulation as well as analysis of even complex tasks.

The context of the classification can be explained quite easily when it is compared to a database query. First the database in the form of an object hierarchy is created by the segmentation algorithm. This database contains a large amount of information represented by the objects and their features. Those features are the means to assign an object to a certain class. The class hierarchy, like the query key in a database query, defines the requirements an object must meet to be assigned to a certain class. Those requirements may be spectral, textural, contextual, or any other of the available features.


There are three chapters in the user guide on classification. This chapter, “Classification Introduction,” gives a short introduction and explains how a class is generated. “Classification Basics” discusses the basic procedures to generate membership functions, the nearest neighbor and the membership function dialog. The features for the classification are briefly treated, and then the actual classification process is described. In the final chapter, “Classification Advanced,” the hierarchical structure of the class hierarchy is the central part. Besides this, strategies for classification are given and the customized feature dialog is explained.

### How to create classes

The class hierarchy is the framework of the knowledge base in eCognition. It contains all the classes of a classification scheme and allows a hierarchical organization of the same. This hierarchy distinguishes between the passing down of class descriptions from parent to child classes on the one hand (inheritance) and meaningful semantic grouping (groups) on the other. A third register, the structure groups (structure), stands beside the other two hierarchies. This is used exclusively for classification-based seg-

mentation and is therefore not a part of the knowledge base for classification. The class hierarchy is a powerful tool for defining class semantics and also helps to reduce complexity when creating a knowledge base.

In the following passage, the creation of classes is described. To get information on the inheritance and groups hierarchies as well as their interrelations, see [Functional Guide > Classification Advanced](#).

The “Class Hierarchy” dialog is opened by selecting “Classification > Open Class Hierarchy...” or clicking . As mentioned, the class hierarchy distinguishes between inheritance, groups and structure. Those distinctions help to give additional meaning to the classes. They do not represent different sets of classes. This means that each class can be found in all the different hierarchies. To switch from one to the other, simply click the respective tab control in the “Class Hierarchy” dialog.

**Short explanation of terms:** In order to keep to an overview, only a brief explanation of the terms used in the following text (knowledge base, class hierarchy, class, class description, and expression) is given here:

<b>knowledge base</b>	Generally, a knowledge base summarizes all algorithms, definitions and parameters necessary to process information for a certain purpose. A complete image analysis protocol, including all necessary steps for segmentation, the loading of class hierarchies and classifications, is the knowledge base for an entire interpretation process. The class hierarchy is the knowledge base for the image object classification.
<b>class hierarchy</b>	The class hierarchy is the knowledge base for the classification of image objects. It contains the sum of all classes with their specific class descriptions. The classes can be structured in a hierarchical form. eCognition distinguishes between an inheritance hierarchy and a groups hierarchy.
<b>class</b>	In eCognition class is used as a synonym for class description. A class is a group of objects, all of which meet the same standard set by a class description.
<b>class description</b>	The class description is an essential part of each class. In it those characteristics of image objects are defined which determine membership in the respective class. This feature description is adjusted by means of fuzzy expressions. In addition, general information like class name or class color is specified in an appropriate dialog.

**expression**

Expressions are the basic elements for the definition of class descriptions. They can be distinguished into nearest neighbor, membership functions, similarities and logical terms. While nearest neighbor, membership functions and similarities represent conditions, operators are the elements used to combine different conditions.

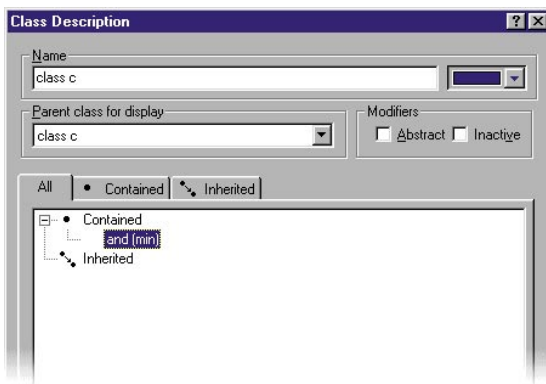
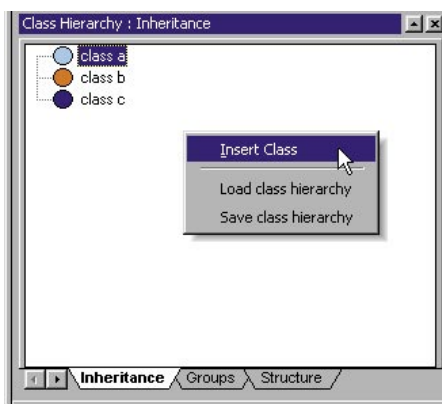
**Inserting a new class**

When starting a new project, you always begin with an empty class hierarchy. You can either insert new classes or, once you have created classes, copy these within the class hierarchy. Adding and copying classes can only be done in the inheritance and the groups hierarchy, since the structure folder is not part of the classification.

To insert a new class, select “Classification > Edit Classes > Insert Class...” in the class hierarchy menu or right-click the “Class Hierarchy” window and select “Insert Class.” The dialog box “Class Description” opens. Insert a new name for the class and choose an appropriate color. You can identify the new class by its colored circle and its name in the class hierarchy. Now the basic class description is generated. Be aware that naming and giving a color to a class does not define it.

To learn how to define a class see “Editing the class description” below.

To copy a class, select the class to be copied and choose “Classification > Edit Classes > Copy Class...” or click the right mouse button and select “Copy Class.” The class inserted in the class hierarchy is




named “Copy of...” and given the same color as the copied class. Use this function to duplicate the feature description of an existing class in a new class, which can then be modified or extended. By doing so, time is saved when creating different classes which have similar features.

## General class settings

There are several general class settings, which can be edited in the “Class Description” window.

To name a class, modify the name given in the “Name” ❶ window.

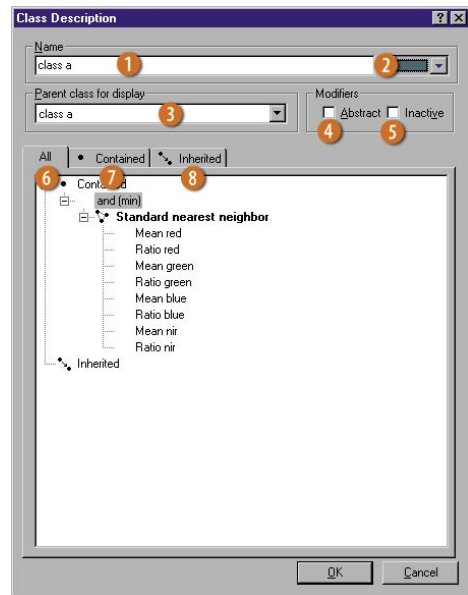
The class color can be defined by clicking the color field. ❷

In the field “Parent class for display” ❸ you can define which of the parent classes in the groups hierarchy – in the case of multiple parent classes – should be displayed for a child class when navigating through the levels of the groups hierarchy using the buttons .

As explained above, classes which have applicable child classes are treated like abstract classes due to the inheritance logic contained in eCognition. Classes without child classes can also be set to abstract. This can be of use when formulating similarities to classes without wanting to classify the class to which the similarities refer. To set a class to abstract, click the “Abstract” ❹ checkbox or right-click a class in the “Class Hierarchy” window and choose “Abstract” or choose “Classification > Edit Classes > Abstract” from the menu bar.

Classes can be set inactive to exclude them from classification without deleting them. This can be done in three ways:

- Select a class in the “Class Hierarchy” window and deselect “Classification > Edit Classes > Active” from the menu bar.



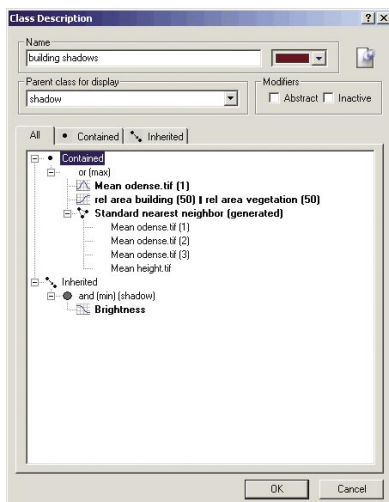
- Right-click a class in the “Class Hierarchy” window and deselect the “Active” check box.
- Click the “Inactive” ⑤ box in the “Class Description” window.

In the “Class Description” window, you can switch between three different view modes. In the “All” ⑥ mode, the inherited as well as the contained expressions are displayed. In the “Contained” ⑦ mode, only expressions defined for this particular class are displayed. In the “Inherited” ⑧ mode, only the expressions inherited from parent classes are displayed.

## Editing the class description

The definition of the features describing a certain class and the logic by which these features are combined are both handled in the main window of the dialog box “Class Description.”

eCognition uses different types of expressions to create a class description. While nearest neighbor and membership functions are used to translate feature values of arbitrary range into a value between 0 (= no membership) and 1 (= full membership), logical operators such as “and” and “or” summarize these return values under an overall class evaluation value, again between 0 and 1. Similarities are a special type of expression: Similarities enable the inclusion of the evaluation of complete class descriptions of other classes. For more information about the different classifiers see [Functional Guide > Classification Basics](#).



The graphical display of the class description is hierarchical: all expressions which are connected by a certain logical operator are displayed as subunits of this logical operator.

## Inserting an expression

To insert expressions, right-click the operator in the “Class Description” dialog and choose “Insert new Expression” (by default you will

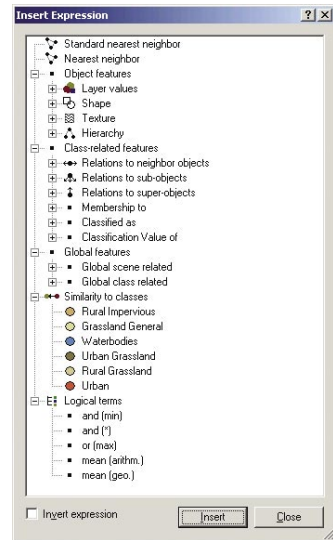


find the operator “and (min)” in a new or in an empty class description) or double-click the operator.

The dialog box “Insert Expression” will open. This dialog has a similar structure to that of Windows Explorer. Choose the expression you want to insert by opening the relevant folders. If you need further logical expressions for your class description, you can find them under “Logical terms” and insert them in the same way as any other expression. Inserted expressions can always be removed.

Either selecting it and clicking “Insert” or double-clicking it inserts the expression. While operators and similarities can be inserted into a class as they are, the nearest neighbor and the membership functions require further definition.

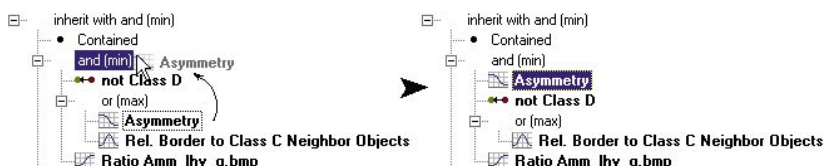
If you have selected a membership function, the dialog box “Membership Function” will open before the expression is inserted as part of the class description. For information about how to edit a membership function, see [Functional Guide > Classification Basics](#).



If you select the nearest neighbor, it will be inserted into the class with default settings. The nearest neighbor alone does not suffice to describe the class since it needs samples. For a description of the sample input and nearest neighbor settings, see [Functional Guide > Classification Basics](#).

## Moving expressions and editing the hierarchy of logical operations

You will often have to move expressions and logical operators when constructing class rules. An expression is moved in a similar way as a class is moved in the class hierarchy. Click the expression you want to move with the left mouse button, drag it over the logical expression you want to combine it with and drop it there.



## Inverting expressions

All expressions can be inverted due to the underlying concept of fuzzy logic in eCognition's class descriptions in order to translate all feature values into the opposite meaning [0; 1]. Mathematically speaking, an inverted expression has the value 1 minus the membership value. Since logical operators are expressions that combine the membership values of a number of expressions, they yield a membership degree themselves and can consequently be inverted as well. An inverted expression appears in the class description with a prefixed "not." The most interesting application for the inversion of expressions is its application to similarities. This makes it possible to define that "*Class A* is not *Class B*."

There are two possibilities of inverting an expression:

- Check the dialog item "Invert Expression" before inserting it into the class description.
- If the expression is already inserted into the class description, right-click it and choose "Invert Expression" in the following context menu.

## Editing an expression

To edit an expression, right-click it and choose the menu item "Edit Expression" in the context menu. If you do this for a feature, the "Membership Function" dialog will pop up. If you select a logical expression, you can choose from all the possible logical operators and, if you select a nearest neighbor classifier, a dialog will pop up which allows you to edit the appropriate feature space.

To edit a feature or a nearest neighbor classifier, you can also double-click the expression.

## Deleting an expression

To delete an expression, right-click it and choose the item "Delete Expression" in the following context menu.

## Deleting a class

To delete a class, select it and press the delete key on your keyboard or choose the menu item "Classification > Edit Classes > Delete Class" from the menu bar.



### Deleting a class hierarchy

To delete a whole class hierarchy, choose the menu item “Classification > Delete Class Hierarchy.”

## Classification Basics

eCognition uses a fuzzy rule base to classify image objects. A fuzzy rule base consists of one or more conditions which are combined by operators. To include features into a fuzzy rule base membership functions for the considered features have to be defined. Membership functions permit the inclusion of knowledge or concepts into your knowledge base. If you know, for example, that a certain class is characterized by low mean values in a certain image layer, then you can easily formulate this knowledge by using and editing a membership function. Membership functions are especially suitable for describing relations to other networked objects, such as the object's embedding in a certain environment.

In eCognition, membership functions can be defined in two ways:

- Nearest neighbor – automatic generation of multidimensional membership functions based on sample objects
- Membership function dialog – design of one-dimensional membership classes with a graphical interface

### Nearest neighbor

Whereas the graphical interface (the membership function dialog) allows the design of one-dimensional membership functions, eCognition automatically develops multidimensional membership functions with the nearest neighbor classification. They are not visualized and only used internally to evaluate the results of nearest neighbor classification.

eCognition distinguishes between two types of nearest neighbor expressions: nearest neighbor (NN) and standard nearest neighbor (standard NN). The main difference between these expressions is that the nearest neighbor and thus its feature space can be defined for each single class independently. In contrast, the feature space of the standard nearest neighbor is valid for the whole project and thereby for all classes to which the standard nearest neighbor expression is assigned. Additionally, it is not possible to apply more than one standard NN to one class description. The standard NN is useful, because in many cases the separation of classes only makes sense when operating in the same feature space. The theoretical background of the nearest neighbor classifier is explained in [Concepts & Methods > Nearest neighbor](#).

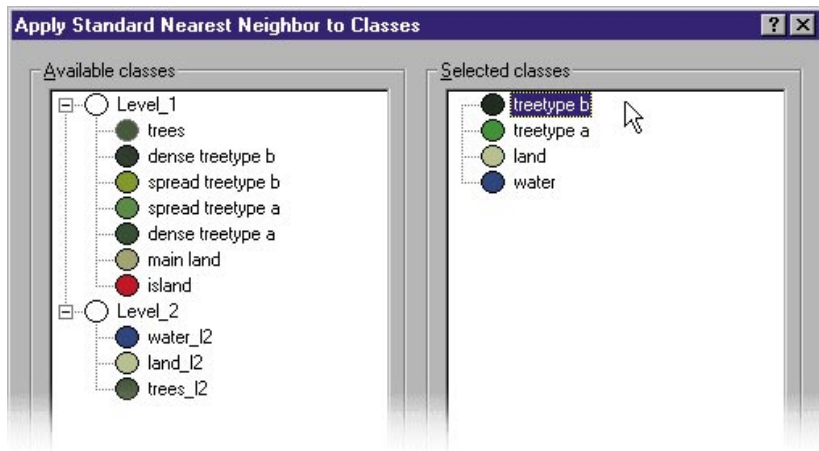
It is not possible to use class-related features to span a feature space for a nearest neighbor classification since an absolute reference is necessary.

It is also possible to use some of the class-related features implemented in eCognition, but only with some restrictions. For more information on how to use class-related features in conjunction with the nearest neighbor classifier see [Concepts & Methods > Nearest neighbor](#).

### Inserting nearest neighbor or standard nearest neighbor

Both the nearest neighbor and the standard nearest neighbor classifier can be inserted into the class description just like any other expression. The standard nearest neighbor can additionally be inserted via a separate menu to make the definition of several classes more convenient. Select “Classification > Nearest Neighbor > Apply Standard NN to Classes...” from the menu bar. In the “Apply Standard Nearest Neighbor to Classes...” menu, you can move classes from the “Available classes” to the “Selected classes” by clicking them.

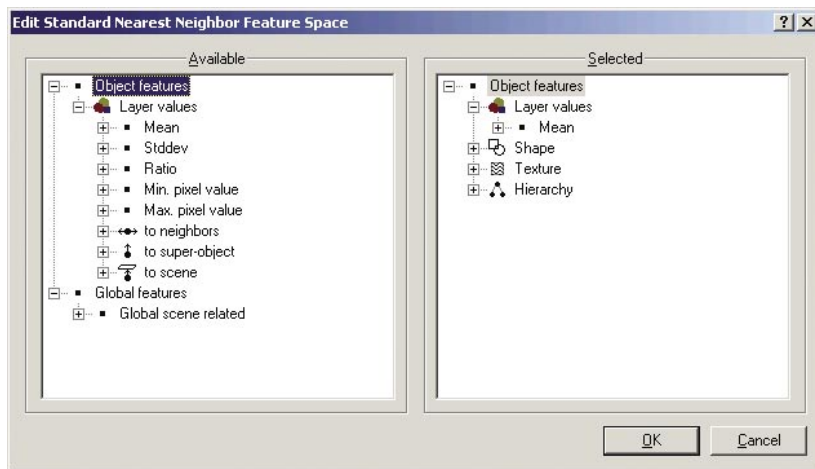
If you select the standard nearest neighbor, it is implicitly inserted with the predefined feature space. The class-specific nearest neighbor, however, has to be edited after insertion for each single class it is assigned to.



### Defining the feature space

Both the nearest neighbor and the standard nearest neighbor classifier can be edited by double-clicking them in the class description. This opens a dialog which allows a selection of available and selected features for the feature space. The “Available” box shows the features that are available for spanning the feature space. In the “Selected” box you can find the selected features. To add a feature to the feature space, navigate to it in the tree in the left box and double-click it. The selected feature will appear in the “Sele-

ted” box. To remove a feature from the feature space, double-click it in the “Selected” box. It will be removed and reappear in the “Available” box.



The feature space of the standard NN can additionally be edited centrally from the menu bar. Select “Classification > Nearest Neighbor > Edit Standard NN Feature Space...”

Note that the standard NN feature space is defined for the entire project. This means that if you change the standard NN feature space in one class description, these changes affect all classes that contain the standard NN expression.

When a class hierarchy is saved, the feature space of the included nearest neighbor expression is saved along with it. To optimize the performance, the feature space is saved only for the selected features. Accordingly it is not possible to change the feature space of a loaded class hierarchy and at the same time still use the old samples. If both standard NN and the individual nearest neighbor are used, the combination of all feature spaces is saved.

### Inserting sample objects manually


A nearest neighbor classification needs training areas. In eCognition, a representative collection of image objects, the so-called sample objects, meets these requirements.

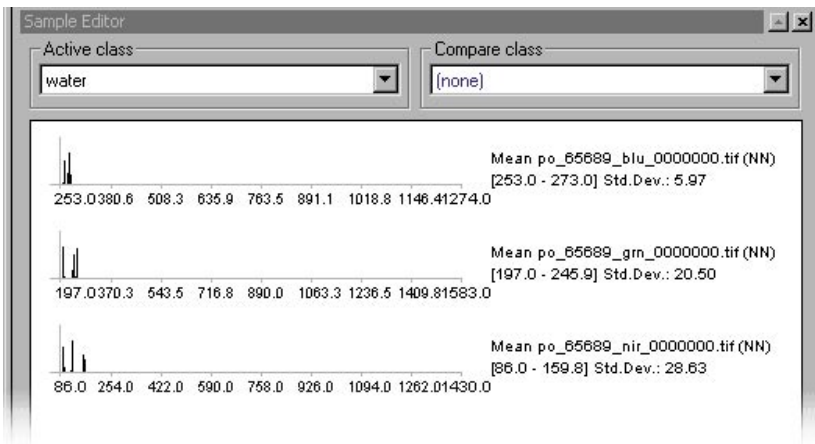
To assign sample objects, set the input mode to “Input Samples.” The sample editor supports the gathering of sample objects. After setting the input mode to “Input Samples,” the sample editor opens automatically and the view is set to the sample mode. To define an object as sample for a class, first select the class in either the class hierarchy or

the “Active class” drop-down menu of the sample editor and then simply double-click the respective object. Alternatively to double-clicking, a combination of the shift key and mouse-click can be used.

As long as the sample input mode is activated, the view will always change back to the sample view when an object is selected. The sample view mode displays sample objects in the class color. This way the accidental input of samples can be avoided.

The feature values of the sample object are displayed as little histograms in the sample editor. To undo the declaration of an object as sample, double-click or shift-click the sample object again. The feature range displayed for each feature is limited by default to the actually detected feature range. To display the whole feature range, right-click the sample editor and select “Display entire Feature Range.”

To open or close the sample editor independently from the sample input mode, either select “Samples > Open Sample Editor...” from the menu, or click  in the tool bar.



It can happen that not all features displayed in the sample editor are part of the nearest neighbor feature space. Those features which belong to the NN feature space are marked with a “(NN)” behind the feature name. To define which features are to be displayed in the sample editor, right-click the sample editor and click “Select Features for Display....”

**Note!** If you already have a class hierarchy, samples of sub-classes are also samples of their super-class(es). Nevertheless, you can add further samples to the super-class(es).

To show the samples coming from the class' sub-classes, right-click the sample editor and select "Display Samples from Inherited Classes."

For a detailed description of the single dialog features see [Functional Guide > Information on Image Objects and Features](#).

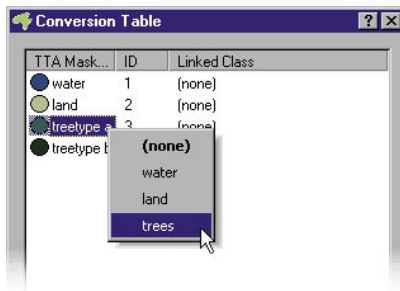
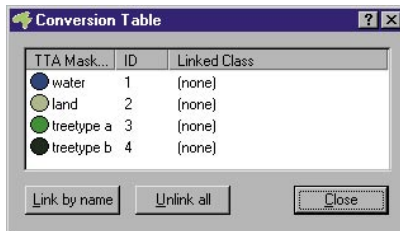
It is recommended to begin with inserting only a small number of sample objects (about three) as representatives for each class. Then start the classification process. The result will certainly contain a number of misclassifications, but it can be improved very quickly by iteratively adding typically wrongly classified image objects samples of the correct class to the sample editor and repeating the classification process. Several runs of this procedure, if necessary, achieve remarkably stable classification results in an easy way.

### Inserting sample objects automatically using the TTA mask

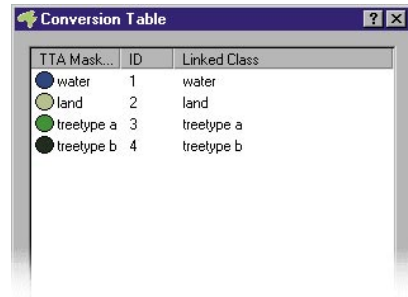
In addition to the manual assignment, eCognition allows the automatic declaration of sample objects using a TTA mask (training and test area mask).. The TTA mask has to contain geocoding information or be of the same geometric dimensions as the image layers used in the project to be loaded. The single training areas contained in the TTA mask do not necessarily have to be identical to your image objects. With this feature, eCognition is able to use information processed with other software as a basis for a classification.

To import a TTA mask, open the respective dialog via the menu "Samples > Load TTA Mask..." Select the files containing both the conversion table and the attribute table that contains the classification of the single training areas. You will be asked whether you want to create classes in your class hierarchy based on the classes in the conversion table. If you have already created a class hierarchy, this may not be necessary. If you select "yes," you can either delete the existing classes if there are any, or add the classes to your class hierarchy.

The next step is to select the file containing the TTA mask itself. After the TTA mask has been loaded, it will be



displayed in the view window. If you previously decided to create classes from the conversion table, you can go straight on with the creation of samples from the TTA mask. Otherwise, you first have to link the classes of the conversion table to their counterparts in your class hierarchy. The link can be created manually, or automatically if the names of your class hierarchy correspond with the classes of the conversion table.



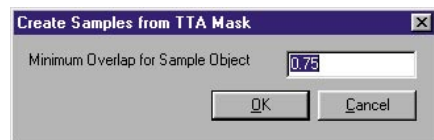
TTA Mask...	ID	Linked Class
water	1	water
land	2	land
treetype a	3	treetype a
treetype b	4	treetype b

How classes included in the TTA mask are linked to the classes of your project is defined in the conversion table. The entries in the conversion table can be checked by selecting the menu item “Samples > Edit Conversion Table...”. Normally, when you first load a TTA mask which was created in another eCognition project or even using another software package, there is no linkage between classes of the conversion table and your project. To automatically link the classes, select “Link by name” from the conversion table. To manually link the TTA mask classes to their counterparts in the class hierarchy, right-click on the class name of the TTA mask and select the corresponding class name from the drop-down menu.

After the classes of the project have been linked to the classes of the TTA mask, information about this linkage is provided by the conversion table.

The next step is the actual generation of samples from the TTA mask. Select the menu item “Samples > Create Samples from TTA Mask...”. A new dialog will pop up in which the image object level for declaring sample objects can be chosen. Choose the appropriate level. After that, another dialog opens in which to define the minimum overlap for TTA mask area and sample object. By default this is 0.75 (75 %). Since the single training areas contained in the TTA mask do not necessarily have to match your image objects, a criterion is needed to decide whether an image object that is not 100 % within a training area in the TTA mask should be declared a sample object. 0.75 means that 75 % of an image object has to be covered by the sample area for a certain class given by the TTA mask in order for a sample object for this class to be generated.

After you have edited the minimum overlap, click “OK” and you will see the sample objects highlighted in their class color. If you change to the sample editor, you will notice that the samples have been inserted.



## Storing samples as a TTA mask

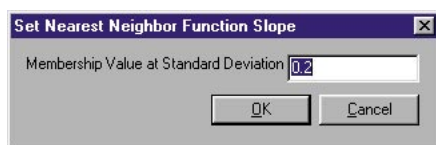
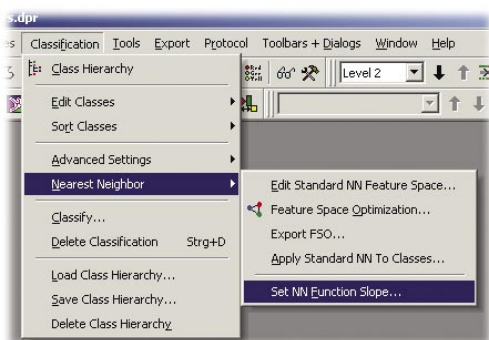
To save samples as training or test areas for other similar eCognition projects, samples can be converted to TTA masks by selecting the menu item “Samples > Create TTA Mask from Samples....” This TTA mask can then be saved in the form of an ASCII raster file to store the actual topology of the samples used.

## Changing the function slope of the nearest neighbor classifier

The basic effect of the function slope is to increase or decrease the distance an object may have from the nearest sample in feature space while still being classified. Therefore, also more overlap between classes can occur. Accordingly, a higher function slope value will result in more classified image objects with less separability. Lower function slope values will produce more unclassified image objects, but also result in better classification stability. A more detailed description can be found in “Concepts & Methods.” The function can be edited by selecting “Classification > Nearest Neighbor > Function Slope....”

The default setting for the function slope is a value of 0.2.

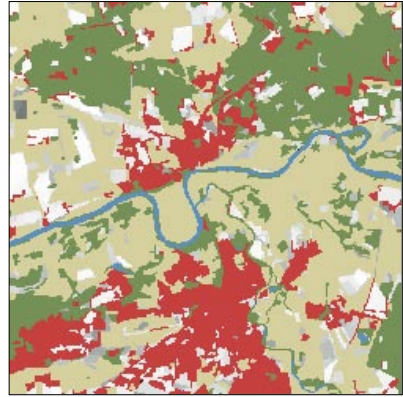
In the following images a number of sample objects are declared and the classification process based on different values for the function slope is executed.





The following images show the effect of changing the value of the function slope. A function slope with a value of 0.1 leads to a classification with numerous unclassified image objects (white and gray objects). A higher value of 0.8 provides a classification for almost every image object.

**NN function slope: 0.1**



**NN function slope: 0.2**



**NN function slope: 0.8**



## Deleting sample objects

To delete single sample objects, simply double-click or shift-click them.

To delete samples of specific classes, select the menu item “Samples > Delete Samples of Classes...” move the respective classes from the “Selected classes” to the “Available classes” window by clicking them. Click “OK” when you have finished.

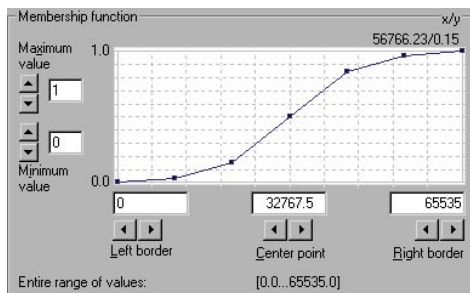
To delete all samples you have assigned, select the menu item “Samples > Delete all Samples.” Click “Yes” in the dialog which appears asking whether all samples of all classes should be deleted.

## Fuzzy rule base

A fuzzy rule base allows the formulation of knowledge and concepts in a very efficient way. The first step is the manual or automatic definition of membership functions. This way, your expert knowledge about general class descriptions as well as relations of classes can be incorporated into the system.

Membership functions are easy to edit and adapt for each feature. They offer a transparent relationship between feature values and the degree of membership to a class. Since their graphical editing is restricted to the definition of one-dimensional membership functions, their use is recommended if a class can be separated from other classes by

only one or a few features. The “Membership Function” dialog is automatically opened when you insert a new expression into a class description. To open this dialog for already inserted expressions, double-click or right-click them and choose “Edit Class” in the following context menu.



A membership function is basically defined by its left and right border values in combination with the function slope. In the example function the range of values is from 0 to 255. The part of this range which is fuzzy (i.e., not definitely “yes” or “no”) lies between 50 and 57, as defined by the left and right border values. The way a feature value is translated into a membership value is defined by the function slope. The example function slope starts at 0 and rises to 1 on the x-axis. This results in a membership value of 0 for a feature value of 50, a membership value of 0.5 for a feature value of 53.5, and a


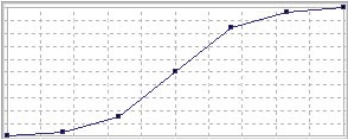

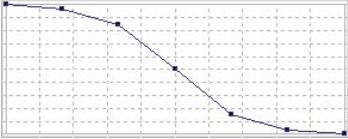

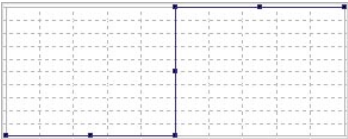

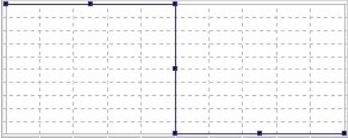
membership value of 1 for a feature value of 57. Any image object with a feature value lower than the left border receives a membership value of 0; any object with a feature value higher than the right border obtains a membership value of 1.

Membership function type

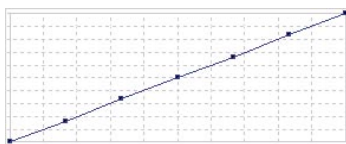
The function slope describes how the membership value for the specific expression is calculated for a certain feature value of an image object.

To choose another form for the function, use the buttons to initialize the function.

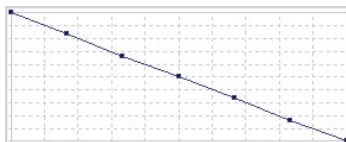


Function form	Button	Function slope
Larger than		
Smaller than		
Larger than (boolean)		
Smaller than (boolean)		

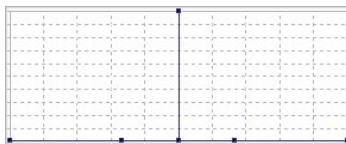
Linear larger than



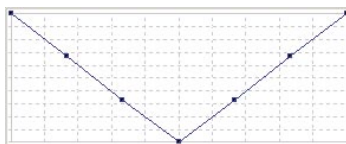
Linear smaller than



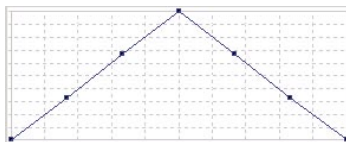
Singleton  
(exact one member)



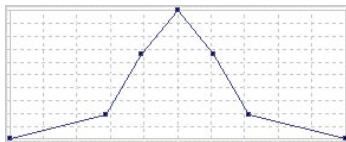
Linear range ("V")



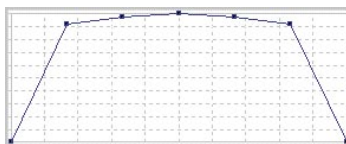
Linear range  
("upside down V")



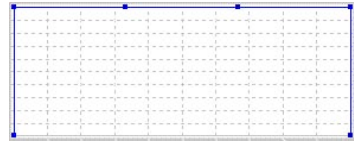
Approximate Gaussian



About range

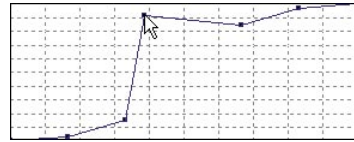


Range (boolean)



Note that in each case the left border value of the membership function is assigned to all feature values less than the left border value and the right border value of the membership function is assigned to all feature values greater than the right border value.

You can adapt the function slope to your requirements by moving the nodal points as well, if this is necessary to integrate special knowledge. However, this is not usually necessary. Consider that the more general and the larger the range of the membership function, the more robust your rule base will be. We recommend generating membership functions which are as general and large as possible while still ensuring satisfying results on a given data set. This method will make it easier to transfer your rule base to other data sets.



## Value range

The value range defines to which range of feature values the function slope is applied. To define the value range of the function slope, the left and right border as well as the center point can be varied. This is done by either defining values in the value boxes or using the arrows. Note that the left border value always needs to be smaller than the right border value. The center value is calculated automatically if you first determine the left and right borders. When changing the center point, the whole range defined by left and right border value will be moved.



## Maximum and minimum values

The maximum and minimum values define the maximum and minimum membership values an expression can achieve. By default the maximum is set to 1 and the minimum to 0. In some cases, however, it might be useful to define that an expression can never produce a membership value of 1, but only of 0.9, for example. This way one class always obtains priority in the case of close classification results. The values can be defined by either directly editing them or using the arrows.

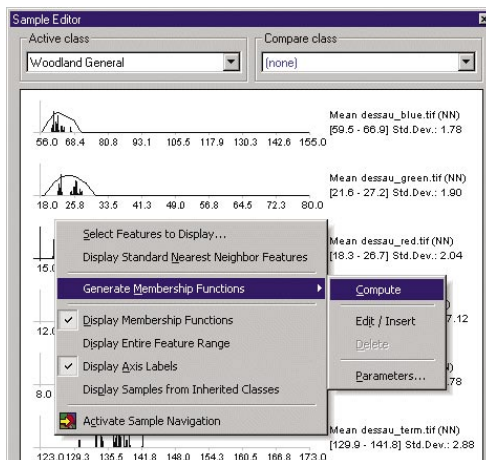


### Generating membership functions automatically

The normal procedure is to define membership functions manually. However, in some cases, especially when classes can be easily and clearly distinguished, it is more convenient to automatically generate membership functions. This can be done within the sample editor.

To generate a membership function, right-click the respective feature and select “Generate Membership Function > Compute.” Instead of having the function automatically com-

puted you can also insert and define it by selecting “Edit / Insert.” The same function allows you to edit an automatically generated function.



To delete a generated membership function, select “Delete.”

The checkbox “Display Membership Functions” allows you to switch on and off the displaying of generated membership functions. Manually inserted membership functions will not be displayed.

**Note!** Computing membership functions is an easy and fast way to create class descriptions. But do not let yourself be tempted by this ease. As described in the “Concepts & Methods,” membership functions are only reliable if no strong overlap within the classes’ feature spaces exists. Rule bases with a multitude of generated membership functions do not necessarily produce good results.

### Operators

After the manual or automatic definition of membership functions, fuzzy logic can be applied to combine these fuzzified features with operators. Generally, fuzzy rules set certain conditions which result in a membership value to a class. If the condition only depends on one feature, no logic operators would be necessary to model it. However, there are usually multidimensional dependencies in the feature space and you may have

to model a logic combination of features to represent this condition. This combination is performed with fuzzy logic. Fuzzy logic allows the modelling several concepts of “and” and “or.” The most common and simplest combination is the realization of “and” by the minimum operator and “or” by the maximum operator.

When the maximum operator “or (max)” is used, the membership of the output equals the maximum fulfilment of the single statements. The maximum operator corresponds to the minimum operator “and (min)” which equals the minimum fulfilment of the single statements. This means that out of a number of conditions combined by the maximum operator, the highest membership value is returned. If the minimum operator is used, the condition that produces the lowest value determines the return value.

The other operators have the main difference that the values of all contained conditions contribute to the output, whereas for minimum and maximum only one statement determines the output.

When creating a new class, its conditions are combined with the minimum operator “and (min)” by default. The default operator can be changed and additional operators can be inserted to build complex class descriptions, if necessary. The available operators are listed below. For given input values the membership degree of the condition and therefore of the output will decrease with the following sequence:

<b>or (max)</b>	“or”-operator returning the maximum of the fuzzy values, the strongest “or”
<b>mean (arithm.)</b>	arithmetic mean of the fuzzy values
<b>and (min)</b>	“and”-operator returning the minimum of the fuzzy values (used by default, the most reluctant “and”)
<b>mean (geo.)</b>	geometric mean of the fuzzy values
<b>and (*)</b>	“and”-operator returning the product of the fuzzy values

Applicable to all of the above:

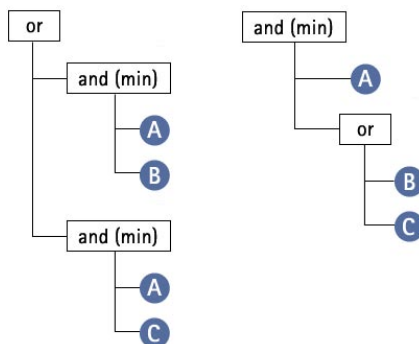
<b>not</b>	inversion of a fuzzy value: returns $1 - \text{fuzzy value}$
------------	--

To change the default operator, right-click the operator and select “Edit Expression.” You can now choose from the available operators. To insert additional operators, open the “Insert Expression” menu and select an operator under “Logical Terms.” To insert an inverted operator, activate the “Invert Expression” box in the same dialog; this nega-

tes the operator (returns  $1 - \text{fuzzy value}$ ): “not and (min).” To combine classes with the newly inserted operators, click and drag the respective classes onto the operator.

### Hierarchy of logical operators

Expressions and logical operators presented in eCognition can be combined to form well-structured class descriptions. Thereby, class descriptions can be designed very flexibly on the one hand, and very specifically on the other. An operator can combine either expressions only, or expressions and additional operators which again link expressions.



An example of the flexibility of the operators is given in the above picture. Both constellations represent the same conditions to be met in order to classify an object.

### Features and expressions

The following paragraphs give a short overview of the features and expressions provided by eCognition and the purposes they serve. For a detailed description see [Concepts & Methods > Features and terms for the fuzzy class description](#). Two groups of features are distinguished: object features and class-related features.

#### Object features

In contrast to class-related features, object features do not depend on the classification of other networked image objects. They are stand-alone.

##### Layer values

Layer values evaluate the first and second statistical moment (mean and standard deviation) of an image object's pixel value and the object's relations to other image object's pixel values. Use them to describe image objects with information derived from their spectral properties.

##### Shape

Shape features evaluate the image object's shape in a variety of respects. The basic shape features are calculated based on the



object's pixels. When polygons are generated, additional shape features based on polygons and skeletons can be used. Another type of shape features, based on sub-object analysis, is available as a result of the hierarchical structure. If image objects of a certain class stand out because of their shape, you are likely to find a form feature that describes them.

<b>Texture</b>	The image object's texture can be evaluated using different texture features. eCognition offers a new type of texture features based on an analysis of sub-objects. These are especially helpful for evaluating highly textured data. Besides, a huge number of features based upon the co-occurrence matrix after Haralick can be used.
<b>Hierarchy</b>	This feature provides information about the embedding of the image object in the image object hierarchy. These features are best suited for structuring a class hierarchy when you are working with an image object hierarchy consisting of more than one image object level.
<b>Thematic attributes</b>	If your project contains a thematic layer, the object's thematic properties (taken from the thematic layer) can be evaluated. Depending on the attributes of the thematic layer, a large range of different features becomes available.
<b>Customized features</b>	All features created in the customized feature dialog and which do not refer to other classes are displayed here.

### Class-related features

Class-related features refer to the classification of other image objects situated at any location in the image object hierarchy. This location can be defined by a vertical distance in the image object hierarchy (super-objects and sub-objects) or by a horizontal distance (neighbor objects). The distance is determined by the feature distance.

<b>Relations to neighbor objects</b>	Use these features to describe an image object by its mutual relationships to other image objects assigned to a certain class on the same level.
<b>Relations to sub-objects</b>	Use these features to describe an image object by its relationships to other image objects, assigned to a certain class on a lower level. Since the resolution increases the lower you

move in the image object hierarchy, you can evaluate sub-scale information using these features.

**Relations to super-objects**

Use these features to describe an image object by its relations to other image objects assigned to a certain class on a higher level in the image object hierarchy. Analogous to the relations to sub-objects, it is possible to evaluate super-scale information here.

**Membership to**

In some cases it is important to incorporate the membership value to different classes in one class. This function allows explicit addressing of the membership values to different classes.

**Classified as**

The idea of this feature is to enable the user to refer to the classification of an object without regard to the membership value. It can be used to “freeze” a classification.

**Classification value of**

This function allows you to explicitly address the membership values to all classes. As opposed to the feature „Membership to“ it is possible to apply all membership values to all classes without restrictions.

**Customized features**

All features created in the customized feature dialog which refer to other classes are displayed here.

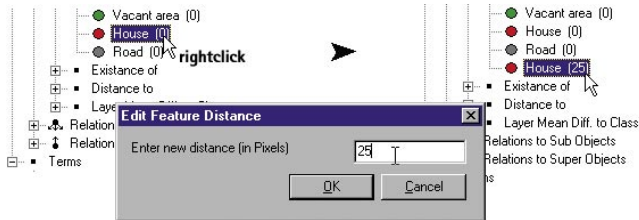
## Feature distance

If you go through the feature list you will notice some features that have a number in brackets attached. This is the so-called feature distance. Two types of feature distance are distinguished: the **“vertical”** distance between image objects on different levels in the hierarchy of image objects and the **“horizontal”** distance between image objects situated on the same level. In the first case the feature distance is the number of image object levels that have to be crossed when navigating from one object of concern to the other. In the second case, it is the spatial distance (measured in pixels) between two image objects.

For both cases, the feature distance is edited in the same way. There are two possibilities:

- If you have already inserted the expression, the feature distance can be edited by clicking the “Feature distance” button in the respective “Membership Function” dialog.

- The feature distance can also be edited in the feature list by right-clicking the respective expression. The feature will then be added to the list with a new distance.



### Object oriented texture analysis

eCognition contains a powerful method for object oriented texture analysis. By analyzing sub-objects, it is possible to describe image objects by their texture. An important aspect of this method is that the respective segmentation parameters of the sub-object level can easily be adapted to come up with sub-objects that represent the decisive structures of a texture.

A straightforward method is to use the predefined texture features provided by eCognition. They enable you to characterize image objects by texture, determined by the spectral properties, contrasts and shape properties of their sub-objects.

Another approach to object oriented texture analysis is to analyze the composition of classified sub-objects. Class-related features (relations to sub-objects) can be utilized to provide texture information about an image object, e.g., the relative area covered by sub-objects of a certain classification.

Further texture features are provided by texture after Haralick. These features are calculated based upon the so-called co-occurrence matrix, which is created out of the pixels of an object.

**Note!** The calculation of Haralick texture features can be very time consuming, since for every pixel of an object a 256 x 256 matrix has to be calculated.

### Think ahead when using class-related features

Class-related features are very powerful tools for introducing semantic context into the classification. However, there is a certain peculiarity in using them intensively. Since class-related features, as the name implies, always relate to the classification of other

image objects, there is the need for absolute references (i.e., objects classified exclusively with the use of object features or by nearest neighbor). If class-related features refer to image objects which have themselves been classified by the use of class-related features, a lack of absolute reference is possible and the classification result might be unstable, especially if a class description refers to another class description, which itself refers to the first class description (cyclic dependency). eCognition provides a method of handling this problem, such as the execution of a user-defined number of classification cycles or a simulated annealing approach, but it is preferable and more elegant to keep the entire class hierarchy along with its set of class rules predictable.

Another problem can arise from the internal classification logic used by eCognition. When classifying without class-related features, all classes containing class-related features in their class descriptions are ignored during the classification process. On the other hand, it is recommended to first classify without class relations to create an absolute reference for class descriptions referring to the classification of other image objects. If class-related features are added to existing class descriptions consisting only of object features, the respective class can no longer be used to create an absolute reference which might affect the classification result. Consequently, always take this into account when adding class-related features to a class description.

## Similarities

Similarities work like the inheritance of class descriptions. Basically, adding a similarity to a class description is equivalent to inheriting from this class. However, since similarities are part of the class description, they can be used with much more flexibility than an inherited feature. This is particularly obvious when they are combined by logical terms.

A very useful method is the application of inverted similarities as a sort of negative inheritance: consider and name a class *bright* if it is defined by high channel mean values. You can define a class *dark* by inserting a similarity feature to *bright* and inverting it, thus yielding the meaning “*dark* is not *bright*.”

It is important to notice that this formulation of “*dark* is not *bright*” refers to similarities and not to classification. An object with a membership value of 0.25 to the class *bright* would be correctly classified as *bright*. If in the next cycle a new class *dark* is added containing an inverted similarity to *bright* the same object would be classified as *dark* since the inverted similarity produces a membership value of 0.75. If you want to specify that *dark* is everything which is not classified as *bright* you should use the feature “classified as.”


Similarities are inserted into the class description like any other expression.

## The classification process

Coming back to the example given at the introduction to the classification, the classification process can be compared to a database query. Each object is compared to each class description. Its contained and inherited expressions produce membership values for each object and according to the highest membership value, each object is then labeled or classified. This process of labeling objects to either one or the other class and thereby producing definite class assignments from membership values is called defuzzification. For additional information see Concepts & Methods.

A classification demands at least one level of image objects and a class hierarchy containing edited class descriptions. You can produce this class hierarchy or load an already existing one. Throughout the classification process each class description in the class hierarchy is applied to each image object of one or of all levels in the image object hierarchy.

An image object is assigned to the class whose evaluation returns the highest membership value. If the membership value of an image object is lower than the predefined minimum membership value, the image object will remain unclassified. This is done to ensure a certain reliability for your classification. The necessary reliability will depend on your application. You can edit the minimum membership value under “Classification > Advanced Settings > Minimum Membership Value.” If two or more class descriptions share the highest membership value, the assignment of an object to one of these classes happens arbitrarily.

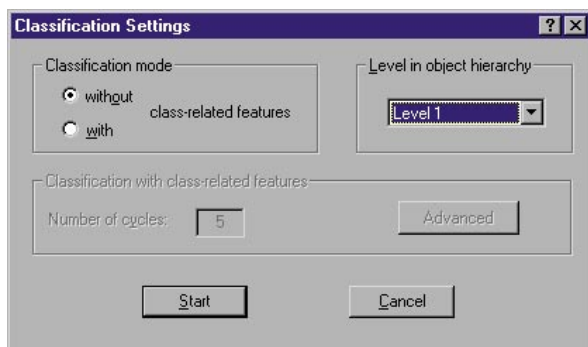
The classification process is started by selecting the menu item “Classification > Classify...” This opens the dialog box “Classification Settings.” Here you can edit the classification parameters and subsequently start the classification process. Additionally to the classification menu, there is also a compressed version of the classification menu available as icons on the menu bar . For a further detailed description see the chapter “User Interface.”

The level box of the classification dialog allows you to select a level in which to perform the classification. When you open the dialog box “Classification Settings,” the level displayed under “Level in object hierarchy” will automatically be synchronized with the one given in the active view.

## Classification without class-related features

This is the method of choice if you are not using any class-related features. In this case all classes whose class descriptions include class-related features are treated as inactive classes.

The only additional setting to make is the correct image object level to be classified. Click “Start” to get the actual classification process going.



### How to handle class-related features

The use of class-related features is explained in detail in the next chapter, “Classification Advanced.” Here only the information necessary to understand the impact on the classification procedure is given.

There are several basic kinds of dependencies between classes. The simplest is when one class is entirely defined by membership functions or a nearest neighbor classifier and another class refers to this class by means of class-related features. The first class in this case could be called grounded *G*, because objects of this class are securely defined and not subject to changes if the local context changes.



A second case could be one grounded class *G* to which a row of classes, in which one class depends on the next, refers. These two cases are rather easy to handle. In the first case one or in the second case several cycles of classification should suffice to produce a stable result. When working with class-related features you should always try to construct dependencies like the ones described.



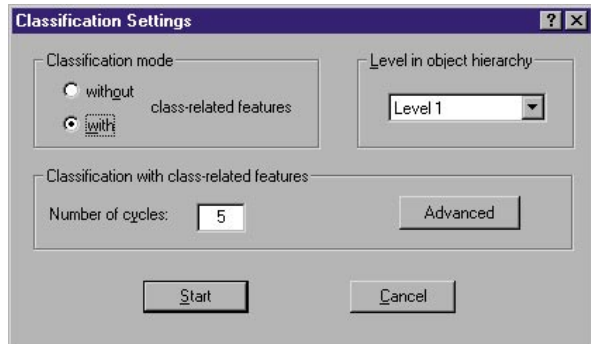
However, in some cases this may not be possible due to the complexity of the classification task. In such cases circular dependencies may occur. Circular dependencies mean that one class refers to another, which again refers to this class. Whenever you have this kind of dependencies in your class hierarchy you should make use of the simulated annealing.



## Simple classification with class-related features

This is the method of choice when your class descriptions include class-related features. In the field “Classification mode,” enable classification with class-related features. Again, the image object level to be classified has to be selected. Note that before you classify with class-related features, you have to perform a classification cycle without using the context.

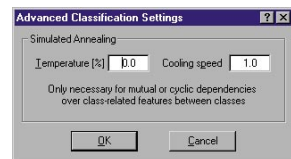
Since the assignment of image objects, especially with the use of neighborhood relations, strongly depends on the initial conditions concerning the classification of other image objects, one cycle might not be sufficient to yield a stable classification result. Therefore, you can define the number of classification cycles in the field “Number of cycles.”



## Classification with class-related features and simulated annealing

In some cases a high number of cycles will not result in a stable classification. More than likely, this is due to mutual relationships and cyclic dependencies in your class descriptions. A simple example would be a *class a*, which is described by its relative border to *class b*, which in turn is described by the relative area of *class a* in a certain vicinity. Such dependencies occur especially in large class hierarchies with a great number of class-related features. To handle this, eCognition provides a simulated annealing approach (see “Concepts & Methods”) in its advanced classification settings. To open the dialog, select “advanced” in the classification dialog.

Two parameters can be set: “Temperature,” which is the amount in which random change of the membership values of the image objects takes place, and “Cooling speed,” which determines how much the initial random deviation defined by temperature is reduced at each classification cycle. Their adjustment is only necessary if you have incorporated cyclic dependencies between classes containing class-related features. Otherwise, just use the default settings.

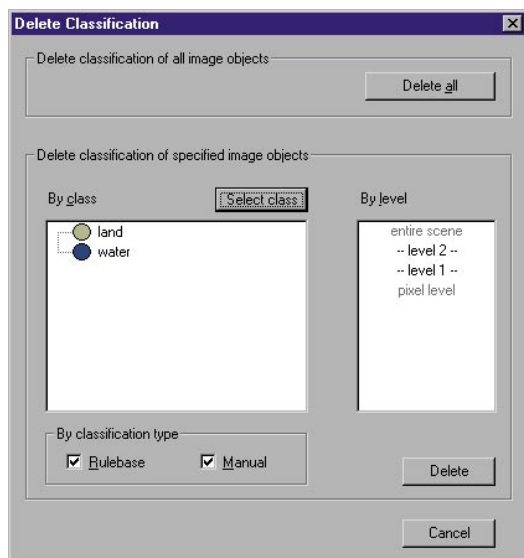


## Deleting classification results

The dialog “Delete Classification” allows deletion of the results of a classification. The dialog opens by selecting the menu item “Classification > Delete Classification.” This dialog offers different possibilities to delete classification results.

To delete the entire classification click “Delete all.”

To delete only parts of the classification, select the class/classes and/or the level for which to delete the classification. Furthermore, the type of classification can be determined. To delete only the results of a manual classification, select “Manual.” Otherwise select “Rule base” or both. The “Delete” button will execute the action.





## Classification Advanced

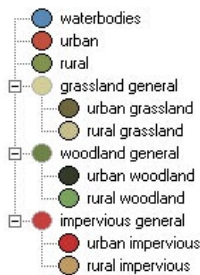
### The class hierarchy

The class hierarchy is the frame of eCognition's language to create the knowledge base for a given classification task. It contains all classes and is organized in a hierarchical structure. How to technically create a class hierarchy is described in the chapter "Classification Basics." This part of the user guide explains advanced usage of the class hierarchy.

The class hierarchy distinguishes between passing down class descriptions (from parent classes to their child classes) in the inheritance hierarchy and the meaningful semantic grouping of classes in the groups hierarchy. The purpose is to reduce redundancy and complexity in the class descriptions, on the one hand, and to create a meaningful grouping of classes, on the other, e.g., turning a mere land cover classification into a land use classification.

### Inheritance

The inheritance hierarchy refers to the physical relations between the classes. Here, the class descriptions defined in parent classes are passed down to their child classes. This aspect helps to differentiate a parent class, for example representing a specific land cover, into two child classes representing land use classes, depending on the embedding within a certain context. It helps to considerably reduce the necessary number of inputs in the class descriptions.



This class hierarchy is an example of how to apply the inheritance hierarchy. The land cover classes *woodland general*, *grassland general* and *impervious general* (parent classes) are further divided into land use classes (child classes). The two classes *urban* and *rural* do not contain any expressions. Their meaning is defined in the groups hierarchy.

If you want to formulate the class description of a general class, you do this once in the relevant parent class in the inheritance hierarchy which will pass this change down to all child classes of the pertinent parent class. Without the inheritance concept you would insert the same description in several classes, losing time, transparency and flexibility in creating your knowledge base.

The inheritance of class descriptions contains an implicit logic concerning the application of classes, i.e., whether or not classes are actually assigned:

**1. Classes are not applied to the classification of image objects whenever they contain applicable child classes within the inheritance hierarchy.**

Parent classes pass on their class descriptions to child classes. These child classes then have additional feature descriptions and – if they are not parent classes themselves – are meaningfully applied during the classification of image objects. The above logic follows the concept that child classes are used to further divide a more general class. Therefore, when defining subclasses for one class always keep in mind that not all objects defined by the parent class are automatically defined by the subclasses. If there are objects which would be assigned to the parent class but do not meet the descriptions of the subclasses, they will be assigned neither to the parent nor to the child classes.

**2. Classes are only applied to a classification of image objects if all contained classifiers are applicable.**

The second rule applies mainly to classes containing class-related features. The underlying reason is that you might generate a class which describes objects of a certain spectral value in addition to a certain contextual information given by a class-related feature. The spectral value taken by itself without considering the context would cover far too many objects, so that only a combination of the two would lead to satisfying results. As a consequence, when classifying without class-related features, not only the expression referring to another class, but the whole class, is not used in this classification process.

**3. Classes described by nearest neighbor with class-related features**

When using class-related features to define the feature space for the nearest neighbor classifier, circular dependencies between the classes must not occur. Consequently, all features which refer to the class itself directly or indirectly are not allowed for the definition of the feature space. Thus, nearest neighbors with a class-related feature space cannot be passed on to child classes. Hence, only child classes on the lowest level of inheritance (leaf classes) can be described by a class-related nearest neighbor classifier.

**4. Samples of child classes are simultaneously samples of their parent classes**

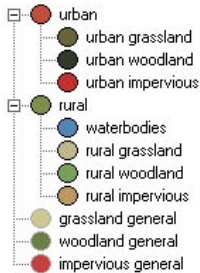
Since child classes are similar to their parent classes by inheritance, all parent classes must have the same samples as their child classes. Especially for classifications with class-related nearest neighbor this constraint is vital. Otherwise leaf classes in the inheritance class hierarchy, which are only described by class-related nearest neighbor, could never be applied. Whenever you take a sample for a child class this sample will be copied to all its parent classes.

## Groups

The groups hierarchy refers to the semantic relations between classes. The child classes of parent classes in the groups hierarchy enable you to group classes, which can even contain very different feature descriptions, to an identical superior semantic meaning. Functionally this grouping has two consequences:

1. **Class-related features only refer to classes (and child classes) in the groups hierarchy. So, when using relationships to a parent class in the groups hierarchy this also applies to the respective child classes.**
2. **You can navigate the classification view through the levels of the groups hierarchy and thereby visualize the semantic grouping of the classes.**

Often the physical features of objects do not represent their semantic meaning. Therefore, the groups hierarchy allows you to group classes semantically. So when classifying with class-related features, you can refer to groups of classes which belong to one semantic entity by referring to their parent class in the groups hierarchy.



This image shows the same class hierarchy as the one displayed in the paragraph about the inheritance hierarchy. This time the groups hierarchy is displayed. The two classes *urban* and *rural* have an empty class description, i.e., they only act as a general semantic group. They do not pass on any information to their subclasses but only summarize all subclasses semantically.

As in the given example, objects classified as *woodland* after an initial land cover classification might be divided into woodland areas in a rural and in an urban environment. The approach would be to subdivide this class into two child classes *urban* and *rural woodland* within the inheritance hierarchy. But to be able to represent the semantic meaning at the same time, two classes (*urban* and *rural*) could be generated to function as groups for all urban and rural classes within the groups hierarchy.

## The interrelations between inheritance and groups

The usefulness of the class hierarchy is based on the synergy of the inheritance and groups hierarchies. Use the inheritance hierarchy to formulate class descriptions and pass down general class descriptions to child classes. Use the groups hierarchy to subsequently combine child classes into parent classes of a general semantic meaning. The result will be an elegant, easy to correct structure of classes.


In many cases, child classes in the inheritance hierarchy are also child classes in the groups hierarchy. The inheritance hierarchy is a hierarchy of similarities concerning the feature descriptions – child classes of the same parent class inherit the same class description, i.e., they are similar. The groups hierarchy, however, is a hierarchy of similar semantic meaning. The following three rules determine the relation between groups and inheritance hierarchy.

- **Classes are not applied to the classification of image objects whenever they contain applicable child classes within the inheritance hierarchy**
- **Classes are only applied to a classification of image objects if all contained classifiers are applicable**
- **Class-related features only refer to classes (and child classes) in the groups hierarchy. So, when using relationships to a parent class in the groups hierarchy, this also applies to the respective child classes**


Sometimes it might be useful to introduce a separate class to combine classes in the groups hierarchy solely to relate a class-related feature to this group. This class may function exclusively as a group without having a class description.

The interrelation between inheritance and groups is also displayed visually within the class hierarchy. When a class is not applicable, its color dot is displayed with a gray outline. If the class is applicable, the outline is black. When switching the classification settings from “without class-related features” to “with class-related features” the change of the outlines indicates which classes will be used depending on the classification settings. In the example below, the two child classes contain class-related features and the parent class contains only object features. Therefore, only the parent class is active without class-related features, while only the child classes are used when classifying with class-related features.



 class hierarchy with the classification settings set to “without class-related features”



 class hierarchy with the classification settings set to “with class-related features”

## Editing the hierarchical structure

The creation of the hierarchical structure is easy. Just drag and drop:

To define a class as a child class, left-click it and, while holding down the mouse button, drag it over the class to be defined as the parent class and drop it.



To move a child class from one parent class to another, left-click it and drag it to the new parent class.



To redefine a child class as a class without a parent class, left-click and drag it below the list of classes and drop it there.



## Multiple inheritance, multiple membership

A class can be the child class of more than one parent classes in the class hierarchy (both inheritance and groups hierarchy). This leads to the inheritance of class descriptions from more than one parent class to one and the same child class in the inheritance hierarchy. In the groups hierarchy it means that the child class is a member of more than one semantic group. To assign the same child class to a second parent class, use the right mouse button to drag and drop it to the new additional parent class. The

**Note!** If a class is defined as a child class of more than one parent class in the groups hierarchy, you have to decide which of these parent classes to display when navigating upwards through the levels of the groups hierarchy in the classification view. To do so, select the correct parent class to be displayed under “Parent class for display” in the class description dialog of the specific child class.

same class will now appear twice, indicating the child class's membership to the respective parent class each time. However, double-clicking one of each of the symbols will always open the same class description.

To remove a class from a certain parent class, left-click the specific symbol, drag it below the list of classes and drop it there.

### Basic strategies for creating class hierarchies

The classification process in eCognition is supervised, allowing you to train the system by introducing sample objects (nearest neighbor) or classification concepts (membership functions and logic combinations). For most applications, you will typically find different ways of coming up with a classification result. The most important classification approaches are discussed in this section. However, none of these proposed techniques has necessarily to be applied in its pure form. Feel free to add functionality to each of them or combine them.

Always keep in mind that the main goal in defining class descriptions is the separation of all classes in your classification scheme. Try to find solutions that are as simple as possible! The simpler a class description is, the more transparent the evaluation of image objects is and the easier you can adapt it to come up with a better classification result.

A crucial issue in building up your knowledge base is achieving information about image objects, features and classification. For detailed information see [Functional Guide > Information on Image Objects and Features](#).

Have a look at the following classification techniques:

- Overview: how to acquire information
- Fast nearest neighbor classification by manual training: click and classify
- Using a fuzzy rule base and membership functions to formulate classification concepts
- Differentiating classes using contextual information (class-related features)
- Masking techniques

## Overview: how to acquire information

eCognition offers a number of tools that help to find out how classes can best be described and separated. You will find in the following a short overview of the tools and the related questions you might have.

### "Image Object Information" dialog

#### Acquiring information about the attributes of a specific image object

The image object information dialog serves an important purpose: Developing class descriptions becomes much easier after computing an object's features in the dialog. Transparency and accessibility of information are the two main reasons why you should regularly use this function. The displayed class evaluation and feature values can help to achieve better classifications.

#### Acquiring the detailed evaluation of any chosen class description for a specific image object

This information can be obtained for the selected object in the image object information dialog. Apart from information about the object attributes, you can obtain detailed information about the actual classification of the image object. Click the class of your interest in the field "Alternative Assignments" to get a detailed evaluation of the image objects attributes for the chosen class description.


### Feature view

#### Obtaining an overview and intuitive access to the effect of any chosen feature over all image objects in a scene

The feature view allows you to obtain an overview of any feature's values over all image objects in a scene. Each image object is displayed in a gray value or a defined range from blue (low values) to green (high values) and represents a feature value. Use the feature view to determine features by which two classes can be distinguished.

## Sample editor, sample navigation and sample assessment

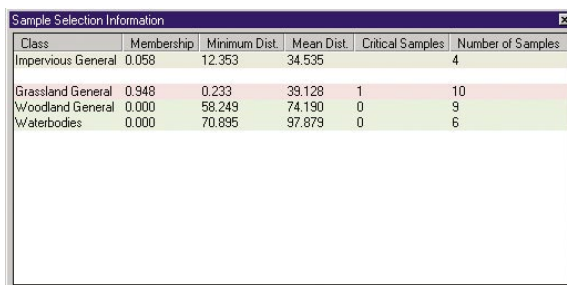
### Collecting samples of similar characteristics

The sample editor is typically used to assign image objects as samples of a certain class and to compare them as to their feature values. It displays the feature signature of the samples of a class. Thus, by marking a new image object you can compare its feature values with the signature of the already trained sample objects. To find the token samples in the image, you can activate the sample navigation by clicking the  button. If this mode is active, click on a slot in the sample editor and you are automatically navigated to the appropriate sample. The respective sample will be highlighted in the image view. If a slot holds more than one sample, you can select one of them from the appropriate pull down menu:



### Assessing the quality of a new sample

If classes already have samples you can assess the quality of a new sample in the Sample Selection Information dialog:



Class	Membership	Minimum Dist.	Mean Dist.	Critical Samples	Number of Samples
Impervious General	0.058	12.353	34.535		4
Grassland General	0.948	0.233	39.128	1	10
Woodland General	0.000	58.249	74.190	0	9
Waterbodies	0.000	70.895	97.879	0	6

Select the appropriate class you want to take a sample for in the class hierarchy. Then click on a potential sample and analyze its distance to the mentioned class and to all other classes within the feature space. In the dialog three distances are shown:

- **Membership:** shows the potential degree of membership according to the adjusted function slope of the nearest neighbor classifier.
- **Minimum Distance:** shows the distance in features space to the closest sample of the appropriate class.
- **Mean Distance:** shows the mean distance to all samples of the appropriate class.

The column “Number of Samples” shows how many samples are already present for the appropriate class, while “Critical Samples” shows, to how many samples the potential new sample is critically close. The respective row will then be highlighted in red,



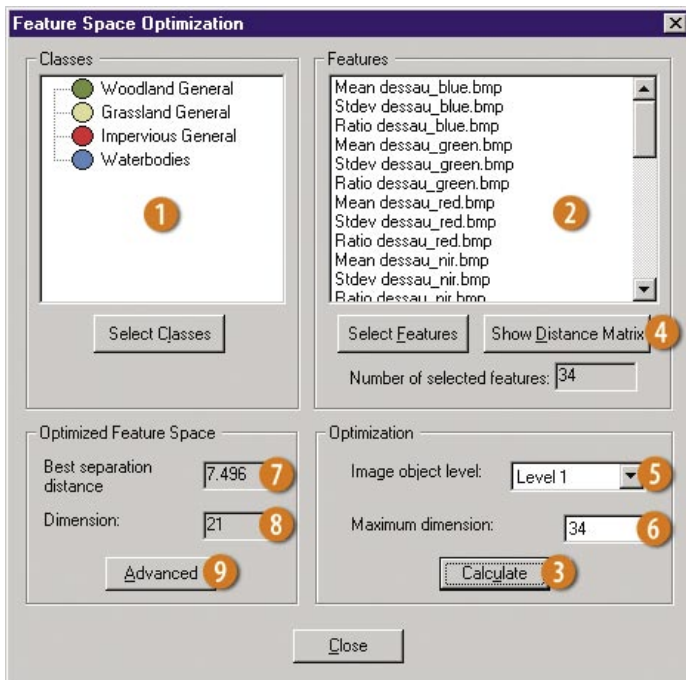


while all noncritical classes are shown in green. To define what 'critical' means you can adjust the minimum allowed membership to other classes by right-clicking the dialog and selecting the appropriate menu entry:


**Note!** you can navigate to all samples by choosing the appropriate class, then switching on the sample navigation mode and selecting either the critical, nearest or current sample(s).

### Finding the features that best separate the classes

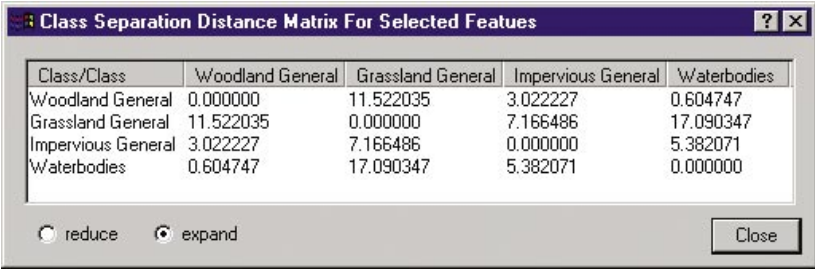
The sample editor allows not only the display of feature histograms for sample objects, but also for all image objects in a scene. This can be used to get an overview of the distribution for each features. To find well distinguishing feature combinations for your classes, you have several possibilities:



Collect typical sample objects for each class and compare the histograms for arbitrary features in the sample editor. Thus, you acquire a clear overview and quantitatively precise information about the separating capability of features. Note that if two classes cannot be separated by one single feature, these classes are not distinguishable using membership functions. However, they might be distinguishable by using nearest neighbor classification since nearest neighbor separates the distributions of classes in a high dimensional feature space far better.

For an automated quest for the best feature combination you can use the “Feature Space Optimization” tool: For all classes you want to distinguish collect several samples. Then choose “Feature Space Optimization” by clicking the  button. The upcoming dialog asks you for the classes **1** you want to distinguish and the feature space **2** you want to reduce to an optimum dimension:

**Note!** it is not possible to use class-related features in the feature space optimization.



The dialog box titled "Class Separation Distance Matrix For Selected Features" contains a table with the following data:

Class/Class	Woodland General	Grassland General	Impervious General	Waterbodies
Woodland General	0.000000	11.522035	3.022227	0.604747
Grassland General	11.522035	0.000000	7.166486	17.090347
Impervious General	3.022227	7.166486	0.000000	5.382071
Waterbodies	0.604747	17.090347	5.382071	0.000000

At the bottom of the dialog, there are two radio buttons: "reduce" (selected) and "expand". A "Close" button is located at the bottom right.

To analyze the separability of the classes with choosable features, select the features by single-clicking them, then click the „Calculate“ **3** button and show the distance matrix of the classes by clicking the appropriate button **4**.

Based upon the distance measuring as described in [Concepts & Methods > Fuzzy classification in eCognition > Creation of conditions in multidimensional feature space by nearest neighbor classification](#) the distances in the features space of all classes are shown. You will realize that the matrix is symmetric, which is due to the fact that each class is compared to each other. In the example above the class Waterbodies is hardly separable from Woodland General.

**Note!** the distance calculation is only based upon samples. Thus, adding or deleting samples also affects the separability of classes.

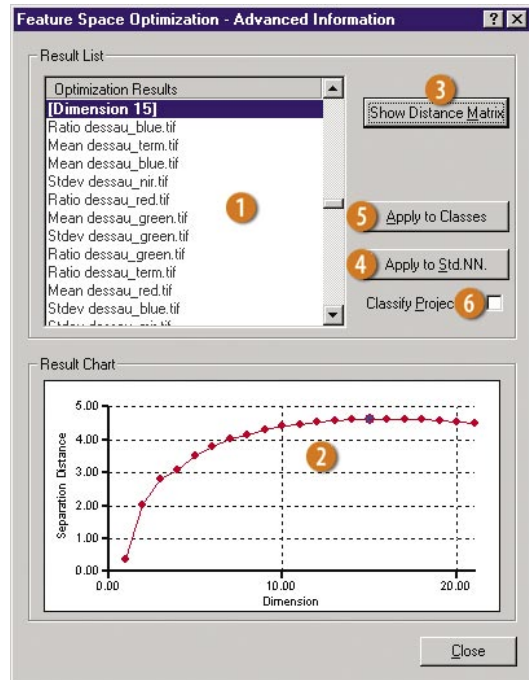
In the field “Optimization” you can select the segmentation level **5** on which the optimization is to be calculated and you can enter the maximum dimension (the number of features) to which the feature space should be reduced/optimized **6**. The program then performs a number of feature permutations and calculates for each feature combination the distances of the classes within this feature space.

**Note!** the higher the number of features in the beginning and the higher the chosen maximum dimension, the longer the calculation takes. Thus, before calculating the optimal feature space you should check whether a selected feature can contribute to distinguishing your desired classes or not. CPU-demanding features like “Texture after Haralick” can expand the calculation time dramatically. If one or more features have no values you will be informed.

After calculation has finished you will see in the field “Optimized Feature Space” a value for “Best separation distance” **7** and “Dimension” **8**. The latter shows the (reduced) number of features to span the feature space, while in the field “Best separation distance” the distance between the closest samples of the chosen classes in this feature space is shown. Clicking “Advanced” **9** will bring up the following dialog:

In the „Result List“ **1** all feature combinations and their according distance values for the closest samples of the classes are displayed. The “Result Chart” **2**

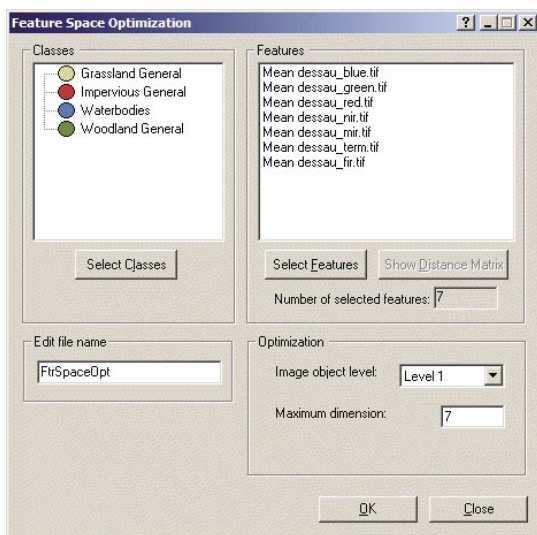
shows the same graphically. The blue marked combination is by default the one with the highest distance between the closest samples. By clicking on another dot in the graph you are automatically navigated to the according feature combination in the “Results List” and vice versa. Clicking the “Show Distance Matrix” **3** button brings



up the distance matrix for the selected feature combination. You can use this to check the distances of samples for all other classes to each other. This is useful, if you want to find out if there is another feature combination which might separate other classes better, although the best separation distance is lower. To update the current distance matrix, you have to click “Show Distance Matrix” again.

“Apply to Classes” ⑤ gives you a choice to select classes to which a nearest neighbor classifier, whose feature space uses the currently selected feature combination, is to be added. Note! all other already existing features or nearest neighbors in the classes will remain. If you click on “Apply to Std.NN.” ④, the current feature combination will be used as the feature space for the standard nearest neighbor classifier. If you mark “Classify Project” ⑥, your project will be classified automatically with the generated nearest neighbor or the updated standard nearest neighbor classifier, when applying them.

You can export the results from the Feature Space Optimization by selecting from the main menu “Classification > Nearest Neighbor > Export FSO ...”. A dialog analogous to the Feature Space Optimization dialog will come up:



The usage of the dialog is more or less similar. In the section “Edit file name” you are asked for a prefix of the exported files. After clicking on OK two \*.csv files will be created:

<prefix>\_comb.csv and

<prefix>\_matrix.csv. The first contains all feature combinations and in the first column the one with the best separation is indicated. The second file contains the distance matrix for the best separating feature combination as it is given when clicking on “Show Distance Matrix” in the Feature Space Optimization dialog.

## 2D feature space plot

**Comparing the attributes and the distribution of all image objects and samples in a two-dimensional feature space**

The 2D feature space plot displays the interrelation of two features in a 2D scatter plot. Each image object of the project is included, and those assigned as samples are highlighted. Using this view you can obtain information about the correlation of two features.

## Getting the spectral histogram of an image layer

Use the option „Tools > Layer Histograms“ to obtain information about the distribution of the spectral values within the different image layers. The blue marked slots indicate significant values, while the red marked slots show values of relatively less frequent occurrence.

## Fast nearest neighbor classification by manual training: click and classify

Using the nearest neighbor as a classifier allows fast and easy classification results. The nearest neighbor is superior to a combination of membership functions when it comes to describing a multidimensional feature space. It automatically models multidimensional membership functions. Therefore, it is better to handle correlated as well as non-continuous and non-Gaussian distributions in the feature space.

Training for the nearest neighbor is performed by labeled samples. Thus it is similar to training areas using supervised classification in conventional pixel-based image analysis systems. However, eCognition uses image objects and not only single pixels as training areas. Class assignments of image objects are determined by their distance to sample image objects in a predefined feature space. These samples need to be declared as typical representatives of the respective class during the training of the system. In eCognition, such typical representatives are referred to as sample objects.

## Sample declaration and classification in iterative steps

The nearest neighbor classifier produces results very quickly and can be easily improved by iterative steps. For training initially declare a small number of sample objects for each class. However, try to cover the typical, different appearances of each class by the choice of samples. Having done so, start the classification process (without the use of class-related features). This first classification will yield many correctly classified image objects, but also a number of unclassified and incorrectly classified ones.

In repeating steps, do the following to improve the classification quickly:

- Assign one or few typically wrong or not classified image objects as sample objects to the right class. Be sure to select samples carefully.
- Repeat the classification.

You will notice that the classification result will improve from step to step. Repeat the described steps, until you obtain a satisfactory result.

Sometimes the selection of a new sample object might change the classification result significantly, causing more wrongly classified image objects of another class rather than correcting the assignments of the class of concern. Do not worry and go on as before: in subsequent steps, assign wrongly classified image objects, this time to other classes. Continuing like that, you are differentiating the borders of the class distributions in the feature space, describing even irregular or noncontinuous distributions.

You can use the Sample Editor to check the selected sample histograms for each class and possible overlaps to other classes (of course, check the histograms of the features that you used for the definition of your feature space). In the event that you have selected a wrong sample you can easily deselect it by double-clicking it another time. Under “Samples > Delete all Samples” you can delete all samples, or under “Samples > Delete Samples of Classes...” only the samples of a specific class.

Using the described technique, it is possible to quickly achieve a good classification result. Starting with a few sample objects and adding further, necessary samples in subsequent steps is a very efficient procedure. It is supported by the advantageous characteristics of the nearest neighbor classifier: it does not rely on a continuous, Gaussian distribution and is able to detect even complexly shaped distributions in the feature space exactly.

At the end you might have reached the limits of the chosen feature space. Further assignment of samples does not result in significantly better differentiation and classification results. However, you can use the achieved knowledge base as a starting point for further differentiation, using for instance contextual features in child classes in the inheritance hierarchy.

As an example for a typical nearest neighbor classification, see the guided tour “LANDSAT TM subset of Orange County (California).”

## Using fuzzy rules to formulate classification concepts

Fuzzy rules allow you to formulate criteria such as “all image objects darker than a certain value for brightness are shadow.”

Technically, a classification with the use of a fuzzy rule base is done by finding out which combination of fuzzy features is suitable to distinguish one class from the others. First you have to find out which features have a correlation with the desired output class. (Use the feature view to get an insight in your data and detect a correlation between the feature range and the desired classification output.) After that, insert the respective feature into the class description and edit the membership function. As there are many cases in which one single feature is not sufficient to fully describe a class, a class description can contain a certain number of features along with their membership functions. All these features can be combined by one or more logical expressions. The whole condition will then be evaluated throughout the rule base and determine the fuzzy classification result.

There is a very clear and transparent relationship between the parameters and the shape of a membership function and the detailed evaluation of a membership function concerning a specific image object in the dialog box “Image Object Information.” These settings can easily be adapted to the needs of a classification. The use of a fuzzy rule base therefore leads to transparent, adaptable and more objective classification results than the use of the nearest neighbor, since the outcome of the classification does not depend on the sample objects chosen. If there is one feature or a small number of features which allow the clear separation of a class from all others, the fuzzy rule base is therefore the best choice.

However, when using many features for the description of one class, the one-dimensional membership functions available in eCognition are inferior to nearest neighbor, since they cannot handle overlaps in the feature space as well as the nearest neighbor. There is also another danger: especially in high-resolution data, classes can be very heterogeneous and therefore very difficult to describe. Finding class rules for them might lead to a very complex and difficult to understand set of class rules, which are, in addition, likely to become unstable and highly specific.

### **Differentiating classes using contextual information (class-related features)**

As soon as an initial classification of image objects has been performed it is possible to use class-related features for a further classification, coming up with a more differentiated result including contextual information. Using class-related features means relating the classification of image objects to the classification of other, networked image objects. Class-related features can only be inserted into class descriptions by means of membership functions.

The sequence of classification cycles when using class-related features is always the same:

1. a cycle “without class-related features,” coming up with an initial classification.
2. a cycle “with class-related features” which refer to the initial classification.

Thus, when no initial classification result is present, a non-class-related classification will always be performed first automatically.

Remember the implicit logic of the inheritance hierarchy: a class is only applied for classification when

1. it has no child classes that are applied and
2. all its contained features can be evaluated.

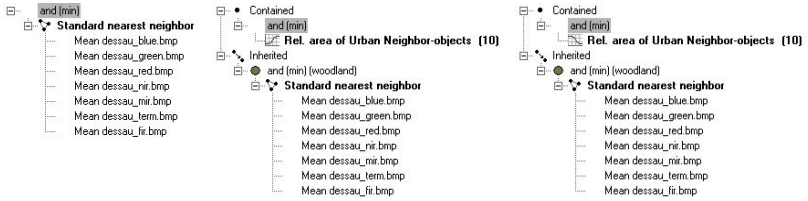
Class-related features are therefore best used in child classes (inheritance hierarchy). Because a class is only applied to classification when all contained features are applicable, a class will not be applied in an initial classification cycle “without class-related features” when it contains class-related features.

In many cases it therefore makes sense to describe one class only by means of objects features and pass on its class description to child classes. In the example given below the class *woodland* has two subclasses, *urban woodland* and *rural woodland*, to differentiate the class with regard to its contextual embedding. In the initial classification cycle “without class-related features,” only *woodland* will be automatically applied for classification, as its contained features, and none of its child classes’, can be evaluated. In the subsequent classification cycle “with class-related features” the classes *urban woodland* and *rural woodland* can be applied. As they are child classes of *woodland*, *woodland* itself will not be applied any more.

The advantage of this concept is that in this case the spectral features are analyzed separately from the contextual features. So if the input data and with it the spectral values change, the classes addressing the spectral values can be adapted while the contextual rules stay the same.

**Note!** If you want to relate features to the class *woodland* including *urban woodland* and *rural woodland*, *urban woodland* and *rural woodland* must be child classes of *woodland* in the groups hierarchy, too.



*woodland**urban woodland**rural woodland*

### Using class-related features with nearest neighbor

When using a nearest neighbor classifier with class-related features for a class description, samples for the appropriate class are needed. To avoid cyclic dependencies between the classification and the samples, the class-related values of the samples always refer to the last classification result without class-related features. If a classification result is not yet present, a prior non-class-related classification is performed automatically. The samples will take their values for class-related features of the last non-class-related classification result.

In order to keep classification results stable, non-deterministic classification results have to be avoided. Classification results become non-deterministic if in the class hierarchy a cyclic dependency is present. The simplest form of a cyclic dependency is the case that a class description refers directly to the class itself:

*woodland general*

**Note!** the example above is only possible when using class-related features as membership functions. Keep in mind, that the classification result for this case will change after each classification.

Indirect cyclic dependencies are more complex to handle and sometimes not easy to detect. Mostly they occur when expressing similarities. In eCognition two basic forms of similarity exist: physical similarities (feature “similarity to” and similarity by inheritance) and semantic similarities (groups hierarchy). Thus, to prevent nondeterministic

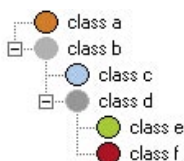
classification results when using class-related features in nearest neighbor, several constraints have been introduced:

- it is not possible to use the feature “similarity to” of a class which is described by a nearest neighbor with class-related features.
- classes cannot inherit from classes which use a nearest neighbor that contains class-related features. Only leaf classes in the inheritance class hierarchy can use class-related features in a nearest neighbor.
- it is impossible to use class-related features which refer to classes in the same group including the group class itself.

If you should violate one of the above constraints while creating a class hierarchy or class description, you are informed and the desired operation is not executed.

## Masking techniques

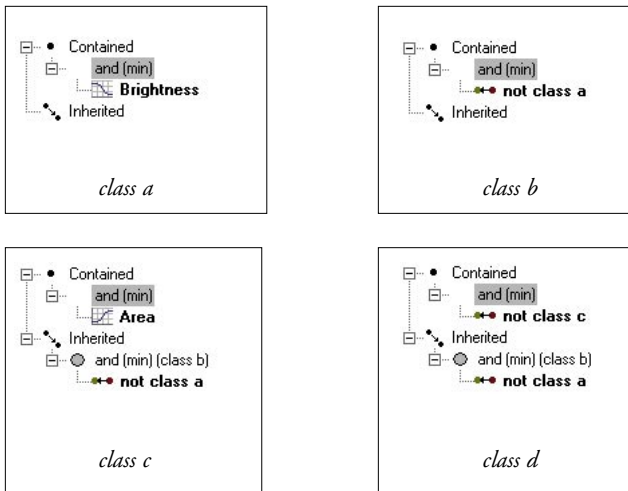
eCognition offers an effective and easy to handle masking technique using the fuzzy rule base. In many cases some classes of a classification scheme are relatively simple to describe, whereas others need more effort. When creating a class hierarchy it can be useful to begin with the class descriptions of clearly distinguishable classes. These classes mask the already classified areas. Using inverted similarity features an additional class covering all unmasked regions can be created. The class description of the unmasked class can be inherited to child classes, allowing further differentiation. For this purpose, features different from those used in the parent classes can be useful. This procedure can be repeated, resulting in a hierarchical tree structure of classes in the inheritance hierarchy. This structure contains easily distinguishable classes on higher levels and passes on more complex classifications to levels further down in the hierarchy.

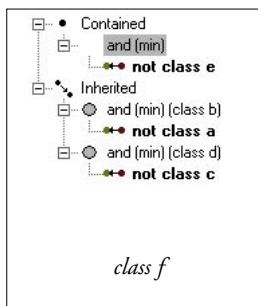
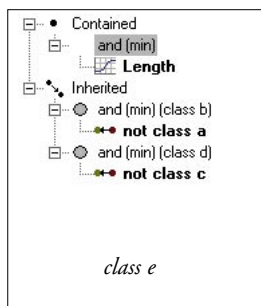


Start with classifying all image objects that belong to *class a*. Then create a new class to cover all the remaining unclassified image objects, for instance *class b*. Insert the inverted similarity “not *class a*” into the class description. Create *class c* as a child class of *class b*. This way, the feature “not *class a*” will be inherited. Now the class description of *class c* can be edited for further differentiation. Repeat the described procedure by introducing a further *class d* with the inverted similarity “not *class b*.” The next child class, *class e*, contains by inheritance the class description that it is not *class a* and not *class c*. This procedure can be repeated until there is no reason or there are no possibilities for further differentiation using membership functions.


Instead of using similarities, the feature “classified as” can also be used in an inverted way, specifying that all objects which are not classified as *class a* are to be *class b*. The difference between similarities and the “classified as” feature is that the “not classified as” features are more rigid and do not make use of fuzzy logic – they sort of “freeze” the classification.

As an example, an object with a low membership to *class a* (like 0.2) will be classified as *class b* with a membership value of 0.8 if similarities are used. The image object remains classified as *class a* when the “not classified as” feature is used in *class b*. The reason is as follows. The inverted similarity to *class a* will produce the opposite membership value of the one for *class a* (i.e., 0.8), therefore, the object will be labeled as *class b*. The expression “not classified as *class a*” used for *class b* will result in a membership value of 0 because the membership value of 0.2 for *class a* was sufficient to classify the object as *class a*.





### Creating customized features

The “Create Customized Feature” tool allows you to create your own features. It includes two different tools which enable you to create arithmetic on the one hand and relational features on the other. All calculations are based on the existing features of eCognition. Customized arithmetic or relational features cannot be included in other customized features due to the resulting complexity. To open the “Create Customized Features” dialog, select “Tools > Customize Features...” or click the respective button  in the menu bar.

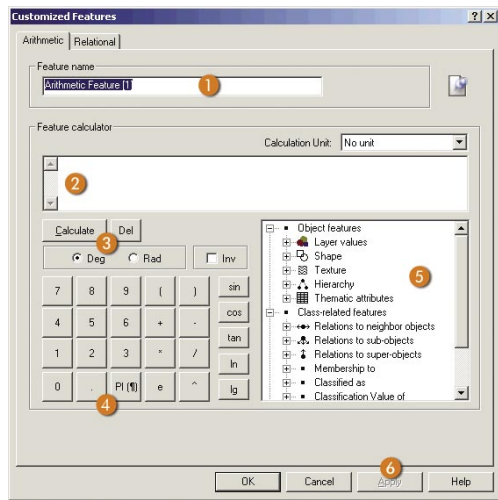
Newly created features can be found in the “Insert Expression” dialog under “Customized.” To edit a customized feature, right-click the respective feature and select “Edit Feature...” To delete the feature, select “Delete Feature.”

## Arithmetic features

This tool (“Arithmetic”) supports the combining of features using basic arithmetic operations. This way new features can be created by combining existing ones. The newly created features can be, e.g., simple ratios or sophisticated calculations. The “Feature name” line ❶ contains the name of the new feature. It can be edited to create individual feature names. The feature calculator below is divided into three parts. The top window contains the actual calculation ❷, the number pad allows you to insert numbers and arithmetic operators ❸, while the right window contains the features ❹.

To insert a feature into the equation, double-click it. To delete parts of the calculation, use the delete button. The calculator also allows you to calculate parts of the equation right away. Select the part which is to be calculated and click “Calculate” ❸. Click “Apply” ❹ to generate the new feature.

There are only a few restrictions on what can be formulated by a new feature. Therefore, the responsibility lies with the user to ensure that the feature makes sense. Invalid operations such as division by 0 will result in an invalid value for this newly created feature.

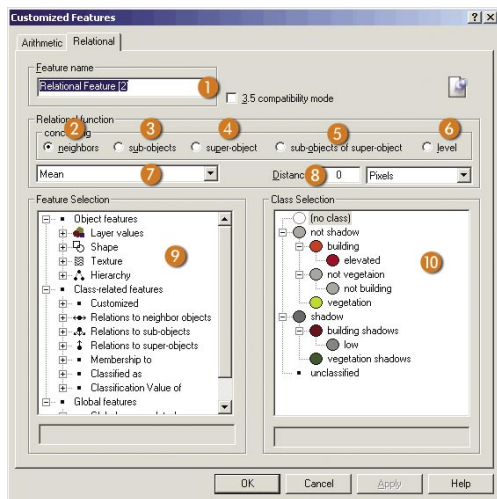


### Examples for arithmetic features:

- All kinds of spectral indices (like NDVI)
- All kind of shape indices
- Normalized features (standard deviation in relation to object size)
- Relation of distance to *class a* to distance to *class b*
- Relation of border to *class a* to border to *class b*

## Relational features

In the relational features register („Relational“) you can create features which compare an object's feature to the features of other objects. You can refer to surrounding objects as well as to sub-objects, a super-object, sub-objects of a super-object or a complete level. Furthermore, you can compare an object's feature to either all surrounding (sub-, super- ...) objects or to surrounding (sub-, super- ...) objects of a certain class.



When creating a new feature, define the name in the “Feature name” line 1. The relational function determines what kind of relation is used. According to whether surrounding objects 2, sub-objects 3, a super-object 4, sub-objects of a super-object 5 or the complete level 6 are used, different relations as stated below can be selected from the drop-down menu 7. In the “Distance” input window 8, the distance can be defined. As mentioned above, distance refers either to the number of hierarchy levels within the object hierarchy or to the distance within one level.

The “Feature Selection” window 9 allows you to determine which feature to compare. Simply click the desired feature and it will be inserted in the selection line below the window. The “Class Selection” 10 contains all classes of the class hierarchy. Select a class to which the relation is to be evaluated by clicking it. If the relation should not be limited to a certain class, select “(no class)”.

**Note!** As with class-related features, the relations refer to the groups hierarchy. This means that if a relation refers to one class, it automatically refers to all sub-classes of this class in the groups hierarchy.

The kind of relations you can evaluate are given in the following.

Relations to “neighbors”, “super-objects”, “sub-objects of super-object”, “level” or “sub-objects” target either directly adjacent objects or, when a distance is defined, all objects within a certain radius or level. It is important to note that the values of each object are weighted by the object size. The operations which can be performed are listed below.

- mean
- standard deviation
- mean difference
- mean absolute difference
- ratio
- sum
- number
- mean difference to higher values
- mean difference to lower values
- portion to higher value area
- portion to lower value area
- portion to higher values
- portion to lower values
- mean absolute difference to neighbors

Examples for relational features:

- Determining the position of an object with respect to the position of another object of a certain class. (x position of an object compared to the x position of surrounding objects of class a)
- Determining objects which have a minimum distance within a certain surrounding to a specified class, independent of their absolute distance (distance of an object from class a compared to the distance of surrounding objects to this class)
- Comparing the direction of an object to the direction of surrounding objects of a certain class.

### Comparison of arithmetic and relational features

Arithmetic and relational features have in common that they can be created from existing features and classes by the user. The basic differences between the two are listed below.

**Arithmetic features:**

- are calculations/combinations of existing features
- can combine different features
- apply to the single object

**Relational features:**

- represent a comparison of one object to its surrounding or sub-objects
- refer to only one feature
- refer to a group of objects

The handling of customized features becomes more flexible if you re-use already-created customized features. For example, if you want to calculate the difference of two channel ratios, you first create the ratios: Ratio 1/2 = <Mean CH1> / <Mean CH2> and Ratio 3/4 = <Mean CH3> / <Mean CH4>. The difference is then calculated by subtracting the customized features: <Ratio 1/2> - <Ratio 3/4>.

## Information on Classification




During the process of generating and refining a class hierarchy, it is essential to always know why and how an object is classified the way it is. Furthermore, it is also necessary to know how an object would be classified when using class-related features. All this information is provided by the image object information dialog. Another dialog which helps to get necessary information to improve a classification result is the 2D feature space plot. To display the objects' assignment to a certain class you can also use the „Feature View“ with the feature „Classification value of.“

### Image object information dialog


The main tool used to review information about the classification is the image object information dialog. This dialog is divided into three parts. “Features,” “Classification” and “Class Evaluation.” The first part, which is explained in the chapter “Information on Image Objects and Features,” is used both before and after the classification to receive information about feature values of an object. The classification part gives information about the current classification as well as the possible classification when using class-related features. The class evaluation part gives detailed information about the evaluation of one object for one class. Whenever an object is not classified the way you expect, the first thing to do is to select it and take a look at the image object information dialog.



**Note!** The values given in the image object information dialog are rounded either to the second or fourth decimal place, in case of smaller values. So when using this information it is necessary to keep in mind that a displayed value of 12.47 could as well be 12.468. Therefore, beware of too narrow definitions of membership functions.



The image object information dialog is opened by selecting “Toolbars & Dialogs > Image Object Information” or by selecting the respective button . Since it is often very useful to compare information of different parts of the image object information dialog, the dialog can be opened two times. There are two buttons   as well as the two menu items to open the dialogs. To get information about an object, simply click the object in the image view window. The information is then displayed in the “Image Object Information” dialog. If the dialog is “Feature” mode, you can right click in it and select the features to be displayed.

## Object table

Alternatively you can use the Object Table  to gain selective information on dedicated objects. When using the Object Table the first time, you are asked to configure it. To do so, you have to right-click in it and a configuration dialog comes up, called Statistics. Within this dialog you can determine what kind of objects together with what object features shall be displayed in the Object Table.

In difference to the Image Object Information dialogs, you can sort the objects by a certain feature. If you do not select any features to be displayed in the table, at least “Class” and “Membership” are displayed. If you know, that a certain class is defined by certain value ranges for one or more features, you can sort the objects according to these features. Then you can mark each object within this range and observe their location in the image view. **Please note: The Object Table is not available in the LDH version of eCognition.**

## Classification

The classification part of the image object information gives information about the current classification of an object as well as the possible classification when using class-related features. The membership values of the current classification can be seen under current classification. Note that only the three highest membership values are stored for this dialog or analysis. The results for other possible class assignments are given under „Alternative Assignment“. To switch between membership values with and without class-related features use the button  resp. .

Current classification



Under “Current Classification,” the classification and membership value for the best class as well as the second and third best classes resulting from the current classification are displayed. In contrast to the alternative assignment information, the current classification information is not calculated on demand when opening the dialog box. The values of the “Current classification” will be kept until the next classification is performed or one of the class descriptions is changed.

Alternative assignment

The window “Alternative Assignment” shows a list of the evaluations of all classes in the current class hierarchy concerning the selected image object. The displayed membership values are calculated on demand when you select an object.

Class evaluation

This interface gives detailed information about the classification of the selected image object. When you select an object, the values displayed are calculated on demand, using the current class descriptions of all applicable classes. This enables easy access to the evaluation of even complex class descriptions. At the same time, borders in feature space which distinguish this class from another can be quantitatively ascertained.

At the top of the „Class Evaluation“ window, the membership value of the selected class is displayed in bold letters ❶. Beneath that, the membership values for the expressions themselves as well as the membership values resulting from a combination of expressions by operators are given. Each expression of the class is listed beneath its operator. To the right of the expression, the objects feature value for this expression is given. In the right column, the resulting membership value for this expression is displayed for the respective classification mode ❸ that is selected by the button  or  (without or with class-related features). If the selected class has parent classes, the membership values for these classes are given as well ❷. No values in the column “Value” indicates that this feature is not applicable.

Feature	Value
<b>Evaluation of Class: Urban Impervious</b>	<b>❶ 0.539</b>
and (min)	1.000
Density : 1.76	1.000
or (max)	1.000
Rel. border to Urban Impervious neighbor-objects : 0.8788	❸ 1.000
Area : 144.00	0.000
<b>Parent Class: Impervious General</b>	<b>❷ 0.539</b>
and (min)	0.539
nearest neighbor : 0.62	0.539
Mean dessau_fir.tif: wd=0.77 d=10.93 w=14.21	
Mean dessau_term.tif: wd=0.76 d=6.37 w=8.40	
Ratio dessau_mir.tif: wd=0.64 d=0.02 w=0.03	
Stdev dessau_mir.tif: wd=0.09 d=0.33 w=3.78	
Mean dessau_mir.tif: wd=0.76 d=14.90 w=19.67	
Ratio dessau_nir.tif: wd=0.18 d=0.01 w=0.05	
Stdev dessau_nir.tif: wd=0.51 d=2.34 w=4.56	
Mean dessau_nir.tif: wd=0.21 d=5.06 w=24.47	

Use this tool to obtain information about objects which are classified incorrectly. Find out why the classification was incorrect and amend the respective class description. Another application is to verify the classification. In order to see how stable objects are in their classification, select the respective objects and compare the membership values of the first and second best classes. Low values or very similar values mean that the class descriptions are not very well suited to separate the classes.

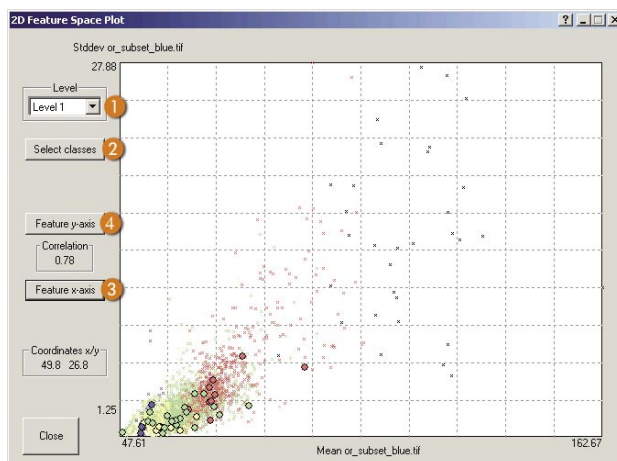
**Note!** The resulting membership values can differ from those given in the „Current Classification“ window, as the calculation is done on demand when you select the specific image object. Therefore, the „Alternative Assignment“ and also the „Class Evaluation“ windows show membership values, even if no classification has so far been performed.

## 2D feature space plot

The 2D feature space plot is used to visualize two image layers simultaneously. As with the „Feature View,“ not only spectral information can be displayed, but all features provided by eCognition.

This tool can be used to analyze the correlation of two features. If two features of a class description correlate highly, one of them can be deleted. The 2D feature space plot is also used for getting information about where an object or a group of objects is situated in the feature space.

To use the 2D feature space plot, select “Tools > 2D Feature Space Plot...” from the menu bar.



Unclassified image objects are displayed as small black crosses, classified image objects as small crosses colored according to the class color. Objects assigned as samples are displayed as circles colored in the class color. If samples are already present, you can navigate to their location in the image the same way as in the sample editor: switch on the sample navigation and click on the desired sample in the 2D feature space plot ⑤.

In the “Level” box, ① you can select the image object level used to create the feature space plot.

You can as well limit the display to a certain set of classes. To do this, open the selection menu by clicking on the “Select classes” button ②. In the selection menu you can choose which classes to display in the 2D feature space plot.

To select the features assigned to the x- and y-axis, click the “Feature x-axis” ③ or “Feature y-axis” ④ buttons and select a feature.

## Image Object Generation II: Classification-based Segmentation/Refinement

Typically, the initial image object levels in eCognition are created through multiresolution segmentation. As this method is entirely based on relatively general homogeneity criteria, the resulting image objects are better described as image object primitives. They become more meaningful image objects as soon as they are classified. In many cases, image objects of interest are not necessarily homogeneous. Classification-based segmentation is a method to extract or refine image objects following a more complex knowledge base. It is based on the classification of already existing image objects and the structure groups edited in the class hierarchy.

In addition, classification-based segmentation suits another purpose: Since different types of information are sometimes represented on different scales, different features can be classified best on different levels in the image object hierarchy. By using classification-based object generation, the classification results and especially the shapes of the image objects of interest on different image object levels can be rejoined into one image object level.

eCognition provides three different methods of classification-based segmentation: Classification-based fusion of image objects, border optimization and image object extraction. All three methods are performed in two steps:

1. Definition of structure groups
2. Object generation

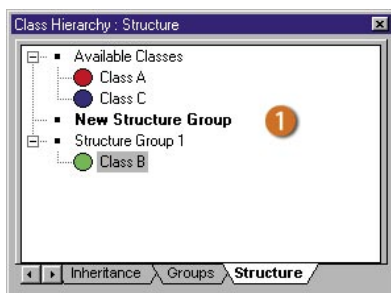
### Definition of structure groups

The definition of structure groups serves to create image objects based on the classification of existing image object primitives by means of a classification-based segmentation process. Structure groups are defined by the user. By editing structure groups you define how the classification-based segmentation will be performed. All classes within one structure group are treated as fitting to each other, even if they represent very different class descriptions. Classes in different structure groups are treated as not suited to each other. This plays a particular role for the refinement of image objects.


To edit structure groups, change to the “Structure” register tab in the “Class hierarchy” editor or select the menu item “Segmentation > Edit Structure Groups...”. In addition, the “Structure” register is opened automatically when opening the “Classification-based Segmentation” dialog.

You can edit the structure groups similar to the way you edit the inheritance and groups hierarchy:

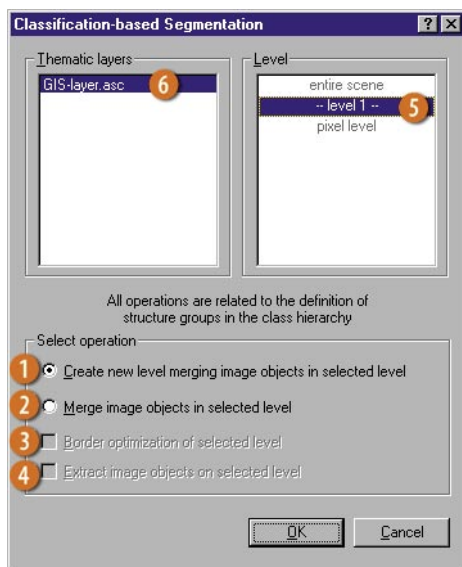
- To define new structure groups, left-click a class and, while depressing the mouse button, drag it over the term “New Structure Group” ① .
- To add a class to an existing structure group, do as described above, dragging the class over the desired group.
- To remove a class from a structure group, drag it back to the “Available Classes.”



## Object generation

To start classification-based segmentation, open the respective dialog by choosing the menu item “Segmentation > Classification-based Segmentation...” or click  in the tool bar. For classification-based segmentation, there are four different possible operations, which can be selected by checking the respective boxes ① to ④.

To define on which level to perform the operation, select and click the level to be used ⑤. To define whether or not a loaded thematic layer should be taken into account, select it in the relevant window ⑥. If a thematic layer is considered, the borders of the thematic layers limit the size of the objects resulting from classification-based segmentation. A thematic layer will be used for the process of classification-based segmentation if it is highlighted. To activate or deactivate a thematic layer, simply click it. If it is not possible to consider the thematic layer, its name will be displayed in brackets.



## Classification-based fusion of image objects

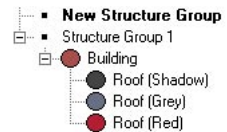
Features in an image are often represented by more than one image object of the same classification. By merging them you will get a new meaningful image object which truly represents the feature and not just a part of it. You can either create a new image object level in which the objects are merged **1** or merge them in the same level **2**. In the latter case former image objects will be lost. Which objects to merge is defined in the structure groups. In the fusion process all neighboring image objects that are assigned to classes organized in the same structure group will be merged.

Example:



Several image objects represent a house. All these objects are correctly assigned to the class *Roof (Red)* and belong to the semantic group *Building*. But you actually want the house to be represented by one meaningful image object.

To merge the image objects correctly, a new structure group has to be defined consisting of all the classes representing a house. After classification-based fusion, neighboring objects of these classes will be merged into individual objects, regardless of their original color.



Start the segmentation by clicking “OK.”



After classification-based fusion, houses are represented by one meaningful image object. For another example see [Guided Tours > Analysis of the degree of urban impervious surface](#).

## Classification-based refinement of image objects

Classification-based fusion of image objects only utilizes information from the image object level on which the objects are to be merged. Classification-based refinement techniques, on the other hand, use information from two image object levels, which must be situated directly above one another. This leads to the creation of new image objects on the more coarsely resolved (upper) image object level. The two different image object levels are brought into relationship with each other so that structures embodied at different resolutions are projected onto one single image object level. Due to this fact, the classification-based refinement techniques are a very interesting feature of eCognition.

### Border optimization of a selected level 3:

This method makes it possible to optimize the border of image objects on a selected level. Due to the fact that different degrees of resolution yield different information, there will always be image objects which do not correctly represent a feature in an image.

Example:



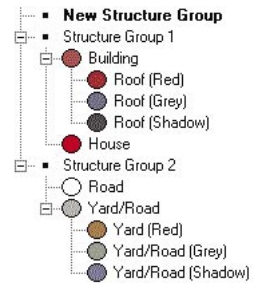
This image demonstrates two problems to which border optimization can be applied. The present segmentation represents houses quite well, but is not suitable for smaller features with less contrast. This is why you can find an image object assigned to the class *Trees* (rendered dark green) that covers part of the road and the roof. Border optimization can now be used to correct the borders of such image objects.



This is the same image segmented using a smaller scale parameter. Using this resolution, features like roads or single bushes are represented much better at this lower level. These features can be classified better here than on the coarser level. The part of the roof that was incorrectly assigned to *Trees* in the higher level is now classified as *House* (rendered red). The parts of the road in question are now assigned to the class *Road* (rendered white) instead of *Trees* as in the higher level.



Look at the definition of the structure groups shown here. Structure group 1 contains the classes representing buildings of image object levels one and two. The same applies to roads in structure group 2, with *Road* as a class of the lower level and the other classes being part of the higher level.



To start border optimization, enable it by checking the appropriate item **3**, choose the appropriate hierarchy level **5** and start the segmentation process by clicking “OK.”



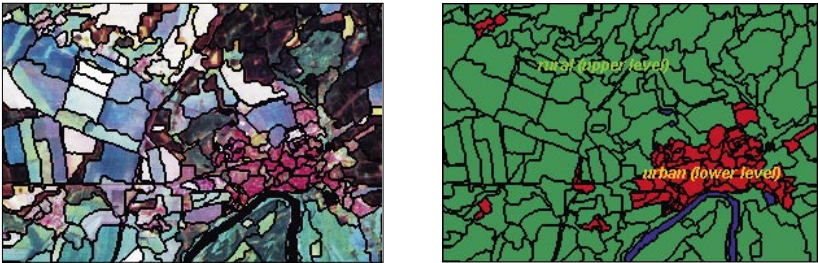
As a result, the borders of the image objects on the coarser resolution image object level have been changed using the classified level of sub-objects. During the process sub-objects were added to the neighboring super-objects of level two, if both belonged to classes within the same structure group.

On the one hand, the result is an image object level with more meaningful image objects and, on the other, information which is best provided by different resolutions has been transferred to one image object level. For further details see Concepts and Methods > Classification-based shape correction of image objects.

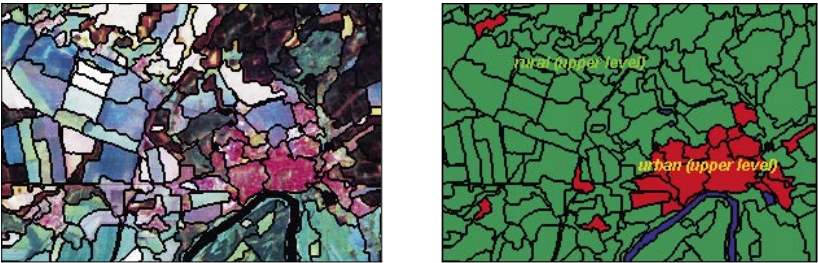
#### Extraction of image objects on a selected level **4**:

While classification-based border optimization is best used for the shape correction of image objects, the “Extract image objects on selected level” method serves two purposes: on the one hand it works similarly to border optimization, but includes objects in the middle of super-objects as well as on the border. On the other hand it makes it possible to combine the structures of two different resolutions in one single image object level.

Example:

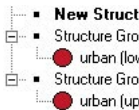


The two images above show a Landsat TM subset. They belong to the same image object level. The main structures visible are rural and urban areas. Note that the rural areas are very well represented by the image objects. This means that the resolution of this image object level represents rural structures to a satisfying extent. The urban structures, however, would be represented in more detail if a finer object resolution were applied.



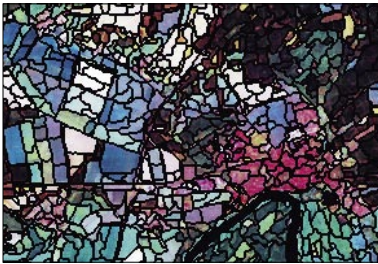
The resolution displayed above represents urban structures that will yield a more detailed classification. The displayed class, *urban (lower level)*, can now be easily differentiated for different types of urban features. However, the resolution is much too fine for the representation of rural structures. It would be ideal to have both resolutions projected onto one image object level. This can be achieved using eCognition's image object extraction technique.

Again, the first thing to do is to define structure groups.



Since the new image object level is supposed to contain rural structures of the higher level and urban structures of the lower level, you have to apply the extraction of image objects of the lower level to the higher level. The rural classes are not included in the design of the structure groups; they remain the way they are in the higher level. The urban classes are separated into two different structure groups. Since the objects of *urban (lower level)* are not of the same structure group as the ones of *urban (upper level)*, they will all be extracted from their super-objects and their borders will appear in the newly generated level.

After the structure groups have been defined, enable the image object extraction by checking the appropriate button **4**, select the higher level in the “Level” field **5** and click “OK” to start the process of image object extraction.



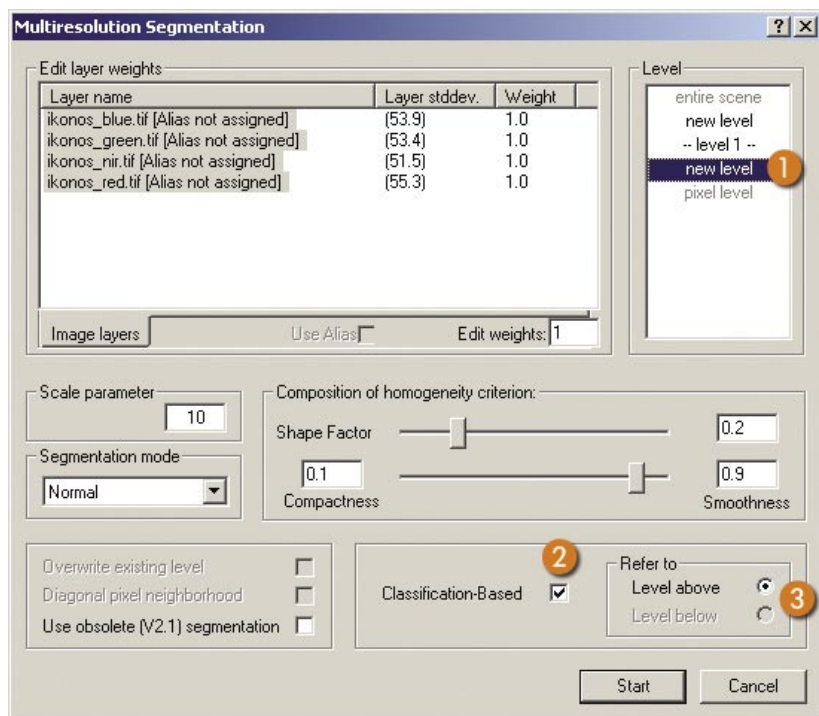
After image object extraction has finished, both coarser rural and finer urban structures are represented in the same image object level. More detailed urban classes can now be applied. For further details, see [Concepts & Methods > Extract image objects on selected level](#).

### Classification-based multi-resolution segmentation

Applying a classification based multi-resolution segmentation can be seen as a selective or local segmentation. When segmenting in this mode, you perform the common multiresolution segmentation only on objects belonging to the structure groups you have created; all other objects remain untouched. This leads to a new segmentation level, wherein objects of different segmentation parameters coexist. You can either perform a classification-based sub- or super-segmentation - both are analogous to normal multi-resolution segmentation except that only dedicated objects are affected.

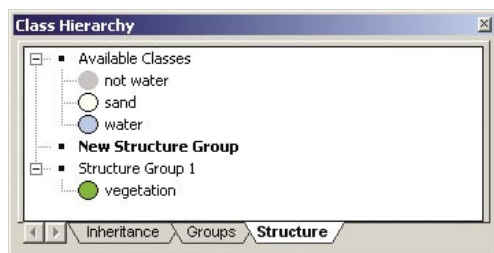
As with other classification-based operations, you first have to define structure groups (see above). Then you choose the multiresolution segmentation dialog with the segmentation parameters of your choice and select a new level **1**. After this you switch on

the “Classification-Based” check box **2**. Depending on already-existing segmentation levels you can either refer to the classification of the level above or below **3**.



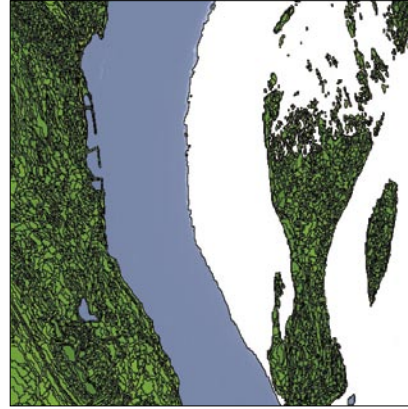
**Note!** if the level you refer to is not classified, it will only be copied into the new level.

Multi-resolution segmentation will only be applied to the determined structure groups:





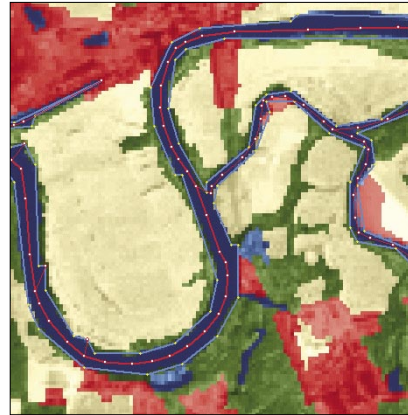
*before classification-based segmentation*






*after classification-based segmentation*

### Classification-based object cut

With the introduction of skeletons it is possible to refine the objects' shapes according to their inner geometrical structure. In more detail: it is possible to cut-off outer parts of image objects, which consequently leads to a less complex geometry of the objects. Especially for man-made objects theoretically a less complex geometry is typical. But in several cases the shape of the generated image objects is inconvenient due to the spectral properties of the image or the objects. In the example below, parts of a river network are shown. Because of their semantic similarity of being waterbodies, all neighboring river objects were merged. Regarding the skeleton of this object, the mainline would represent the main stream of the river network, while the longer branches represent the inflows.



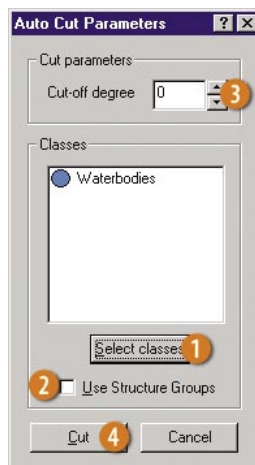
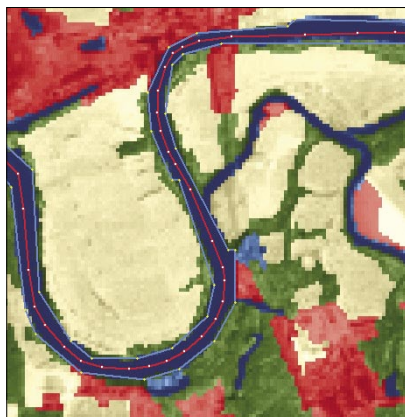
Once classified as *waterbodies*, these objects can be enhanced by cutting off all branches of a certain degree. Choose the appropriate dialog from the menu "Segmentation > Classification-based object cut ..." or click the  button.

Press the „Select classes“ button  1 to select the class(es) to be affected by the cut. You can also use all defined structure groups by checking the appropriate check box  2.

By entering the “Cut-off degree” ③ you determine the branch order(s) to be cut off. The “Cut” button ④ executes the cut and creates several smaller and one remaining river object.

Switching the “Cut-off degree” to 0 will cut off all branches; only the main lines remain:

If the object is very narrow, it is cut also along the mainline in a way that only objects with a branch order of 0 (mainline) will be created.



**Note!** Depending on the number of objects and their complexity an automated object cut can be extremely time consuming. For these cases we recommend reducing the objects' complexity by cutting them by hand into several smaller ones at convenient locations before performing an automated cut. Another way to avoid too complex objects is to increase the polygons' thresholds, when creating them. Canceling the automated cutting process leads to an incomplete result, which is irreversible.



## Accuracy Assessment and Statistics

eCognition provides accuracy assessment methods serving different purposes. These tools produce statistical and graphical outputs which can be used to check the quality of the classification results. The tables from the statistical results can be saved in comma-separated ASCII \*.txt files. The graphical results can be exported in their current view settings as several raster files.

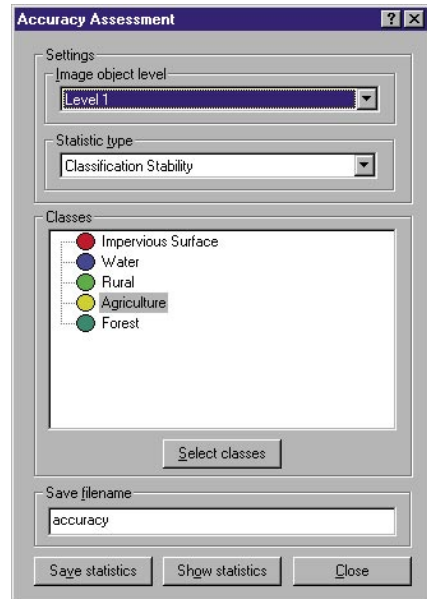
In addition, eCognition provides a powerful statistics tool with basic GIS functionality. These tables can also be saved as ASCII \*.txt file with comma-separated columns.

### The Accuracy Assessment dialog box

Open the “Accuracy Assessment” dialog by selecting the menu item “Tools > Accuracy Assessment...” from the main application menu bar.

A project can contain different classifications on different image object levels. Specify the image object level of interest by using the “Image Object Level” drop-down menu.

In the “Classes” window all classes and their inheritance structure are displayed. To select the classes you want to assess, click the button “Select classes” and make a new selection in the following dialog box. By default all available classes are selected. You can deselect classes by a double-click in the right frame.



In the “Statistic Type” drop-down menu, select a method of accuracy assessment. eCognition offers the following four methods, which are discussed below.

- Classification Stability
- Best Classification Result
- Error Matrix based on TTA Mask
- Error Matrix based on Samples

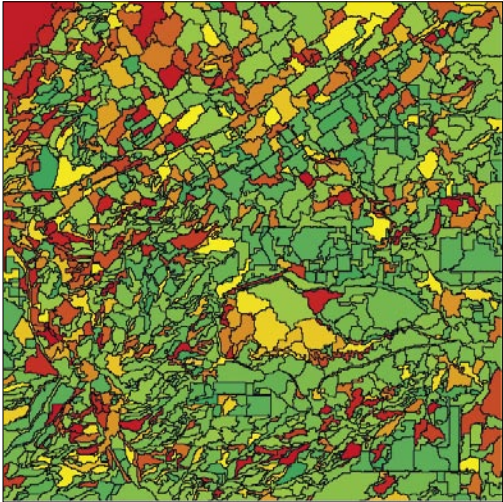
To view the accuracy assessment results, click “Show Statistics.”

To export the statistical output to a text file, click “Save Statistics” in the respective dialog. You can enter a file name of your choice, the extension \*.txt is attached automatically.

### Classification stability

Select “Classification Stability” from the “Statistic Type” box and click “Show Statistics.”

Due to eCognition’s fuzzy classification concept, an image object has a membership degree to more than one class (see also “Concepts & Methods” for further reading). With this tool you can explore the differences in degrees of membership between the best and the second best class assignments of each object, which can give evidence about the ambiguity of an object’s classification. The graphical output can be displayed by selecting “Classification Stability” in the “View Settings” dialog. The value is displayed for each image object in a range from dark green (1.0, nonambiguous) to red (0.0, absolutely ambiguous). Move the mouse over an image object to get the exact difference between the best and second-best class assignment via tool tip.



The statistical output displays, in terms of class, the basic statistical operations performed on the differences between the best and second best degrees of membership.

For an example see the guided tour “Landsat TM subset of Orange County”: Checking classification stability.

Classification Stability					
Class	Objects	Mean	StdDev	Minimum	Maximum
Impervious Surface	86	0.51	0.29	0.00322	1
Water	14	0.449	0.199	0.00559	0.652
Rural	430	0.47	0.266	0.00211	1
Agriculture	392	0.347	0.239	0.00163	1
Forest	336	0.309	0.197	0.00345	0.839

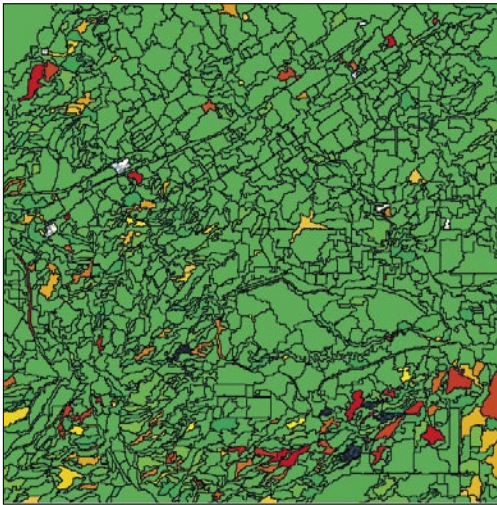
☐ reduce
 ☒ expand
 Close



## Best classification result

Select “Best Classification Result” from the “Statistic Type” box and click “Show Statistics.”

This tool generates a graphical and statistical output of the best classification results for the image objects of a selected level. Due to eCognition’s fuzzy classification concept, an image object has memberships in more than one class (see also “Concepts & Methods” for further reading). The classification with the highest assignment value is taken as the best classification result.



To display the object’s best classification result graphically, select “Best Classification Result” in the “View Settings” dialog. The highest membership value is displayed for each image object in a color range from dark green (1.0 maximum degree of membership) to red (0.0 no membership). The precise value of degree of membership and the assigned class are shown via tool tip, if you move the mouse above the image object of interest in the main view window.

The “reduce” and “expand” commands refer to the abbreviated or written out subjects of the columns in the “Best Classification Result” dialog.

In the statistical output the best classification result is evaluated per class. Basic statistical operations are performed on the best degrees of membership of the image objects for each class (number of image objects, mean, standard deviation, minimum value and maximum value). Thus it is possible to evaluate how the objects of a class fulfill the class description.

Class	Objects	Mean	StdDev	Minimum	Maximum
Impervious Surface	86	0.604	0.275	0.104	1
Water	14	0.855	0.231	0.169	1
Rural	430	0.57	0.245	0.101	1
Agriculture	392	0.643	0.213	0.105	1
Forest	336	0.716	0.206	0.189	1

☐ reduce
 ☒ expand
 Close

See the guided tour [“Landsat TM subset of Orange County”: Check the best classification result](#) for an example.

### Error matrix based on TTA mask

This method uses test areas as a reference for classification quality. Test areas can be generated outside eCognition and imported into an eCognition project by means of a TTA mask to compare the classification with ground truth based on pixels.

The TTA mask has to be created or imported first. To create a TTA mask from sample objects, select “Samples > Create TTA Mask from Samples.” In the dialog choose the appropriate level from which you want to create the TTA mask – usually the same on which you want to perform the accuracy assessment. If you want to perform the accuracy assessment based only on your sample objects, you do not need to load a TTA mask.

If you want to compare your classification result with an externally created TTA mask you must load it first. The respective dialog is opened via the menu item “Samples > Load TTA Mask....” If necessary assign the classes of the TTA mask to appropriate classes in your project: select “Samples > Edit Conversion Table.” In the left column of the dialog the classes of the TTA mask are displayed. By right-clicking on one of them you can choose to which class of your project you want to assign the selected class.

After the TTA mask has been loaded, it will be displayed in the main view window.

After the TTA mask has been imported into the project and its classes have been linked to the respective classes in your class hierarchy, select “Error Matrix based on TTA Mask” from the “Statistic Type” box and click “Show Statistics.”

The statistical output is presented as a confusion matrix with several accuracy measures. The first column of the confusion matrix shows the classes you want to assess (user’s classes). In the following columns, the numbers of pixels covered by the TTA mask (reference classification) for each class are displayed. The sum for each class in the TTA mask (reference classification) is shown in the last row. Accordingly, the last column shows the sum of all pixels classified by the classification you want to assess. As one can see, there are differences between the appropriate sums. They can be explained by looking at the matrix, which gives evidence of the quality of the classification compared to the TTA mask: Regarding the class Rural there are 6636 pixels covered by the rural areas of the TTA mask. 6414 of those were classified as Rural, 222 have been classified as Impervious Surface. This results in a producer’s accuracy of 0.967. On the other side 6846 pixels are classified as Rural. Thereof 6414 pixels are also assigned Rural in the TTA mask, but 432 pixels are classified as Agriculture. Therefore, the user’s accuracy

is 0.937. As a rule of thumb, no zeros should occur in the top-left-bottom-right diagonal. In the example, however, this is the case for the class Agriculture. Obviously this class is hardly separable from Rural and Forest. For further reading about the interpretation of confusion matrices see “Concepts & Methods” or consult standard literature such as Lillesand & Kiefer.

User \ Reference Class	Impervious Surface	Water	Rural	Agriculture	Forest	Sum
<b>Confusion Matrix</b>						
Impervious Surface	1750	0	222	0	0	1972
Water	0	4280	0	0	0	4280
Rural	0	0	6414	432	0	6846
Agriculture	0	0	0	0	1321	1321
Forest	0	0	0	2164	4996	7160
Unclassified	0	0	0	0	0	0
Sum	1750	4280	6636	2596	6317	
<b>Accuracy</b>						
Producer	1	1	0.967	0	0.791	
User	0.887	1	0.937	0	0.698	
Hellden	0.94	1	0.951	0	0.741	
Short	0.887	1	0.907	0	0.589	
KIA Per Class	1	1	0.951	-0.0652	0.687	
<b>Totals</b>						
Overall Accuracy	0.808					
KIA	0.745					

Based upon the confusion matrix the dialog shows several accuracy measures for each class and two overall measures. They are explained in detail in “Concepts & Methods.” All of them have a range of value between 0 and 1 except the KIA (Kappa Index of Agreement). This measure can also have negative values. As a rule of thumb, you can say the closer these values are to 1, the better the classification (i.e., the higher the accuracy) of this class.

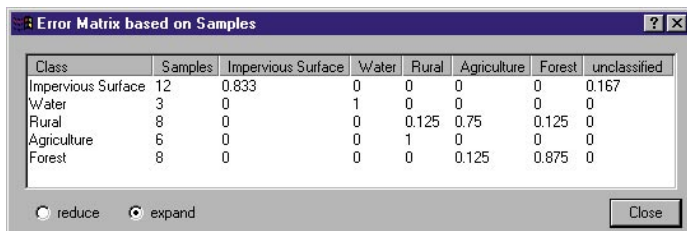
See also the guided tour “Landsat TM subset of Orange County”: Compare the test areas with classification for an example.

### Error matrix based on samples

Select “Error Matrix based on Samples” from the “Statistic Type” box and click “Show Statistics.”

This tool is similar to the “Error Matrix based on TTA Mask,” but considers samples (not pixels) derived from manual sample inputs. Note that it clearly does not make sense to use the same sample objects for accuracy assessment as you used for the nearest neighbor classification, since they have been assigned to the correct classes anyway. Delete these samples and declare new sample objects for the calculation of the error matrix.

The resulting error matrix is interpreted in the same way as the confusion matrix based on a TTA mask, but the match between the sample objects and the classification is expressed in parts of class samples. Have a look at the error matrix above: The first



Class	Samples	Impervious Surface	Water	Rural	Agriculture	Forest	unclassified
Impervious Surface	12	0.833	0	0	0	0	0.167
Water	3	0	1	0	0	0	0
Rural	8	0	0	0.125	0.75	0.125	0
Agriculture	6	0	0	1	0	0	0
Forest	8	0	0	0	0.125	0.875	0

☐ reduce
 ☒ expand
 Close

column shows the class names of the sample objects. The second column shows the number of samples taken for the respective class. Regarding the class *Rural*, there are eight sample object assigned to *Rural*.  $\frac{1}{4}8$  (0.125) of them (one sample object) is assigned to *Rural* by the classification, while  $\frac{6}{4}8$  (0.75) – which corresponds to six sample objects – have been classified as *Agriculture*.  $\frac{1}{4}8$  (0.125), which corresponds to one sample object, has been classified as *Forest*.

A value of 1 means that all sample objects of a class are assigned to this class; a value of 0 means none of the class's sample objects have been assigned to it. In the example above, all sample objects of *Water* and *Impervious Surface* are also classified accordingly.

For an example of how to create a Training and Test Areas Mask, see the guided tour “Landsat TM subset of Orange County”: Declaring sample objects.

## Statistics

To open eCognition's statistics tool, select the menu item „Tools > Statistics...“ This powerful tool enables you to perform a variety of basic statistical operations class by class at any image object level and with any feature provided, including your own customized features. It also enables you to evaluate the mean degrees of membership of the classes' objects, which can give evidence of the overall ambiguity of a class. This is useful for evaluating a fuzzy accuracy assessment.

The image object level on which to perform the statistics can be selected in the drop-down menu „Image Object Level“ ①.

Select the statistics type in the “Statistic type” menu ②. It is possible to produce statistics based on all the classes that are selected (“By Classes”) or on each single object of the selected classes (“All Objects”).

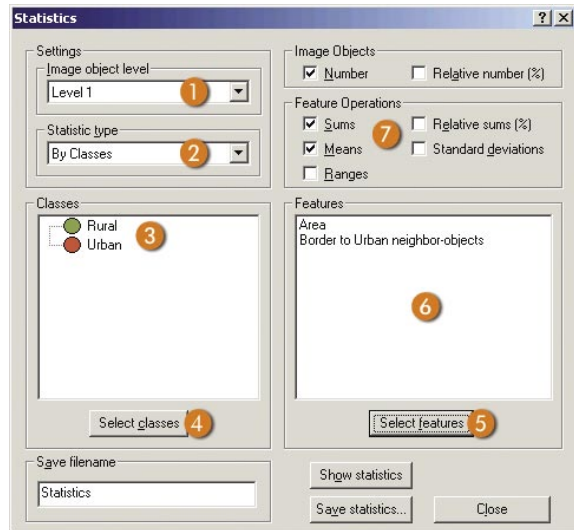
The “Classes” ③ field displays those classes for which statistics are to be calculated. To edit the selection, click the button “Select Classes” ④.

To make a feature selection, click the button “Select Features” ⑤. This will open a dialog in which you choose the features on which the statistics are to be performed.

The current feature selection is displayed in the field, “Features”  
6.

Select the statistical operations to be performed on the features by checking the appropriate boxes. The following operations are available 7:

Select the statistical operations to be performed on the features by checking the appropriate boxes. The following operations are available:



#### Sums

For each class the sum of the specified feature values of all image objects assigned to this class is calculated.

#### Rel. Sums

For each class the proportion of the **sum** of the specified feature values compared to the sum of this feature values for all selected classes is calculated.

#### Means

For each class the mean value of the specified feature values of all image objects assigned to this class is calculated.

#### Standard deviation

For each class the standard deviation of the specified feature values of all image objects assigned to this class is calculated.

#### Range

For each class the range (max-min) of the specified feature values of all image objects assigned to this class is calculated.

#### Relative Number %

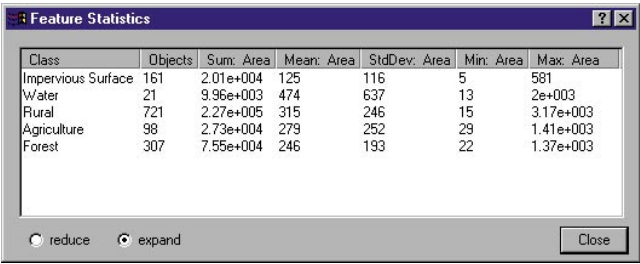
The relative number is the number of objects of the specific class, divided by the total number of objects.

To view the statistics, click the button „Show Statistics.“ The statistical output is presented in a table.

If for a feature no value is available, a line (--) is shown. Values exceeding three digits are shown in exponential format.

**Note!** Select at least one operation, otherwise the selected features will not be added to the table.

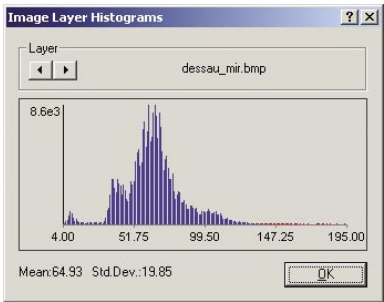
To save the statistics to a file, click the button “Save Statistics.” The matrix is exported as a text file; commas separate the columns.



See the guided tour “[Analysis of the degree of urban impervious surface](#)”: [Analyzing the classification result](#)“ as an example.

### Image layer histograms

To view pixel value histograms of the single image channels, choose „Tools > Layer Histograms ...“ and navigate through all available image layers. This tool additionally provides information on the mean value and standard deviation of the image layer. The blue slots indicate areas of significant values, while the red slots mark areas of lower occurrence.



## Automation of Operations

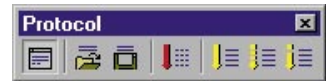
To automate part of your workflow, eCognition allows you to record, execute and edit protocols. These protocols can be used to automatically run a user defined analysis on more than one project. Once you have created a stable sequence for handling one image, this can be transferred to other images.

A protocol consists of one or more operations which can be executed all at once or step by step.


### The protocol tool bar

In the “Toolbars & Dialogs” menu you can select to display the protocol tool bar. This tool bar allows you to start the basic protocol functions.

Furthermore, all operations can be started from the menu bar or the context menu, which is opened with a right-click at the protocol editor.

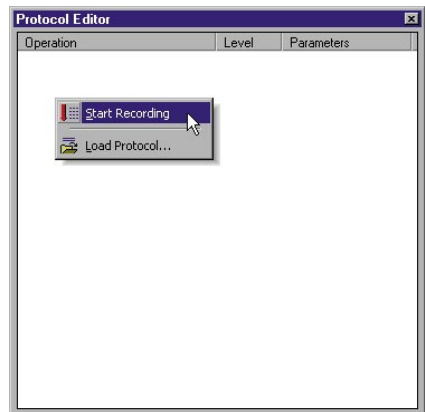


### Recording protocols

Select “Protocol > Open Protocol Editor...” from the menu bar or click the icon  to open the “Protocol Editor” dialog. To record a new protocol, select “Protocol > Start Recording” from the “Protocol Editor” dialog box or right-click and choose “Record New” from the context menu.

Major operational steps are now automatically recorded.

The recorded operations, the level at which they are recorded, and the parameters are displayed in the “Protocol Editor” window. To stop recording, select “Protocol > Stop Recording” or right-click and choose “Stop Recording.”



To continue recording, select “Protocol > Start Recording” or right-click and choose “Start Recording” from the context menu. To save a protocol choose “Protocol > Save Protocol” or right-click and choose “Save Protocol.”

**Note!** The following operations are recorded:

- “Segmentation > Multiresolution Segmentation...”
- “Segmentation > Classification-based Segmentation...”
- “Segmentation > Delete Level...”
- “Analysis > Load Class Hierarchy...”
- “Analysis > Classify...”
- “Polygons > Create Polygons”
- “Export > Export Image Objects...”
- “Export > Export Classification...”
- “Export > Export Current View...”

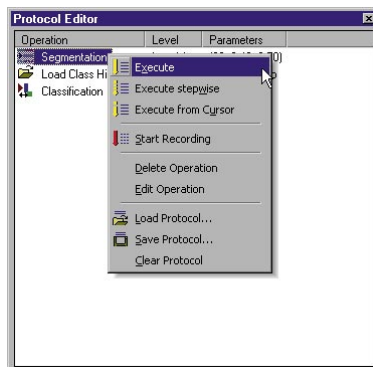
## Executing protocols

To load a protocol, select “Protocol > Load Protocol” from either the “Protocol Editor” dialog box or the context menu.


**Note!** Protocols are not saved as part of the project. They are only temporarily available after recording or after loading. Thus, parametrized features such as „Distance to line“ have to be updated in the project before executing the protocol, because they still contain the values relevant for the original project for which the protocol was created.

To execute a protocol, choose “Protocol > Execute Protocol” from the protocol editor dialog box or right-click and choose “Execute” from the context menu. A protocol cannot be executed while in the recording phase; the recording has to be stopped prior to executing the protocol.

To execute a protocol stepwise, you first have to select an operation from which to start the stepwise execution by highlighting





it. Then choose “Protocol > Execute stepwise” or click the icon  or right-click and choose “Execute stepwise.”

The function “Execute from Cursor” allows you to execute a protocol from the highlighted operation. To execute a protocol from the cursor position, highlight the operation from which you wish to start and select “Execute from Cursor” or right-click and select “Execute from Cursor.”

To execute a protocol for a new project, the following prerequisites have to be fulfilled:

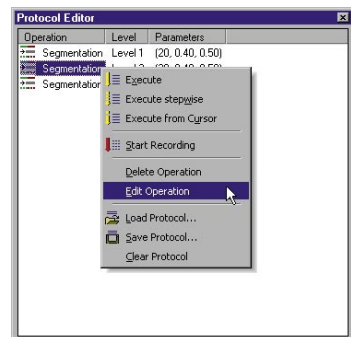
- the number of channels used for multiresolution segmentation must be identical to that in the project used for recording the protocol.
- the number of image object levels has to correspond to those of the project used for recording the protocol.

**Note!** When using aliases for multiresolution segmentation, the alias names of the recorded and executing project must be identical. Choose „Project > Assign layer alias“ from the menu bar to assign appropriate aliases to your layers

## Editing protocols

After you have recorded a protocol, you can also edit or delete single steps of the protocol. To modify the parameters of any entry, double-click the item or right-click it and select “Edit Operation.” You can then insert new parameters for this operation.

The dialog opens with the settings as defined by the protocol. They can be changed within the dialog. To accept the new settings, execute the operation (for example click “Start” after changing the segmentation settings). The operation will not be executed; the protocol is merely modified. The modified protocol can now be executed; furthermore you can save it under a new name and apply it to different images.







## Manual Editing of Image Objects

eCognition provides two modules for manually editing image objects: The manual fusion tool is used to manually merge selected, neighboring objects. The manual classification tool enables easy class assignment of selected objects. Both tools can be used to correct the result of an eCognition analysis and to bring it into a final form.

**Note!** Both manual object fusion and manual classification operate on the current image object level only.

### Selecting objects to fuse or classify

To select the objects to fuse or classify by hand you have three possibilities: you can select them individually by switching on the “Default Selection Mode” . Besides, you can select several objects at once along a line by switching to the “Line Selection”  mode. In this mode all object which are hit by the line you draw, will be selected. To select objects within an area, you can either switch to the “Polygon Selection”  mode or to the “Rectangular Selection”  mode. If you are in line or polygon/rectangular mode, you have to define the line or area in an image view window to select the respective objects.


### Manual object fusion

This tool allows you to manually modify the object hierarchy. If you want to merge neighboring objects into a new single object, you first have to choose “Input Mode > Manual Object Fusion” from the menu bar or choose “Manual Object Fusion” from the “Input Mode” combo box to activate the “Manual Object Fusion” input mode.

Select all objects you want to combine with a single mouse-click. A second mouse-click will deselect an object. Selected objects are displayed in the manual object fusion selection color. The default setting is yellow. Objects which cannot be merged with the already selected ones are displayed in the overall selection color, which is red by default. Both colors can be changed under “View > Edit Highlight Colors.”

**Note!** Due to the hierarchical organization of the image objects, an object cannot have two super-objects. This also limits the possibilities for manual object fusion, since two neighboring objects cannot be merged, if they belong to two different super-objects.

To clear the selection result, click the “Clear Selection for Manual Object Fusion” button or deselect single objects with a single mouse-click.

To merge selected objects, click the “Merge selected Objects” button  from the “Manual Object Fusion” tool bar.

To deactivate the “Manual Object Fusion” input mode, choose “Input Mode > None” from the menu bar or choose “Input off” from the “Input Mode” combo box.

## Manual classification

Manual classification can be used for the following purposes:

- manual correction of previous classification results (incl. classification of previously unclassified objects)
- classification without knowledge base (in case the creation of an appropriate knowledge base is more time- and labor-consuming), using the initial segmentation run for automated digitizing

**Note!** By means of nearest neighbor classification eCognition provides a very easy method for quick interactive classification. Unlike knowledge-based classification, manual classification does not build up decision rules for additional class assignment. When using both manual and knowledge-based assignments for one class, manual classification has the highest priority and determines the assignment result.

To perform a manual classification, choose “Input Mode > Manual Classification” from the menu bar or choose “Input Manual Classification” from the “Input Mode” combo box to activate the manual classification input mode.

In the “Class Hierarchy” window select the class you want to manually assign objects to.

Objects can now be manually classified by a single mouse-click. As soon as the object is classified, it appears in the previously defined pertinent class color.

If no class is selected, a mouse-click will delete the previous class assignment.

**Note!** An important precondition for manual classification of objects is the existence of a class hierarchy within the currently opened project.

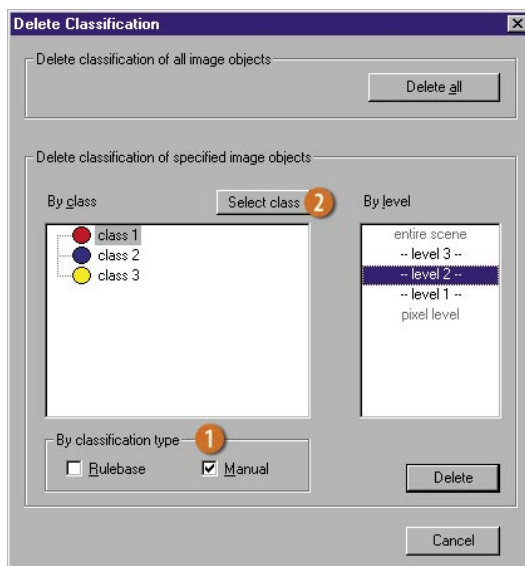
To undo a manual classification, simply click the object a second time.

To delete the manual classification results, choose “Classification > Delete Classification.”

In the resulting dialog, select the classification type “Manual” to delete only the manual class assignments **1**.

**Note!** Since manual classification overwrites the previous class assignment, the deletion of manual classification results in unclassified objects.

To delete the manual classification for selected classes, click the button “Select class” **2** and select one or more classes. Your selection will be displayed in the left window. In the right window you can select the level on which you want to remove the manual classification of objects.



Press “Delete” to delete the classification according to the previously selected setting.

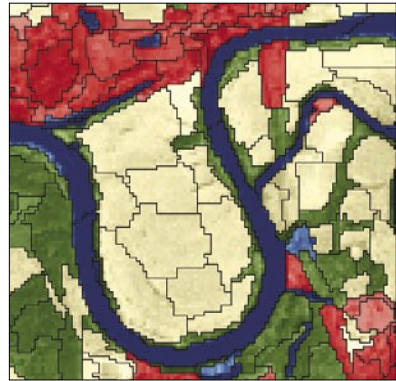
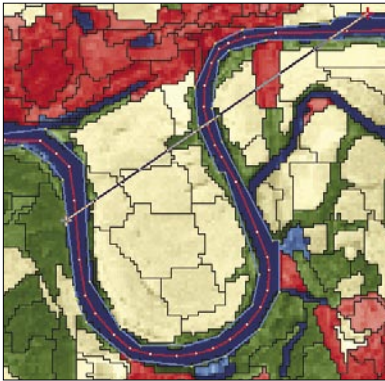
Press “Cancel” to quit the “Delete Classification” dialog.

To deactivate the manual classification input mode, choose “Input Mode > None” from the menu bar or choose “Input off” from the “Input Mode” combo box.

## Manual object cut

Objects can be cut manually cut by choosing the appropriate input mode.

Objects can only be cut along a straight line defined by two points on the object's outer border. Depending on the object's shape, the cut line can cross the object's outline several times. Or two or more new objects will be created. To cut an object select the appropriate input mode and click on the object of concern. Then you can draw the cut line. Note that the end point of the cut line will always remain on the outer line of the object.





# 6 USER INTERFACE

## Overview

In this section of the user guide you will find detailed descriptions of the user interface.

eCognition tool bar .....	309
---------------------------	-----



eCognition menu .....	312
-----------------------	-----



### List of eCognition dialogs:

Menu .....	312
Dialogs .....	319
2D Feature Space Plot .....	319
Accuracy Assessment .....	320
Apply Standard Nearest Neighbor to Classes .....	322
Apply TTA Mask to Level .....	322
Auto Cut Parameters .....	323
Class Description .....	324
Class Hierarchy .....	325
Classification-based Segmentation .....	327
Classification Settings .....	328
Colors .....	329
Conversion Table .....	329
Create Customized Feature .....	330
Create Polygons .....	332
Create Project .....	332
Create TTA Mask from Level .....	334
Customize .....	334
Define Brightness .....	336
Delete Classification .....	337
Delete Level .....	338
Delete Samples of Selected Classes .....	338
eCognition Hardlock Utility .....	339

Edit Standard Nearest Neighbor Feature Space .....	340
Edit Highlight Colors .....	341
Edit Layer Mixing .....	341
Edit Minimum Membership Value .....	343
Export Classification .....	343
Edit Parametrized Feature .....	344
Export Image Objects .....	345
Export Image Object Shapes .....	346
Export Scale Parameter Analysis .....	347
Feature Space Optimization .....	347
Feature Statistics .....	350
Feature View .....	351
Help Keyboard .....	352
Image Layer Histograms .....	352
Image Object Information .....	353
Input Mode .....	354
Insert Expression .....	355
Layer Properties .....	355
Membership Function .....	356
Membership Function Parameters .....	358
Message Console .....	358
Multiresolution Segmentation .....	359
Object Table (not available in the LDH version) .....	360
Options .....	361
Pan Window .....	362
Project Info .....	362
Protocol Editor .....	362
Sample Editor .....	363
Sample Navigation .....	365
Sample Selection Information .....	365
Scale Parameter Analysis .....	367
Select Layer .....	369
Select Level .....	369
Select Displayed Features .....	369
Select Features for Statistic .....	370
Select ID Column .....	370
Select Operator for Expression .....	371
Select Single Feature .....	372
Set Nearest Neighbor Function Slope .....	372
Statistics .....	373
Subset Selection .....	374
System Info .....	375
User Information .....	375
View Settings .....	375



## Tool Bar



### Create new Project

Start a new project and load image layers and thematic layers.



### Open Project

Open an existing project.



### Save Project

Save project to disk.



### User Information

Open the user information dialog.



### Message Console

Open the “Message Console Dialog”.



### Multiresolution Segmentation

Open the “Multiresolution Segmentation” dialog and invoke segmentation.



### Classification-based Segmentation

Open the “Classification-based Segmentation” dialog and invoke segmentation.



### Delete Level

Delete a level of the image object hierarchy.



### Create/Modify Polygons

Create polygon outlines for image objects.



### Delete Polygons

Delete polygon outlines of image objects.



### Edit Class Hierarchy

Open the “Class Hierarchy” dialog.



### Classification-Based Automatic cut

Open the “Auto Cut Parameters” dialog



### Sample Editor

Open the “Sample Editor” dialog.



### Image Object Information 1

Open the “Image Object Information 1” dialog.



### Image Object Information 2

Open the “Image Object Information 2” dialog.



### Sample Selection Information

Open the “Sample Selection Information” dialog.



### Scale Parameter Analysis

Open the “Scale Parameter Analysis” dialog.



### Feature View

Open the “Feature View” dialog.



### Customize Features

Open the “Customized Features” dialog.



### Feature Space Optimization

Open the “Feature Space Optimization” dialog



### Input Mode

Change the input mode.



### Without or with class-related Features

Switch classification mode.



### Number Classification Cycles

Define the number of classification cycles for class-related features.



### Advanced: Simulated Annealing

Open the “Advanced Classification Settings” dialog.

**Classify**

Invoke image object classification.

**View Settings**

Opens the “View Settings” dialog.

**View Layer**

Render current level using layer mean values of pixel.

**View Classification**

Render classification of current level.

**View Samples**

Render samples in current level.

**Feature View**

Switch back to feature view display.

**View Pixel / Object Mean**

Switch between pixel and object mean display.

**Show/Hide Polygons**

Display of computed polygon.

**Show/Hide Outlines**

Display of computes pooutlines.

**Show/Hide Skeleton**

Display of skeletons.

**Image Layer Mixing**

Open the “Edit Layer Mixing” dialog.

**Edit Highlight Colors**

Open the “Edit Highlight Colors” dialog.

**Mix Single Layer Grayscale**

Display image data in single layer grayscale format.

**Mix Three Layers RGB**

Display image data in three layer RGB format.

**Show Previous Image Layer**

Display the previous image layer.

**Show Next Image Layer**

Display the next image layer.

**Select Level in Object Hierarchy**

Select level in object hierarchy for processing.

**Next Level down in Object Hierarchy**

Go to next lower level in the hierarchy of image objects.

**Next Level up in Object Hierarchy**

Go to next higher level in the hierarchy of image objects.

**Next Level down in Groups Hierarchy**

Displays classes at next lower level in the groups hierarchy.

**Next Level up in Groups Hierarchy**

Displays classes at next higher level in the groups hierarchy.

**Activate/deactivate sample navigation**

Enables the navigation to samples selected in the “Sample Editor” or the “2D Feature Space Plot”.

**Normal Cursor**

Switch from zoom or pan mode to normal cursor.

**Area 100%**

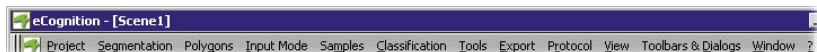
Reset zoom factor to 100 %.

**Zoom In**

Zoom nearer.

- |   |   |   |  |
|---|---|---|--|
|    | <b>Zoom Out</b><br>Zoom further out.  |  | <b>Execute Protocol from Cursor</b><br>Execute protocol from selected action.                          |
|    | <b>Area Zoom</b><br>Zoom into area.   |  | <b>Polygon Selection</b><br>Select a polygon.  |
|    | <b>Zoom Out</b><br>Zoom in to center.   |  | <b>Line Selection</b><br>Select a line.  |
|    | <b>Zoom In</b><br>Zoom out from center.                                       |  | <b>Rectangle Selection</b><br>Select a rectangle.  |
|    | <b>Pan</b><br>Pan current view.   |  | <b>Image Object Cutting</b><br>Cut image objects.  |
|    | <b>Pan Window</b><br>Open Pan window.   |  | <b>Manual Image Object Classification</b><br>Classify an image object manually.                        |
|    | <b>Zoom Scene to Window</b><br>Zoom the scene to fit into the current window. |  | <b>Image Object Fusion</b><br>Fuse selected objects  |
|    | <b>Open Protocol Editor</b><br>Open the "Protocol Editor" dialog.             |  | <b>Merge selected Objects</b><br>Merge selected objects to one object.                                 |
|    | <b>Load Protocol</b><br>Load protocol from disk.                              |  | <b>Clear Selection for Manual Object Fusion</b><br>Clear all selected object and do not merge objects. |
|    | <b>Save Protocol</b><br>Save protocol to disk.                                |   |  |
|    | <b>Record Protocol</b><br>Start and stop recording.                           |   |  |
|    | <b>Execute Protocol</b><br>Execute entire protocol.                           |   |  |
|  | <b>Execute Protocol stepwise</b><br>Execute protocol step by step.            |   |  |

## Menu



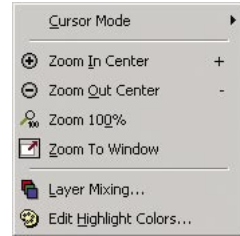
### Menu items in "Project":

<b>New...</b>	Create a new eCognition project and load raster and thematic layers	
<b>Open...</b>	Open an existing eCognition project	
<b>Close</b>	Close the current project	
<b>Save</b>	Save the current project to disk	
<b>Save as...</b>	Save the current project to another file	
<b>Project Information...</b>	Display information about the currently loaded project	
<b>User Information</b>	Display information about the person currently locked on the machine	
<b>Add Image Layer...</b>	Load additional image layer to current project	
<b>Add Thematic Layer...</b>	Load additional thematic layer to current project	
<b>Assign layer alias</b>	Edit and assign aliases to the layers	
<b>Assign No Data Value</b>	Assign No Date Value	
<b>Exit</b>	Exit eCognition.	

### Menu items in "View":

<b>Cursor Mode</b>	Open the View Settings dialog
<b>Zoom In Center</b>	Zoom in at current view center

<b>Zoom Out Center</b>	Zoom out at current view center
<b>Zoom 100 %</b>	Reset zoom factor to 100 %.
<b>Zoom to Window</b>	Zoom the scene to fit into the current window.
<b>Layer Mixing...</b>	Define a color composition for the display
<b>Edit Highlight Colors...</b>	Select highlight color settings and the display color of nonsample and unclassified objects.



#### Menu items in "Image Objects":

<b>Multiresolution Segmentation...</b>	Open the "Multiresolution Segmentation" dialog and initiate segmentation.	
<b>Scale Parameter Analysis</b>	Open the "Scale Parameter Analysis" dialog to support the investigation of the scale parameter.	
<b>Export Scale Parameter Analysis</b>	Open the „Export Scale Parameter Analysis“ dialog to export the results	
<b>Classification-based segmentation</b>	Open the "Classification-based Segmentation" dialog and initiate segmentation.	
<b>Create Polygons...</b>	Vectorize objects of specified object level.	
<b>Delete Polygons...</b>	Delete polygons of selected level.	
<b>Classification-based object cut...</b>	Open dialog to select classes and parameters for the automated object cut.	
<b>Delete Level...</b>	Delete a level of the image object hierarchy.	

### Menu items in "Samples":

**Select Samples** Switches the sample input mode

**Sample Editor** Opens the Sample Editor

**Sample Editor Options** Displays further sample functions in the sub-menu.

**Delete Samples of Classes...** Delete samples of selected classes.

**Delete all Samples** Delete all samples.

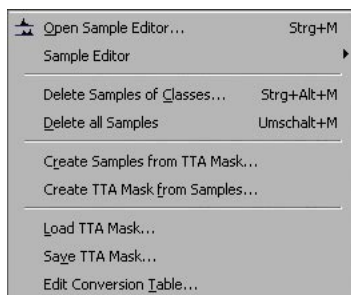
**Create Samples from TTA Mask...** Use a TTA mask to declare sample image objects.

**Create TTA Mask from Samples...** Create a TTA mask out of existing sample objects.

**Load TTA Mask...** Load TTA mask image and conversion table from disk.

**Save TTA Mask...** Save TTA mask layer and conversion table to disk.

**Edit Conversion Table...** Display and edit the conversion table.



### Menu items in "Classification":

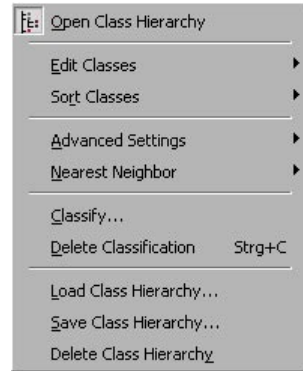
**Open Class Hierarchy** Opens the "Class Hierarchy" dialog.

**Edit Classes** Functions to edit, insert and copy classes.

**Sort Classes** Functions to sort classes in the class hierarchy.

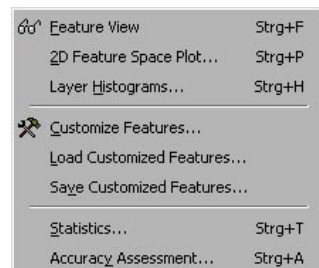
**Advanced Settings** Further settings for brightness layers and minimum membership values.

<b>Nearest Neighbor</b>	“Edit Standard Nearest Neighbor Feature Space...,” “Apply Standard NN to classes...,” “Slope Function...”
<b>Classify...</b>	Open the “Classification Settings” dialog and invoke image object classification.
<b>Delete Classification</b>	Delete classification of all image objects.
<b>Load Class Hierarchy...</b>	Load a Class Hierarchy from a file.
<b>Save Class Hierarchy...</b>	Save class hierarchy to a file.
<b>Delete Class Hierarchy...</b>	Delete the existing class hierarchy.



#### Menu items in “Tools”:

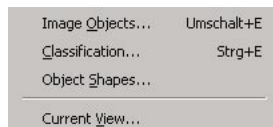
<b>Feature View... 2D Feature Space Plot...</b>	Render the scene using a specific object feature. 2D plot of selected features.
<b>Image Layer Histograms</b>	Shows the histograms of the scene
<b>Feature Space Optimization</b>	Opens the Features Space Optimization dialog
<b>Scale Parameter Analysis</b>	Open the Scale Parameter Analysis dialog
<b>Customize Features...</b>	Create arithmetic and relational new features for image objects.



<b>Load Customized Features...</b>	Load previously created customized features from a file.
<b>Save Customized Features...</b>	Save definitions of new created customized features to a file.
<b>Statistics...</b>	Statistical evaluations of image object features.
<b>Accuracy Assessment...</b>	Statistical information on the actual classification.
<b>Manual Image Object Editing</b>	Opens the sub-menu for the manual editing modes: normal selection, polygon, line, rectangular selection; image object cut manual classification and image object fusion
<b>Options</b>	Opens the Options dialog

#### Menu items in "Export":

<b>Image Objects...</b>	Export image object level as a thematic layer (raster or vector format).
<b>Classification...</b>	Export classification of current image object level as image.
<b>Current View...</b>	Copy current view to clipboard.
<b>Object Shapes...</b>	Export points, lines or polygons of objects belonging to a selected class.

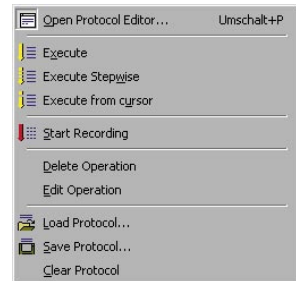


#### Menu items in "Protocol":

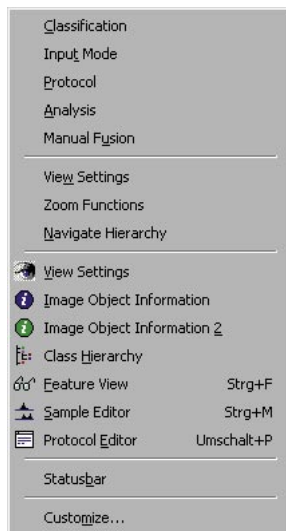
<b>Open Protocol Editor...</b>	Open the dockable dialog box "Protocol Editor."
<b>Execute</b>	Execute the entire protocol.
<b>Execute Stepwise</b>	Execute protocol step by step.
<b>Execute from cursor</b>	Execute protocol from the cursor position downwards.
<b>Start Recording</b>	Record new protocol.



<b>Delete Operation</b>	Delete an operation in the current protocol.
<b>Edit Operation</b>	Edit parameters of protocol operation.
<b>Load Protocol</b>	Load a protocol from disk.
<b>Save Protocol</b>	Save current protocol to disk.
<b>Clear Protocol</b>	Delete all entries in the protocol.



#### Menu items in "Toolbars & Dialogs":

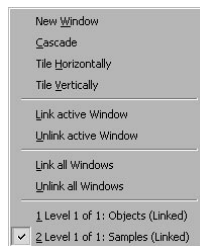


All menu items open or close the specified dialogs.

#### Menu items in "Window":

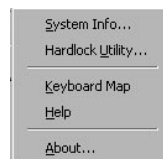
<b>New Window</b>	Open a new scene window.
<b>Cascade</b>	Arranges the windows as overlapping tiles.

<b>Tile Horizontally</b>	Arranges the windows as nonoverlapping, horizontal tiles.
<b>Tile Vertically</b>	Arranges the windows as nonoverlapping, vertical tiles.
<b>Link active Window</b>	Synchronize the display area of other views with the active window.
<b>Unlink active Window</b>	Delete the display synchronization between the active window and another view
<b>Link all Windows</b>	Synchronize the display area of all views.
<b>Unlink all Windows</b>	Delete the display synchronization between all views.



#### Menu items in "Help":

<b>System Info...</b>	Display system configuration and state information.
<b>Test Hardlock</b>	Open dialog "Test Hardlock" to check hardlock properties.
<b>Keyboard Map</b>	Overview of all eCognition short cuts.
<b>Help</b>	Open eCognition User Guide.
<b>About...</b>	Application info.

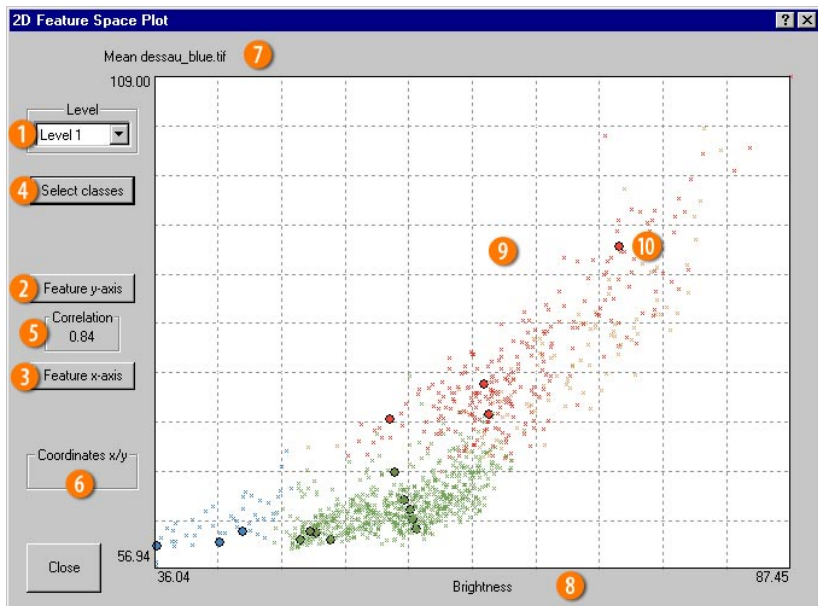


## Dialogs

### 2D Feature Space Plot

This dialog box helps with checking the distribution of feature values over two different features.

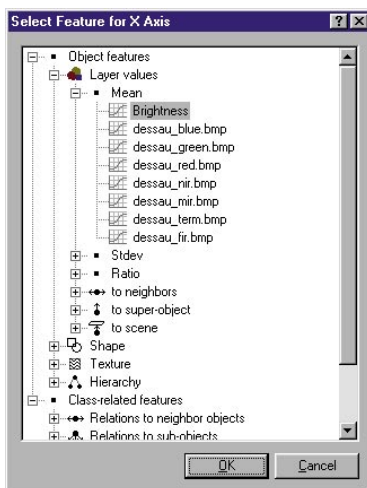
To open this dialog choose the item “Tools > 2D Feature Space Plot...” from the menu bar of the main application window.



- 1 Select the image object level for which feature values are to be plotted.
- 2 Click here to select the feature to be plotted along the y-axis (see dialog below).
- 3 Click here to select the feature to be plotted along the x-axis (see dialog below).
- 4 Click here to select the classes for the feature space plot in their feature range.
- 5 This field displays the correlation (Pearson) between the values of the selected features.
- 6 In this field the mouse position in the feature space plot is displayed.

- 7 Name of the feature plotted along the y-axis.
- 8 Name of the feature plotted along the x-axis.
- 9 The colored circles show the selected samples (sample editor).
- 10 Navigate to the sample by clicking on it (Sample Navigation must be activated)

Select the feature to be plotted along the y-axis 2 or x-axis 3 in the following dialog by double-clicking or highlighting it and clicking “OK.”

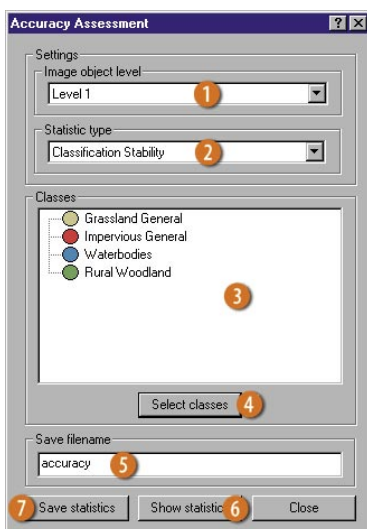


## Accuracy Assessment

This dialog box provides information about the classification quality.

To open this dialog box choose the item “Tools > Accuracy Assessment...” from the menu bar of the main application window.

- 1 Select the image object level to perform the accuracy assessment on.
- 2 Select the type of accuracy assessment.
- 3 This list shows the classes that are used for accuracy assessment.
- 4 Click here to edit the class list above (see User Interface > Select Classes).
- 5 Export file name of the statistic results.
- 6 Click this button to display the accuracy assessment results.

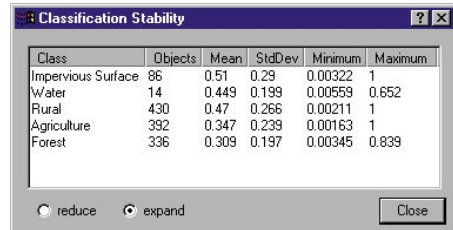


- 7 Click this button to save the accuracy assessment results to a file.

For a detailed description of the following dialogs and methods see also [Functional Guide > Accuracy assessment and statistics](#).

“Classification Stability”:

Difference between the best and the second best class assignment calculated as a percentage. The statistical output displays basic statistical operations (number of image objects, mean, standard deviation, minimum value and maximum value) performed on the best-to-second values per class.



Classification Stability

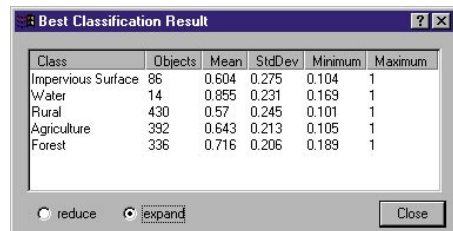
Class	Objects	Mean	StdDev	Minimum	Maximum
Impervious Surface	86	0.51	0.29	0.00322	1
Water	14	0.449	0.199	0.00559	0.652
Rural	430	0.47	0.266	0.00211	1
Agriculture	392	0.347	0.239	0.00163	1
Forest	336	0.309	0.197	0.00345	0.839

☐ reduce ☒ expand Close

To display the comparable graphical output, choose “View Settings > Mode > Classification Stability.”

“Best Classification Result”:

Statistical output for the best classification result evaluated per class. Basic statistical operations are performed on the best classification result of the image objects assigned to a class (number of image objects, mean, standard deviation, minimum value and maximum value).



Best Classification Result

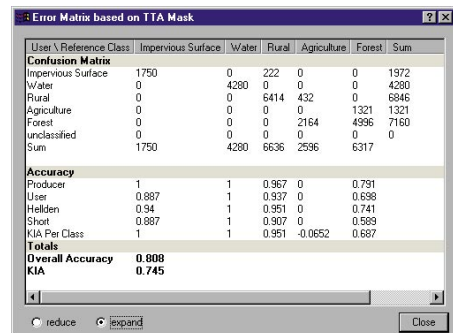
Class	Objects	Mean	StdDev	Minimum	Maximum
Impervious Surface	86	0.604	0.275	0.104	1
Water	14	0.855	0.231	0.169	1
Rural	430	0.57	0.245	0.101	1
Agriculture	392	0.643	0.213	0.105	1
Forest	336	0.716	0.206	0.189	1

☐ reduce ☒ expand Close

To display the comparable graphical output, choose “View Settings > Mode > Best Classification Result.”

„Error Matrix based on TTA Mask”:

Test areas are used as a reference to check classification quality by comparing the classification with ground truth based on pixels.



Error Matrix based on TTA Mask

User \ Reference Class	Impervious Surface	Water	Rural	Agriculture	Forest	Sum
<b>Confusion Matrix</b>						
Impervious Surface	1750	0	222	0	0	1972
Water	0	4280	0	0	0	4280
Rural	0	0	6414	432	0	6846
Agriculture	0	0	0	0	1321	1321
Forest	0	0	0	2164	4996	7160
Unclassified	0	0	0	0	0	0
Sum	1750	4280	6636	2596	6317	
<b>Accuracy</b>						
Producer	1	1	0.967	0	0.791	
User	0.887	1	0.937	0	0.638	
Hellden	0.94	1	0.951	0	0.741	
Short	0.887	1	0.907	0	0.583	
KIA Per Class	1	1	0.951	0.0652	0.687	
<b>Totals</b>						
<b>Overall Accuracy</b>	<b>0.808</b>					
<b>KIA</b>	<b>0.745</b>					

☐ reduce ☒ expand Close

“Error Matrix based on Samples”:

Similar to “Error Matrix based on TTA Mask” but considers samples (not pixels) deri-

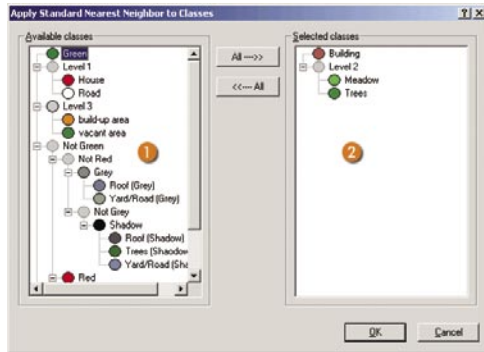
ved from manual sample inputs. The match between the sample objects and the classification is expressed in parts of class samples.

Error Matrix based on Samples							
Class	Samples	Impervious Surface	Water	Rural	Agriculture	Forest	unclassified
Impervious Surface	12	0.833	0	0	0	0	0.167
Water	3	0	1	0	0	0	0
Rural	8	0	0	0.125	0.75	0.125	0
Agriculture	6	0	0	1	0	0	0
Forest	8	0	0	0	0.125	0.875	0

## Apply Standard Nearest Neighbor to Classes

To open this dialog box choose the item “Classification > Nearest Neighbor > Apply Standard NN to Classes” from the menu bar of the main application window.

- 1 All available classes in the project. Single-click on class to switch to “Selected classes.”
- 2 All selected classes where the standard nearest neighbor will be applied. Single-click to deselect them.



## Apply TTA Mask to Level

Choose in this dialog to which level a loaded TTA mask is to be applied.

To open this dialog box, choose the item “Samples > Create Samples from TTA Mask...” from the menu bar of the main application window.

Mark the image object level where the TTA mask is to be applied (no multiple selection possible).



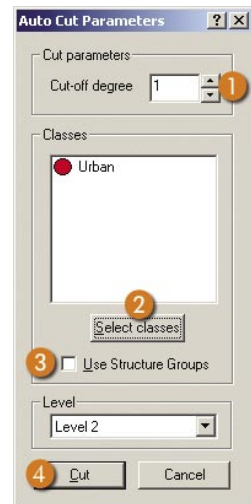
## Auto Cut Parameters

Use this dialog to cut-off outer parts of image objects, which consequently leads to a less complex geometry of the objects. Select the classes or objects you want to perform the automated cut on and choose the cut-off degree that determines the degree of branching to cut-off.

To open this dialog, select the item “Image Objects > Classification-based object cut...” from the menu bar of the main application window.

**Note!** you have to create polygons before an automated object cut can be performed.

- 1 Enter the cut-off degree that determines the degree of branching to cut-off.
- 2 Select the classes to which the automated cut is to be performed on the corresponding image objects.
- 3 Activate the radio button to perform the automated cut on the image objects defined by the structure groups.
- 4 Carry out the automated cut.

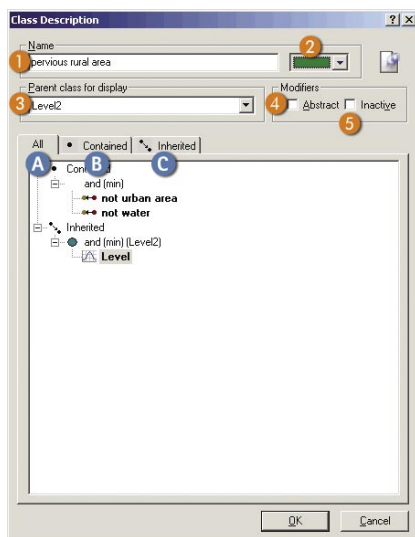


## Class Description

This dialog box is used to create and edit a class description.

To open this dialog box, double-click on a class in the “Class Hierarchy: Inheritance” dialog or select the item “Classification > Edit Classes > Edit Class” from the menu bar of the main application window.

- A** Displays the contained as well as the inherited part of the class description.
- B** Displays only the contained part of the class description.
- C** Displays only the inherited part of the class description.




- 1** Current name of the class.
- 2** Current color of the class.
- 3** Select the display parent group for navigating up the semantic groups hierarchy (right green arrow). This function is only necessary when a class belongs to multiple parent classes in the groups hierarchy.
- 4** Change class status to abstract. Abstract classes do not apply themselves to image objects directly. They only inherit or pass on their class descriptions to child classes.
- 5** Makes a class inactive. Inactive classes are ignored in the classification process.

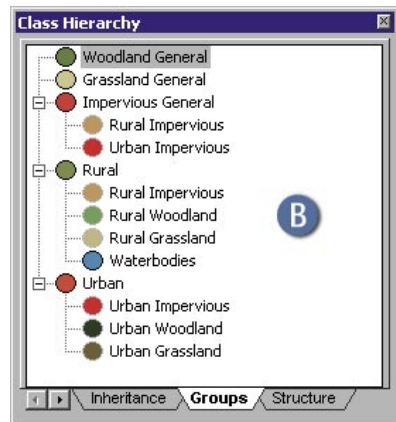
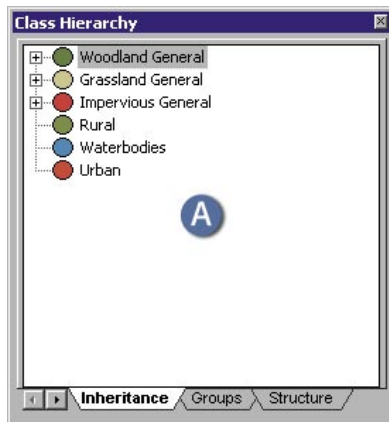


## Class Hierarchy

This dialog box is used to create and edit a class hierarchy.

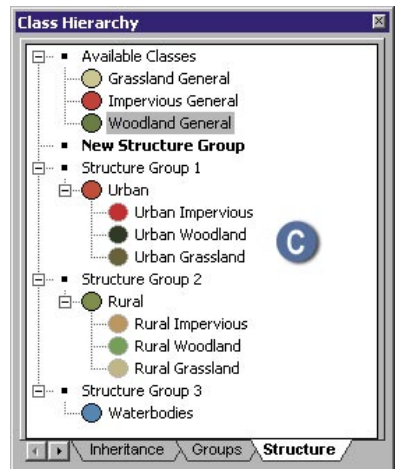
To open this dialog box either

- click the button  on the tool bar of the main application window.
- choose between the items “Classification > Open Class Hierarchy...” and “Toolbars & Dialogs > Class Hierarchy” from the menu bar of the main application window.
- choose the items Image Objects > Classification-based Segmentation” or Image Objects > Edit Structure Groups...” from the menu bar of the main application window and automatically the dialog box with the register tab “Structure” is opened.



- A** Displays the inheritance of the class hierarchy for the purpose of classification. Right-click for context menu. Drag a class with the left mouse button to move it. Drag a class with the right mouse button for multiple inheritance.

- B** Displays the group structure of the class hierarchy for the purpose of classification. Use the right mouse button for context menu. Drag a class with the left mouse button to

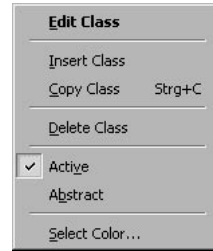


move it. Drag a class with the right mouse button for a multiple membership to more than one semantic group.

- C** Displays the structure groups for the purpose of classification-based segmentation. To create a new structure group, drag a class and drop it on “New Structure Group.” Drag a class and drop it on an existing structure group to add it to that structure group.


**Menu items of the “context menu” (click right mouse button):**

<b>Edit Class</b>	Open the “Class Description” dialog of selected class.
<b>Insert Class</b>	Create a new class and insert it into the class hierarchy.
<b>Copy Class</b>	Copy selected class.
<b>Delete Class</b>	Delete selected class.
<b>Active</b>	Activate/deactivate selected class. Inactive classes are ignored in the classification process.
<b>Abstract</b>	Change to an abstract class. Abstract classes do not apply themselves to image objects directly. They only inherit their class descriptions.
<b>Select Color</b>	Choose a class in the class hierarchy and select the display color.

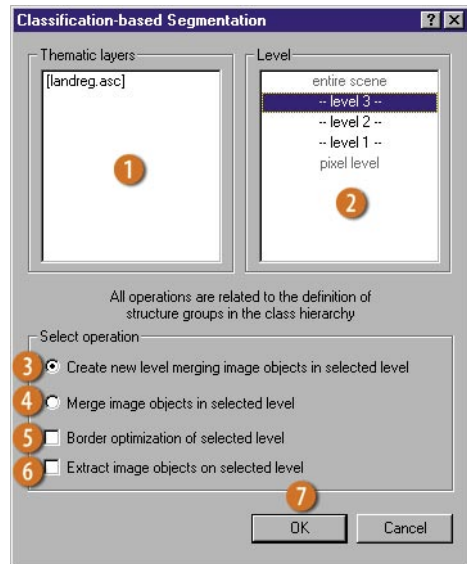


## Classification-based Segmentation

Use this dialog box to edit the parameters and to determine the type of a classification-based creation of image objects.

To open this dialog box choose the item “Image Objects > Classification-based Segmentation...” from the menu bar of the main application window or click  in the tool bar.

- 1 List of the thematic layers that are regarded during classification-based segmentation. A thematic layer will be used during the segmentation process, if it is highlighted blue. To activate or deactivate a thematic layer, simply click on it.
- 2 Select the image object level on which you want to perform the classification-based segmentation.
- 3 Check this button to create a new image object level on which the classification-based fusion is executed.
- 4 Check this button to fuse the image objects to the selected image object level. The old image object level will be deleted.
- 5 Check this button to enable border optimisation of the selected image object level. You need a level of sub-objects to perform this operation.
- 6 Check this button to enable image object extraction on the selected image object level. You need a level of sub-objects to perform this operation.
- 7 Start the classification-based segmentation with the current settings.



## Classification Settings

This dialog box is used to edit the classification parameters and start the classification process.

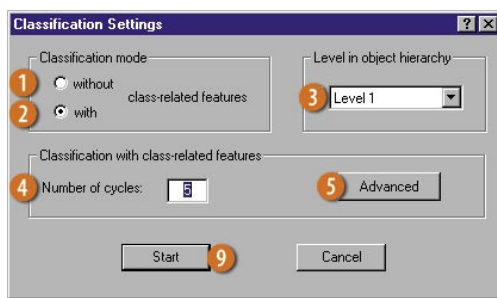
To open this dialog box, choose the item “Classification > Classify...” from the menu bar of the main application window or click  in the tool bar.

- 1 Only use classes without any class-related features for classification.
- 2 Also use classes with class-related features for classification.

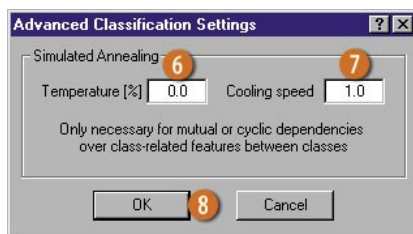
- 3 Select the image object level to be classified.

- 4 Define the number of classification cycles.

- 5 Click the button “Advanced” to open the dialog box “Advanced Classification Settings.”



- 6 Initial random deviation of the membership value incorporated in the classification result. Range = [0.0,...,100.0]. A value of 10 means: that the assigned membership value may be higher or lower by a factor of 10 (i.e. 0.1) than the computed value.

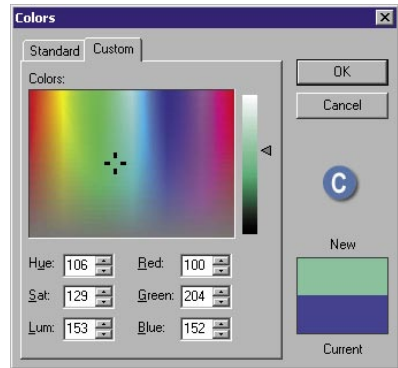
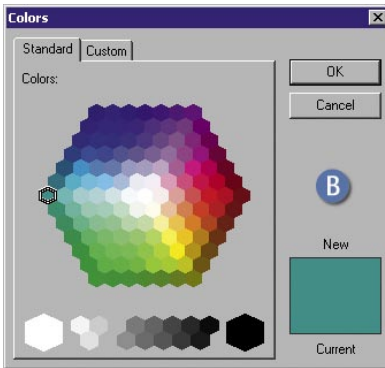
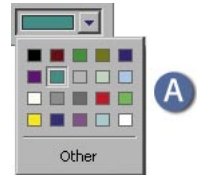


- 7 Reduces the initial random deviation in each cycle. Range = [0,...,number of cycles]. The number entered is the number of cycles until the random deviation reaches 0. Example: With 10 cycles a cooling speed of 6 means that after each cycle the deviation is reduced by  $\frac{1}{46}$  and no more deviation is applied after the sixth cycle.
- 8 Click the button “OK” to close this dialog box.
- 9 Start the classification process with current parameters.

## Colors

Use this color combo box to edit the color information of classes and all highlight colors.

To open this combo box check the color list item in the “Class Description” dialog or double-click in the left window of the “View Settings” dialog.



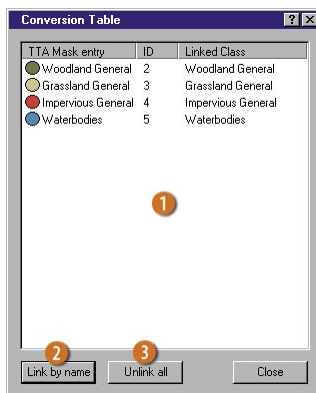
- A** Pick a standard color in the color combo box. Click “Other” to get more predefined colors.
- B** Dialog to get more predefined standard colors. Choose Custom to define your own colors.
- C** Dialog to define your own color mixing for classes and highlight colors.

## Conversion Table

This dialog is used to display and edit the linkage between the classes of the project and the classes of a training and test area mask.

To open this dialog choose the item “Samples > Edit Conversion Table...” from the menu bar of the main application window.


- 1 This list displays how classes of the project are linked to classes of the training and test area mask. To edit the linkage between the TTA mask classes and the classes of your project right click on a TTA mask entry and connect it with the appropriate class in the project.
- 2 Click here to link all class names automatically by identical names.
- 3 Click here to unlink all classes.



## Create Customized Feature

This dialog box is used to create and edit customized features. There are two kinds of features: Arithmetic and relational expressions.

To open this dialog box either

- click the button  on the tool bar of the main application window.
- choose the item “Tools > Customized Features...” from the menu bar of the main application window.

To edit or delete a customized feature, this menu is activated by a right click on the feature name in the “Feature View > Edit Feature” or “Delete Feature.”

## Create Arithmetic Features

- 1 Feature name of the new created attribute.
- 2 Display the arithmetic expression.
- 3 Calculate and delete arithmetic expressions.

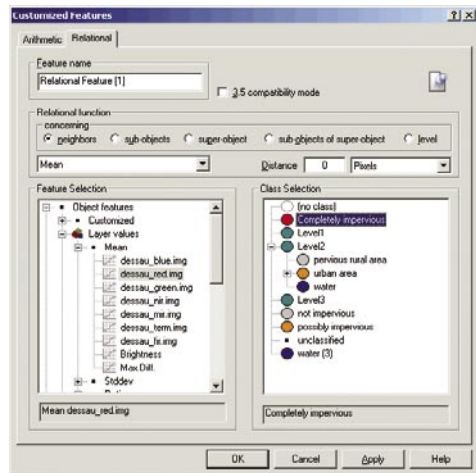


To use these buttons an expression in the feature calculator display has to be marked with the cursor.

- 4 Numeric calculator. Click here to insert arithmetic expressions.
- 5 Click here to select any image object feature or class related feature for the arithmetic expression.
- 6 Click here to create the new customized feature without leaving the dialog.


### Create Relational Features

- 1 Feature name of the customized relational feature.
- 2 Check to define a relational feature referring to neighbors.
- 3 Check to define a relational feature referring to sub-objects.
- 4 Check to define a relational feature referring to a super-object
- 5 Check to define a relational feature referring to sub-objects of a super- object.
- 6 Check to define a relational feature referring to level.
- 7 Select the relational function.
- 8 Define the vertical distance (layers) or horizontal distance (pixels).
- 9 Select the feature for which to compute the relation.
- 10 Select a class (group) or “No class” for the relation.

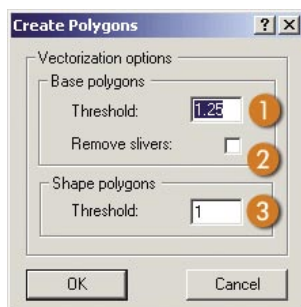


## Create Polygons

Use this dialog boxes to create polygons from the objects in the selected level.


To open this dialog box, choose the item “Polygons > Create Polygons” from the menu bar of the main application window or select  in the tool bar.

- 1 Edit the threshold for the generation of base polygons
- 2 Activate if sliver removal is desired
- 3 Edit the threshold for the generation of shape polygons



## Create Project

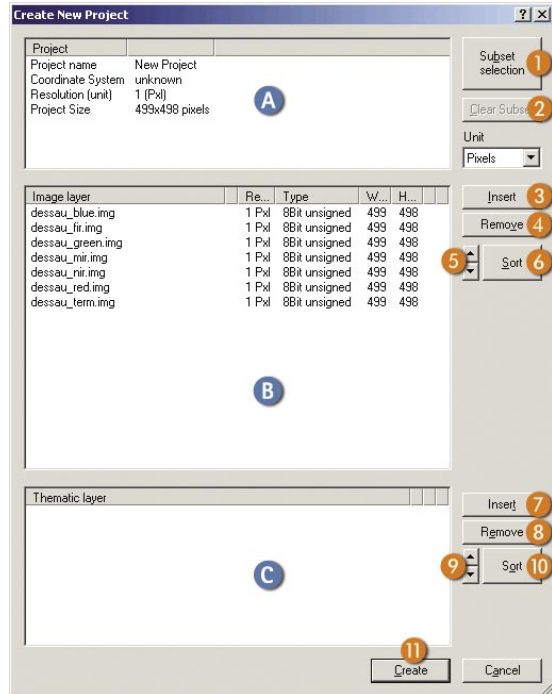
Use this dialog to import and sort the image and thematic layers into the project, define a subset, and in addition edit the georeferenced information of the layers and their aliases.

To open this dialog, select the item “Project > New...” from the menu bar of the main application window or click  in the tool bar.

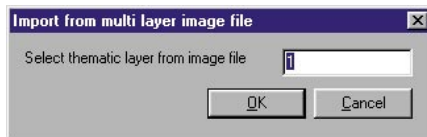
- A Project header information.
- B Displays the image layers selected for loading along with their properties.
- C Displays the thematic layers selected for loading together with their attribute tables.
- 1 Definition of image subset.
- 2 Clear subset definition.
- 3 Insert new image layers to the project.
- 4 Remove the selected image layers from the project.
- 5 Move the selected image layers up or down in the image layer order.



- 6 Sort image layers alphabetically. To do this mark which layers are to be sorted.
- 7 Insert a new thematic layer to the project.
- 8 Remove selected thematic layers from the project.
- 9 Move the selected thematic layers up and down in the thematic layer order.
- 10 Sort the selected thematic layers alphabetically.
- 11 Create the project. At least one image layer must be used.
- 12 Adjust the measurement unit of the project



When loading a thematic layer from a multilayer image file (e.g. \*.img stack file), the appropriate layer that corresponds with the thematic information is requested in the following dialog.

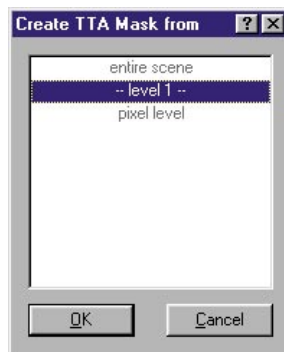


## Create TTA Mask from Level

Choose in this dialog from which level a TTA Mask is to be created based on inserted samples.

To open this dialog box, choose the item “Samples > Create TTA Mask from Samples...” from the menu bar of the main application window.

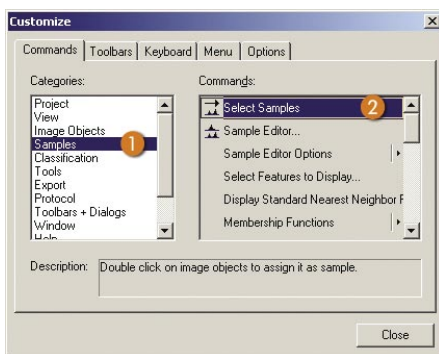
Mark the image object level from where the TTA Mask is to be created (no multiple selection possible).



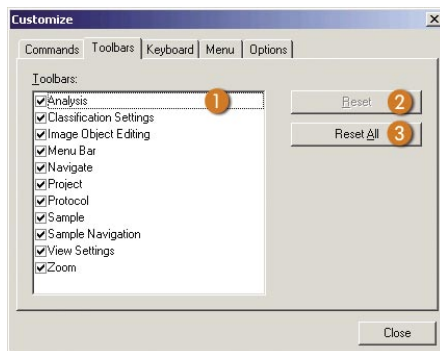
## Customize

This menu enables customization and reset of toolbars, keyboard shortcuts and menus. The “Customize” menu enables a moving of toolbar buttons by “drag and drop.”

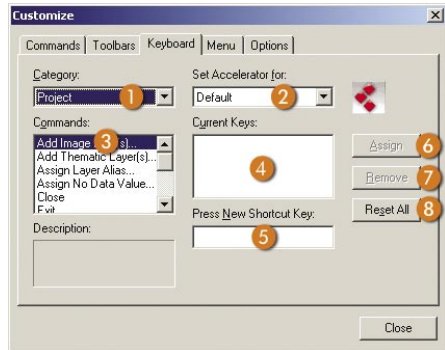
- 1 All menu bar categories.
- 2 All eCognition commands which can be dragged into the toolbar by holding down the left mouse button.



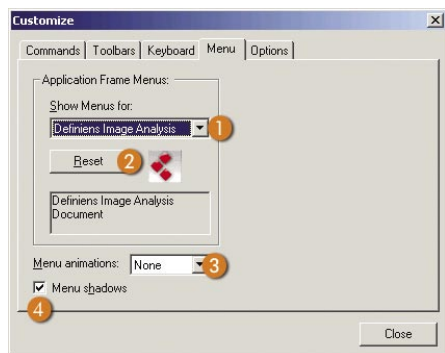
- 1 All toolbars can be switched on and off.
- 2 The changes for the activated toolbar are set to default settings.
- 3 All changes in all toolbars are set to default settings.



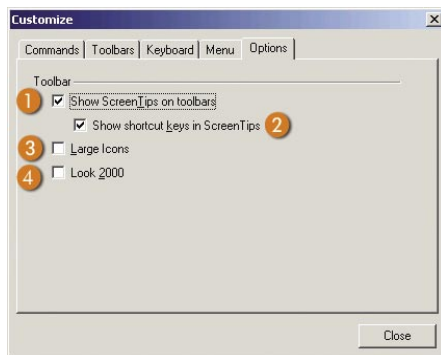
- 1 Categories of the menu bar.
- 2 Set shortcuts for the default appearance of eCognition. No other selection is possible.
- 3 All eCognition commands in the selected menu bar category. An accelerator key can be assigned for the activated command.
- 4 Display the current accelerator key(s).
- 5 Display the newly assigned shortcut. Changes can only be made if the cursor is positioned in this field.
- 6 Click here to assign the new accelerator key to the eCognition command.
- 7 Click here to remove the accelerator key from the eCognition command.
- 8 Click here to reset all accelerator keys to default values.



- 1 "Default Menu" entry not yet used. Please use the "eCognition" menu bar entry.
- 2 Click here to reset all items in the menu bar.
- 3 Set desired menu animations
- 4 Activate for menu shadows



- 1 Activate/deactivate tool tip display.
- 2 Activate/deactivate shortcut key display in tool tips.
- 3 Activate to display large toolbar icons.
- 4 Activate for Windows 2000 look

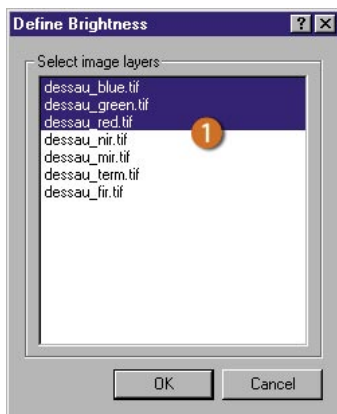


## Define Brightness

Use this dialog box to select those image layers from which the brightness is to be calculated.

To open this dialog box, choose the item “Classification > Advanced Settings > Select Image Layers for Brightness” from the menu bar of the main application window.

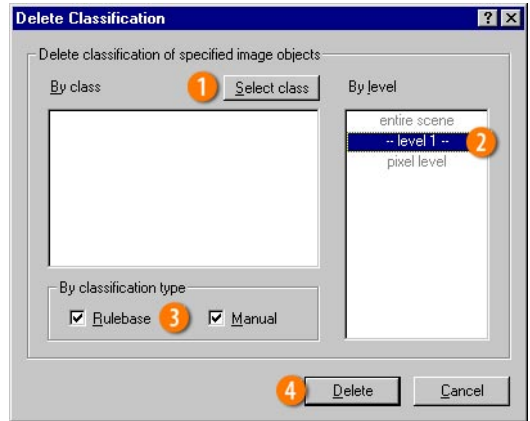
- 1 Mark image layers. The brightness will be calculated using the marked image layers only.



## Delete Classification

Use this dialog to delete the classification of all image objects or of specified image objects.

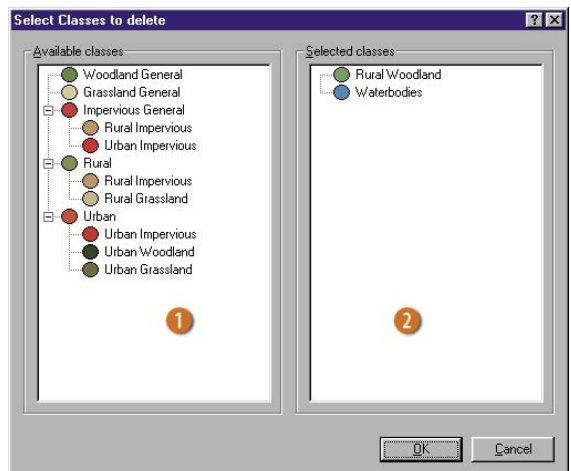
To open this dialog box choose “Classification > Delete Classification” from the menu bar of the main application window.



- 1 Deletes the classification of specified classes; to select the classes, click “Select class” (see dialog below).
- 2 Delete the classification of a selected level.
- 3 Decide whether only the manual and/or the classification based on your rule base is to be deleted.
- 4 Deletes the classification of the specified image objects.

In this dialog you can select specific classes for which the classification is to be deleted:

- 1 This list displays all available classes where the classification will not be deleted. Single-click a feature to select it.
- 2 This list displays the selected features where the classification is to be deleted. Single-click a feature to deselect it.

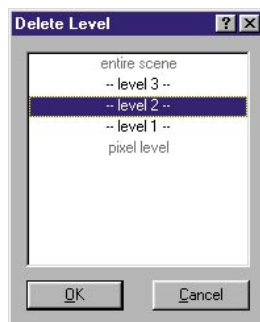


## Delete Level

Use this dialog box to delete a level.

To open this dialog box, choose the item “Image Objects > Delete Level(s)” from the menu bar of the main application window.

Mark the image level(s) to be deleted (no multiple selection possible).



## Delete Samples of Selected Classes

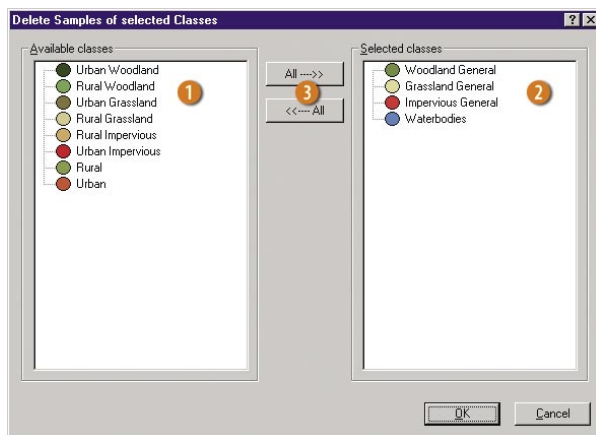
In this dialog box select the classes for which samples are to be deleted.

To open this dialog box choose the item “Samples > Delete Samples of Classes...” from the menu bar of the main application window.

1 All available classes in the project. Single-click on class to switch to “Selected classes.”

2 All selected classes of which the samples will be deleted. Single-click to deselect them.

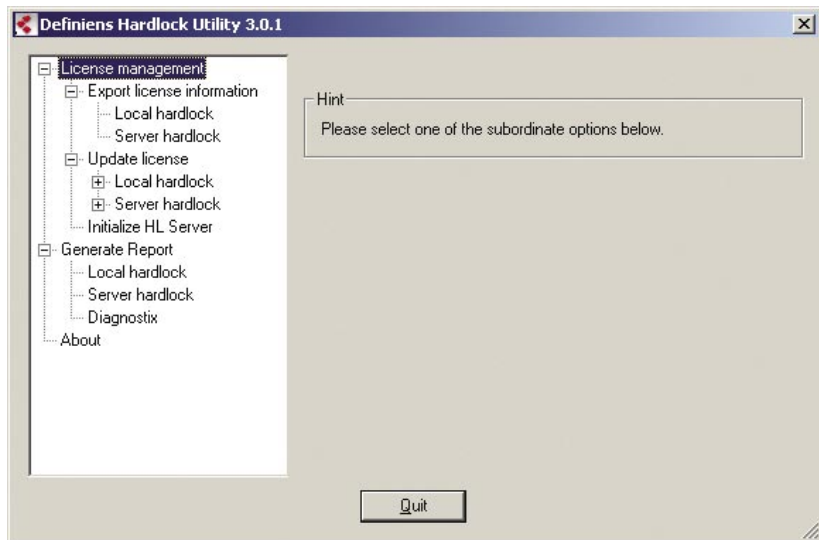
3 Move all classes from or to the selection list



## eCognition Hardlock Utility

This dialog can be used to test the eCognition hardlock and to view hardlock internal information. Additionally, this utility can be used to install or remove the hardlock server software.

To invoke this dialog choose “Start > Program > eCognition > Hardlock Utility” in the Windows menu bar or choose “? > Hardlock Utility...” from the main application window.

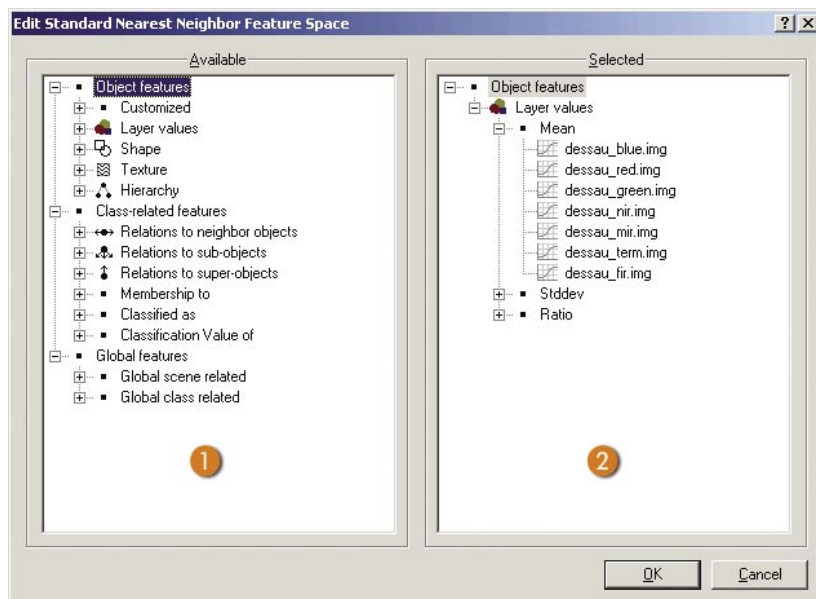


See the chapter „Installing eCognition“ for more details on using the Hardlock Utility Software.

## Edit Standard Nearest Neighbor Feature Space

Use this dialog box to edit the standard nearest neighbor feature space.

To open this dialog box choose the item “Classification > Nearest Neighbor > Edit Standard NN Feature Space” from the menu bar of the main application window.




- 1 List of all selectable features for the standard nearest neighbor feature space. Double-click a feature to select it
- 2 This list displays the selected features that form the standard nearest neighbor feature space. Double-click a feature to deselect it. Defaults are all mean values of the image layers.

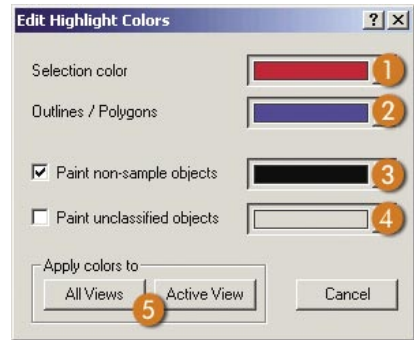


## Edit Highlight Colors

This dialog enables the selection of highlight color settings and the display color of nonsample and unclassified objects.

This dialog can be activated by “View > Edit Highlight Colors...” from the main menu bar, by the  button from the tool bar, or by double-clicking in the left window of the “View Settings” dialog.


- 1 Color of selected image objects.
- 2 Display color of computed polygon outlines
- 3 Color of nonsample objects in the “Sample View” display mode.
- 4 Color of unclassified objects in “Classification View” display mode.
- 5 Click “All Views” or “Active View” to apply changes to either all open views or the active view only. Exit the dialog.

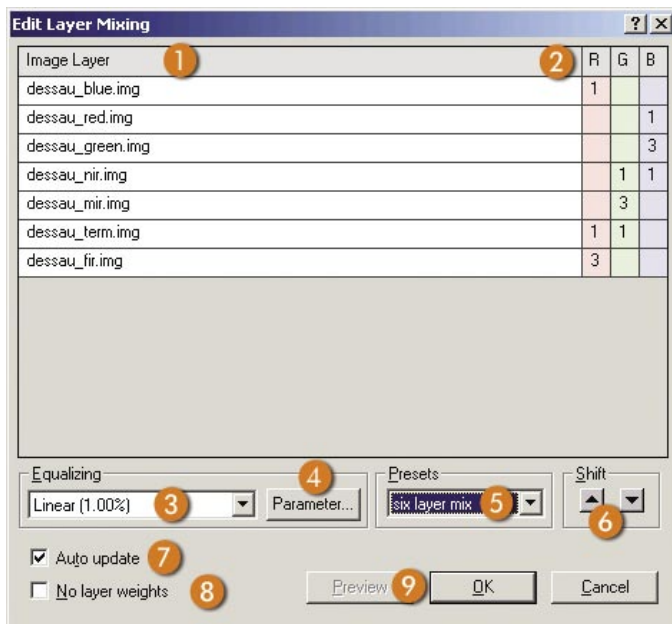


## Edit Layer Mixing

Use this dialog box to define a color composition for the display of image objects. It is possible to assign more than one layer to a color.

To open this dialog box either

- click the  button from the tool bar of the main application window.
- click the item “View > Layer Mixing” from the menu bar of the main application window.
- choose the dialog box “View setting” and double-click on the dialog window.



- 1 Name of the image layer(s).
- 2 Select layer(s) to be colored in red, green and blue on the screen.
- 3 Choose display-equalizing algorithm.
- 4 Insert desired equalization parameter.
- 5 Choose display layer mixing presets.
- 6 Shift between image layers.
- 7 Switch between immediate display update and display update by the „Apply“ button.
- 8 Switch between channel weights and unweighted display of channels.
- 9 Activates a display update; only active if auto update switch is off.

## Edit Minimum Membership Value

The minimum membership defines the minimum value for the classification. If the membership value of a classified image object is lower than this value, the object remains unclassified.

To open this dialog box select “Classification > Advanced Settings > Minimum Membership Value...” from the main menu bar.

- 1 Minimum membership value for classification. All lower values are defined as unclassified.

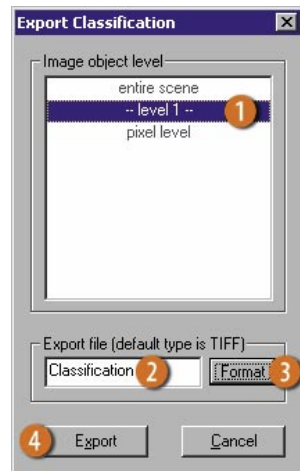


## Export Classification

This dialog box is used to export the classified image as a raster file (\*.tif, \*.img., \*.asc).

To open this dialog box, select the item “Export > Classification...” from the menu bar of the main application window.

- 1 Select classified image object level to be exported.
- 2 Choose the file's name.
- 3 Click to choose the file format for export. Default is TIFF format.
- 4 Click to export the classification.

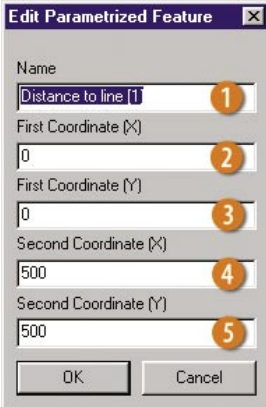


## Edit Parametrized Feature

This kind of dialog could be used to change the parameters of some features like „Distance to line (1)“. Thus it is possible to adapt these features to the requirements of the analyst, for instance to define the flight line of an aircraft.

To open this dialog click on the corresponding feature with the right mouse button and choose the item “Edit Feature...” from the popup menu. It is sensible to copy the feature (right click and chose “Copy Feature”) before you change the parameters, so that the original, unchanged feature is still available.

Since there are a few parametrizable features implemented in eCognition the following explanations concern only the feature “Distance to line” ❶ as an example.



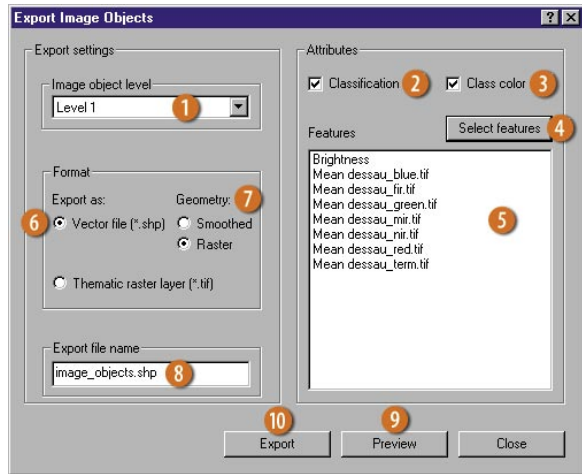
The image shows a screenshot of a software dialog box titled "Edit Parametrized Feature". It contains five input fields, each with a numbered orange circle next to it, indicating the steps for editing the feature. The fields are: 1. Name: "Distance to line 1" 2. First Coordinate (X): "0" 3. First Coordinate (Y): "0" 4. Second Coordinate (X): "500" 5. Second Coordinate (Y): "500" At the bottom of the dialog are two buttons: "OK" and "Cancel".

- ❶ Write a name for the feature.
- ❷ X-coordinate for first point of the line.
- ❸ Y-coordinate for first point of the line.
- ❹ X-coordinate for second point of the line.
- ❺ Y-coordinate for second point of the line.

## Export Image Objects

This dialog box is used to export an image object level as a thematic raster layer or as an ArcView vector file (\*.shp).

To open this dialog box, select the item “Export > Image Objects...” from the menu bar of the main application window.

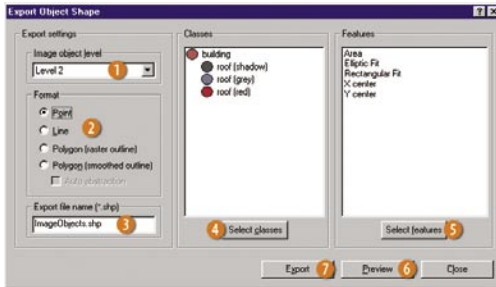


- 1 Select image object level to be exported as thematic raster layer or as ArcView polygon shape file.
- 2 Check to add the classification of the image objects as an export attribute.
- 3 Check to add the image object color of the current view as an export attribute.
- 4 Click to add and remove features to/from the attribute list.
- 5 Display the selected features for export.
- 6 Check to export as ArcView polygon shape file.
- 7 Check to define geometry smoothing. Raster geometry exactly follows the image pixels, whereas smoothed geometry avoids single pixel steps.
- 8 Choose the export file name.
- 9 Preview the export attribute table on the screen.
- 10 Save the geometry and attribute information to the disk.

## Export Image Object Shapes

This dialog box is used to export an image object level as a thematic raster layer or as an ArcView vector file (\*.shp).

To open this dialog box, select the item “Export > Image bects...” from the menu bar of the main application window.



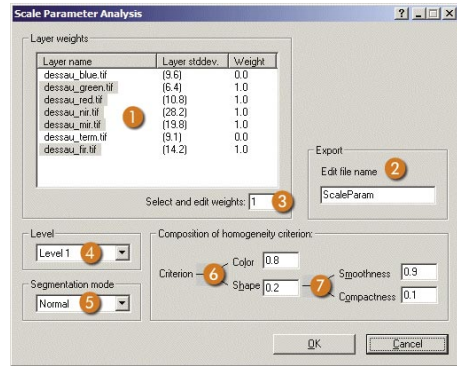
- 1 Select the image object level where the classes to be exported are situated
- 2 Choose the format whether to export polygons, points or lines. For the geometry of the polygons check either „Polygon (raster)“ or „Polygon (smoothed)“. As an additional feature you can choose auto abstraction for smoothed polygons. This yields a very high degree of abstraction that should produce very smooth borders (for instance to get roofs with only four edges).
- 3 The default name ImageObjects could be changed here. The results are stored in the directory where the project is saved.
- 4 Use this button to add or remove classes to be exported. The selected classes are displayed in the right area of the window that pops up. The available classes are displayed in the left area. They can be added and removed by a click with the left mouse button. With a right click it is possible to select a parent class including all child classes.
- 5 Use this button add or remove features from the attribute list. The selected features are displayed in the right area of the window that pops up. The available features are displayed in the left area.
- 6 A preview of the attribute table that is exported can be requested by choosing the „Preview“ button.
- 7 Choose „Export“ to save the geometry and the attribute information to the disk.

## Export Scale Parameter Analysis

Use this dialog to export the results from Scale Parameter Analysis as \*.csv files.


To open this dialog, select the item “Image Objects > Export Scale Parameter Analysis ...” from the menu bar of the main application window.

- 1 Select the layers for analysis.
- 2 Enter the file prefix.
- 3 Edit the channel weights.
- 4 Select the segmentation level.
- 5 Select the segmentation mode.
- 6 Select the color vs. shape weight
- 7 Select the smoothness vs. compactness weight.

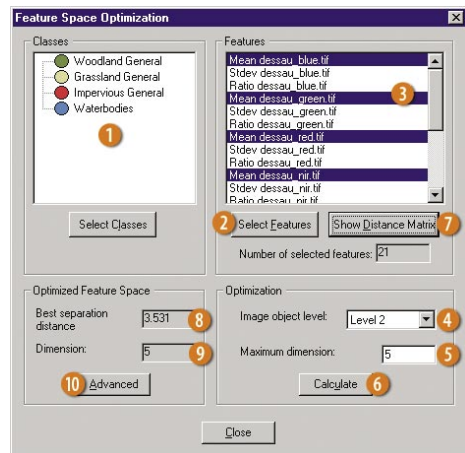


## Feature Space Optimization

With this dialog you control the feature space optimization.

To open this dialog box click the button  from the toolbar of the main application window

- 1 Select the classes for which you want to calculate the optimal feature space
- 2 Create the initial feature space to be reduced.
- 3 Select a subset of the initial feature space by clicking single features.
- 4 Select the image object level of concern.

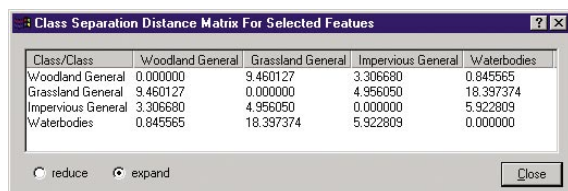


- 5 Enter the maximum dimension of the optimized feature space.
- 6 Click this button to generate feature combinations and their distance matrices.
- 7 Click this button to display the distance matrix of the currently selected feature combination.
- 8 Shows the largest distance between the closest samples of classes within the best separating feature space.
- 9 Shows the dimension of the best separating feature space.
- 10 Click this button to display advanced information about the results.

For a detailed description of the following dialogs and methods see also [Functional Guide > Classification Advanced > Basic strategies for creating class hierarchies > Finding the features that separate the classes at best.](#)

### Class Separation Distance Matrix For Selected Features

This matrix shows the distances between samples of the selected classes within a selected feature space.



Class/Class	Woodland General	Grassland General	Impervious General	Waterbodies
Woodland General	0.000000	9.460127	3.306680	0.845565
Grassland General	9.460127	0.000000	4.956050	18.397374
Impervious General	3.306680	4.956050	0.000000	5.922809
Waterbodies	0.845565	18.397374	5.922809	0.000000

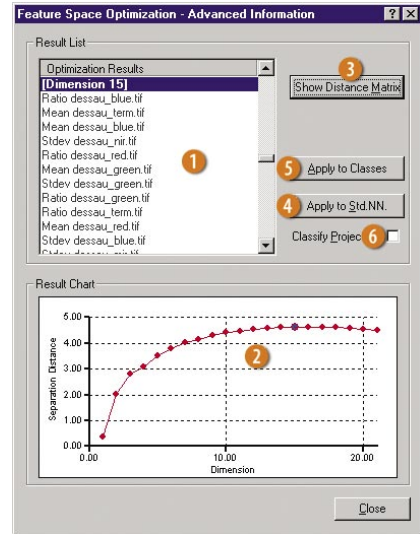
### Advanced

This dialog shows advanced information about all feature combinations and their separability of the classes' samples.

- 1 Select a feature space.
- 2 Shows the calculated maximum distances of the closest samples along the dimensions of the feature spaces. The blue dot marks the currently selected feature space. Click another dot to show other feature combinations.



- 3 Click this button to display the distance matrix of the currently selected feature combination.
- 4 Click this button to use the currently selected feature space for the standard nearest neighbor classifier.
- 5 Click this button to generate a nearest neighbor classifier using the current feature space for selectable classes.
- 6 Check this box to automatically classify the project when pressing “Apply to Std.NN.” or “Apply to Classes”.



### Class Separation Distance Matrix

This matrix shows the distances between samples of the selected classes within a selected feature space within the dialog “Feature Space Optimization - Advanced Information”. Select a feature combination and re-calculate the according distance matrix.

**Class Separation Distance Matrix**

Class/Class	Woodland General	Grassland General	Impervious General	Waterbodies
Dimension: 15				
Woodland General	0.000000	9.421040	4.644772	4.833777
Grassland General	9.421040	0.000000	8.830433	23.281022
Impervious General	4.644772	8.830433	0.000000	7.547036
Waterbodies	4.833777	23.281022	7.547036	0.000000

☐ reduce ☒ expand

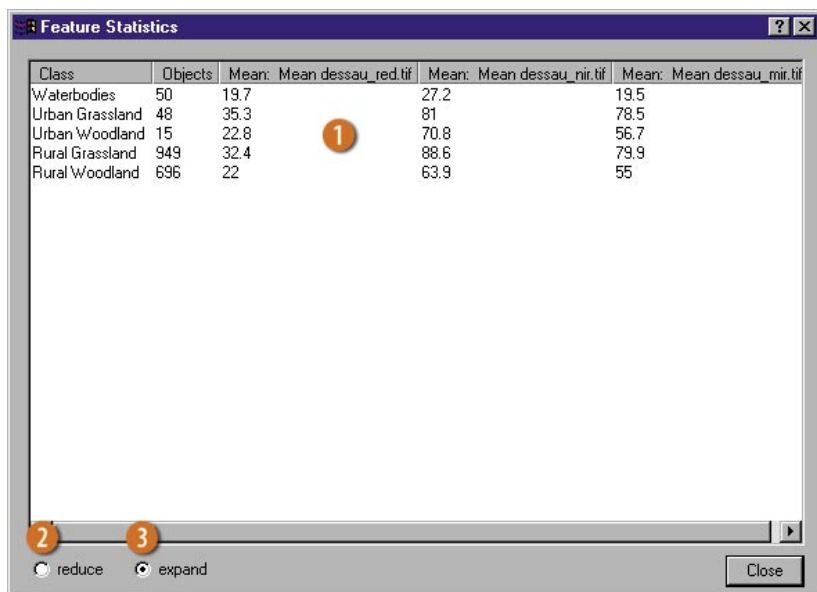
Close

## Feature Statistics

This dialog box displays the classification accuracy, feature statistics and object attributes in matrix form.

The dialog box is opened via the following dialogs:

- “Tools > Accuracy Assessment > Show statistics”
- “Tools > Statistics... > Show statistics”
- “Export > Image Objects > Preview”



Class	Objects	Mean: Mean dessau_red.tif	Mean: Mean dessau_nir.tif	Mean: Mean dessau_mir.tif
Waterbodies	50	19.7	27.2	19.5
Urban Grassland	48	35.3	81	78.5
Urban Woodland	15	22.8	70.8	56.7
Rural Grassland	949	32.4	88.6	79.9
Rural Woodland	696	22	63.9	55


☒ reduce
 ☐ expand
 Close

- 1 Attributes and statistics are here displayed in matrix form.
- 2 Reduce column width to minimum.
- 3 Enlarge column width to display complete column captions.

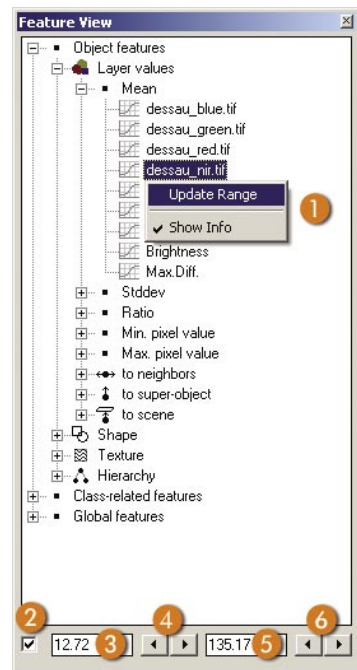
## Feature View

This dialog box is used to determine the display of features in the main view. The main purpose of this dialog is to investigate threshold values for the fuzzy logic membership values.

To open this dialog choose either

- “Tools > Feature View” or “Toolbars & Dialogs > Feature View” from the menu bar, or
- press the  button in the toolbar.

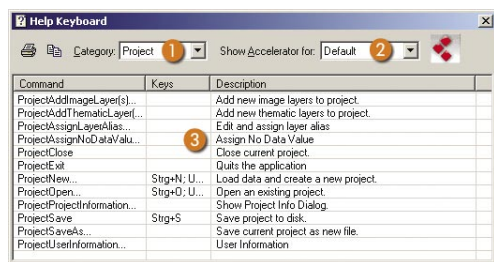
- 1 Choose an image object feature for display with a double-click or use the right mouse button and select „Update Range“ from the popup-menu. While „Update Range“ adapts the active feature range to the selected feature, a double-click only selects the new feature without an update of the range.
- 2 Check to edit the display of the feature range. All image objects whose values are within the range are colored according to the adjusted range in a smooth transition from blue (low values) to green (high values)
- 3 Minimum value of the feature range display.
- 4 Use the arrows to change the minimum feature value for display.
- 5 Maximum value of the feature range display.
- 6 Use the arrows to change the maximum feature value for display.



**Note!** It is not necessary to open the “Feature View” dialog to visualize a feature. In each dialog you use to select features, like “Insert Expression” or “Select Displayed Feature”, you select the feature you want to display with a right click and choose “Update Range”.

## Help Keyboard

This menu gives an overview of all eCognition default accelerator keys and user defined short cuts. To edit the short cuts, use the “Customize” menu in the “Toolbars & Dialogs” menu of the menu bar.



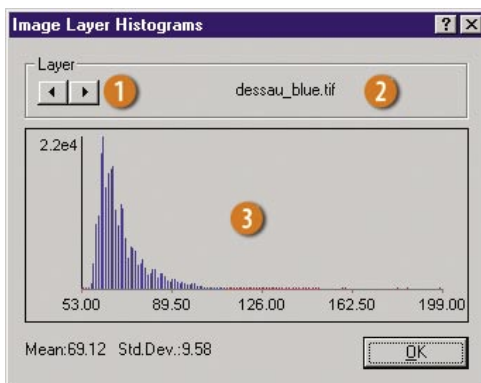
- 1 Command category of the menu bar items.
- 2 Switch between default program accelerator keys and user defined short cuts in eCognition.
- 3 Overview of all defined short cuts.

## Image Layer Histograms

This dialog provides information about the distribution of grey values in single image layers.



To open this dialog choose the item “Tools > Layer Histograms...” from the menu bar of the main application window.

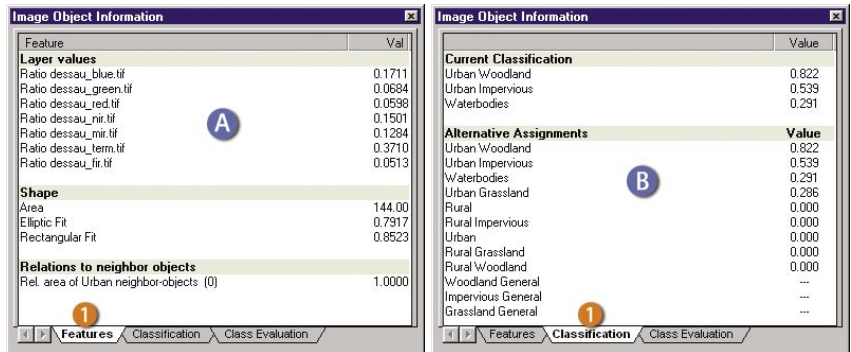
- 1 With the left and right spinners switch to next/previous image layer.
- 2 Name of the current image layer.
- 3 This field provides statistical information about the grey value distribution of the current image layer.



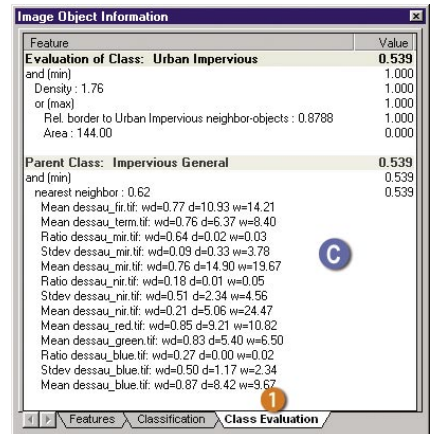
## Image Object Information

This dialog box provides detailed information about the features and classification of an image object.

Both dialog boxes („Image Object Information“ and „Image Object Information 2“) have the same functionality and layout. With these two dialog boxes you can examine features, class description and class evaluation at the same time. To open this dockable dialog box click the buttons  or  in the toolbar or open the item „Toolbar & Dialogs > Image Object Information / Image Object Information 2“ in the menu bar of the main application window.



- A** Shows feature values of a selected image object. To change the display feature list, right-click in the dialog and choose the features to be displayed in the list.
- B** Displays the result of the last classification of the image object. The three best classification results are stored and displayed. Alternative assignments can differ from the current classification due to changes in the membership functions since the last classification run.



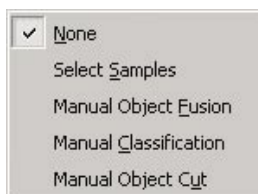
- C** Displays detailed classification evaluation for the selected class. Double-click an item to edit its membership function.
- 1** Click the register tab to open the dialog „Features“, „Classification“, or „Class Evaluation.“

## Input Mode

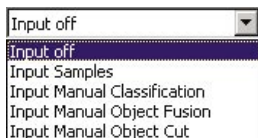
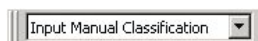
Use this menu item to select samples, manually fuse image objects, perform manual class assignments for image objects, or cut objects by manually. The input mode does not change automatically if the “Sample Editor” or the “Manual Fusion” button are activated.

To open this menu item either

- choose the drop-down menu “Input Mode” from the menu bar of the main application window and choose the input mode.



- choose the combo box “Input Mode” from the tool bar and choose the desired input mode.

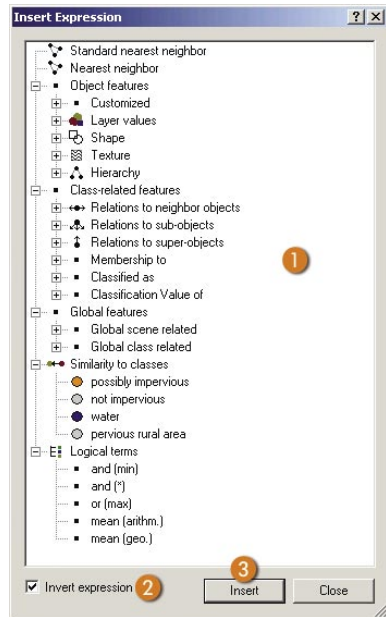


## Insert Expression

This dialog box is used to choose and insert an expression into a class description.

Open it by right-clicking the logical term in the class description and choose “Insert new Expression” from the context menu or by double-clicking on the logical expression.

- 1 Select the desired expression by navigating through the hierarchy of features. To insert the expression, double-click it or mark it and click the “Insert” button.
- 2 When this box is checked the selected expression will be inverted (1 – assignment value) and linked with a “not” expression.
- 3 Insert the selected expression into the class description.



## Layer Properties

Use this dialog box to check and edit the georeferencing information of the imported image layers and thematic layers. You can also assign an alias for the chosen layer in this dialog.

Generally, the georeference data can only be edited at the beginning of a new project. After creating the project, no changes can be made.

To open this dialog box move the mouse into the dialog “Import Raster Layers: Create Project” and right-click or double-click in the window of the imported image or thematic layers.

- 1 The name of the imported image layer or thematic layer in which the geocoding is to be edited.
- 2 X-coordinate of the lower left corner of the image.

- 
- Layer Properties** [?] [X]
- Name:  1
- Geocoding
- Lower left X:  2
- Lower left Y:  3
- Pixel size:  4
- 5 No Geocoding ☐
- Channel Attribute:  6
- OK Cancel

In this dialog box the function slope and the values of membership functions are defined and edited.

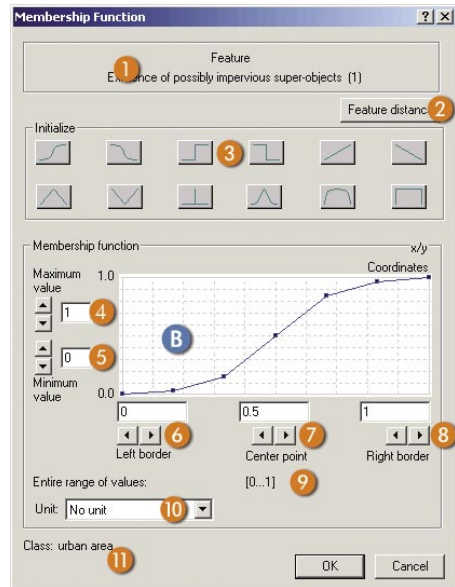
A The dialog box of object features without the button “Feature distance.”

- 1 Display of the feature name.

- 
- Membership function**
- Feature  
Mean dessau\_nir.img
- Initialize
- Maximum value: 1.0  
Minimum value: 0.0
- Left border: 50  
Center point: 50.97887324 / 0.25  
Right border: 51
- Entire range of values: [0..255]
- Unit: No unit
- Class: water
- OK Cancel



- 3 Initialise the shape of the membership function here.
- 4 Edit the maximum value of the resulting membership function.
- 5 Edit the minimum value of the resulting membership function.
- 6 Edit the left border of the transition zone of the membership function. The membership value for the left border will be assigned to all feature values less than the left border.
- 7 Edit the centre point of the transition zone of the membership function.
- 8 Edit the right border of the transition zone of the membership function. The membership value for the right border will be assigned to all feature values larger than the right border.
- 9 This field shows the entire value range of the recent feature.
- 10 Text f
- 11 The name of the class whose class rules are being edited.



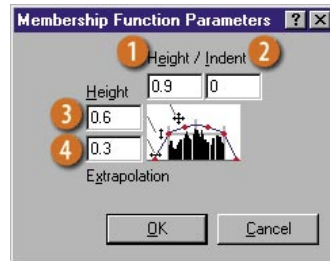
## Membership Function Parameters

Use this dialog box to edit the parameters for membership functions computed from sample objects.

To open this dialog box either:


- choose the item “Sample Editor > Generate Membership Functions > Parameters...” from the menu “Sample” of the menu bar.
- open the dockable dialog “Sample Editor,” right-clicking and choosing the item “Generate Membership Functions > Parameters.”

- 1 Edit absolute height of membership function here.
- 2 Edit indent of membership function here.
- 3 Edit height of linear part of membership function here.
- 4 Edit extrapolation width of membership function here.

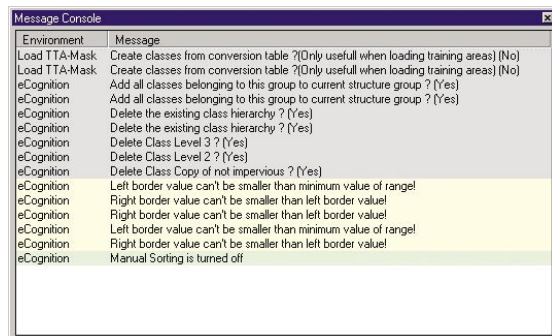


## Message Console

This dialog provides information about warnings or other messages from eCognition concerning the workflow in your project. The message console will automatically come up whenever an operation cannot be executed. Warning messages are shown in yellow; messages that need user interactions are marked in grey. Hint messages are shown in green and error messages that immediately stop the execution of eCognition are shown in red.

To open this dialog press the  button from the tool bar of the main application window.

In the left column is shown the eCognition environment which prompted the message.



The right column shows the output message and - if present - the user's interaction with the dialog.

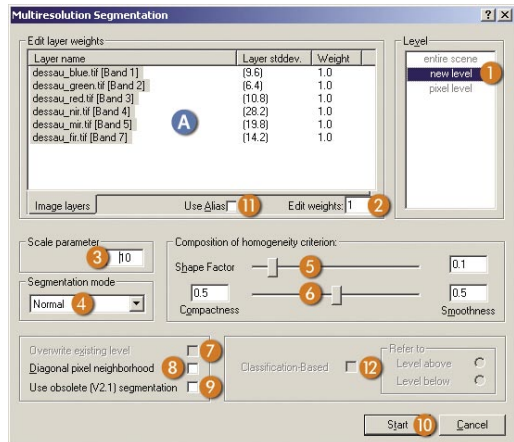
You can clear the console by right clicking on it and selecting “Clean Console”.

## Multiresolution Segmentation

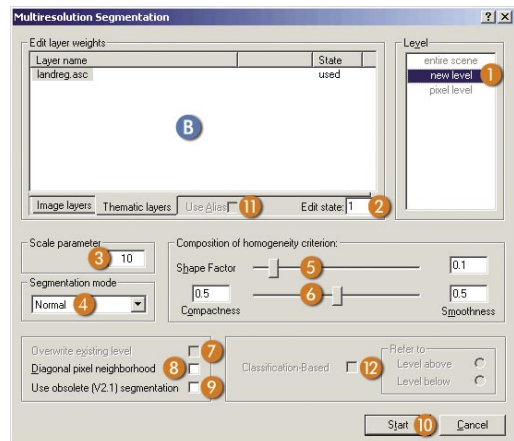
This dialog box is used to edit the parameters for eCognition's multiresolution segmentation.

To open this dialog box select the item “Image Objects > Multiresolution Segmentation...” from the menu bar of the main application window or click  in the tool bar.

- A** Displays image layers in the scene and their weights for segmentation. The weights are internally standardized to 1.
- B** Displays thematic layers in the scene and whether they are used or not in the segmentation process. This register tab is only available if a thematic layer has been loaded into the project.



- 1 Select the level in the image object hierarchy on which the segmentation will be performed.
- 2 Use this field to define the layer weight for the selected image layers. Choose 0 to deactivate the use of a thematic layer during segmentation.
- 3 Use the scale parameter to define the resolution



of the image object level. Increase the value to create larger image objects. The scale parameter refers exclusively to the weights of the image layers (weight \* scale parameter).

- 4 Select the desired segmentation algorithm (in most cases "Normal").
- 5 Weight color homogeneity against form homogeneity here. The sum of the weights is 1. Edit the value for color or shape. The counterpart is fitted automatically.
- 6 Use smoothness and compactness to define how shape homogeneity is described. Increase smoothness to achieve smoother edges of image objects, increase compactness to create image objects of a more compact form.
- 7 If you activate this box, an existing selected image object level will be removed and replaced by the new one.
- 8 Enable diagonal 8-pixel neighborhood instead of default 4-pixel neighborhood.
- 9 Use obsolete segmentation method of eCognition version 2.1 and lower to create image objects.
- 10 Start the segmentation with current settings.
- 11 Select if the multiresolution segmentation shall refer to layer aliases.
- 12 Perform multiresolution segmentation only on objects within structure groups of the level above or below.

### Object Table (not available in the LDH version)

In this table information about the objects' properties is given. The table is also linked to the image view and Image Object Information dialog. You can sort the objects by the displayed items. The Items ID, Class and Membership are always shown.

ID	Class	Membership	Mean dessau bl...	Ratio dessau bl...	Min. pixel value ...	Max. p
26	Waterbodies	0.83519	61.444	0.21167	60	63
25	Waterbodies	0.8471	61.571	0.20771	59	64
24	Waterbodies	0.82671	60.32	0.2095	57	66
23	Waterbodies	0.93091	62.261	0.2171	59	65
22	Waterbodies	0.93923	57.004	0.22659	53	64
21	Waterbodies	0.97927	61.538	0.19162	59	66
20	Waterbodies	0.90978	61.5	0.22083	60	64
19	Waterbodies	0.97393	60.508	0.2303	54	79
18	Waterbodies	0.53764	66.9	0.21908	61	79
17	Waterbodies	0.89903	65.667	0.20844	60	78
16	Waterbodies	0.71641	61.429	0.21256	59	64
15	Waterbodies	0.99703	59.295	0.23234	53	97
14	Waterbodies	1	64.867	0.22286	62	68
13	Waterbodies	0.95136	64.868	0.2136	61	70
12	Waterbodies	0.83639	65.143	0.2152	63	69
11	Waterbodies	0.91699	65.000	0.20900	60	60

## Options

In the Options dialog you can adjust several general settings for your eCognition sessions.

To open this dialog, select the item “Tools > Options” from the menu bar.

- 1 Use the separator symbol of your regional windows settings. If switched . is used.

- 2 Define the color for skeleton display.

- 3 Automatically link new image windows to the currently opened.

- 4 If switched on, the midpoint of an selected object is displayed.

- 5 If switched on, the outlines of the sub-objects of an selected object are displayed.

- 6 If switched on, all warnings are given as single message boxes.

- 7 If switched on, the current view settings will be set for all new view windows.

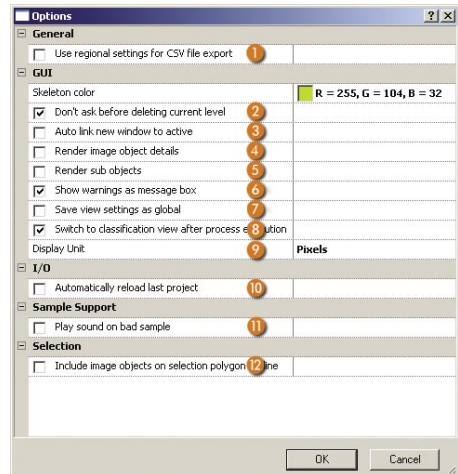
- 8 If switched on, after each classification the view mode “Classification” will be turned on.

- 9 Shows the current unit used for displaying.


- 10 If switched on, after restarting eCognition the recent project will be loaded automatically.

- 11 If switched on, a sound will be played if a critical sample is taken and the Sample Selection Support dialog is active.

- 12 If switched on, also image objects will be selected which are touched by the selection polygon or rectangle.



## Pan Window

The “Pan Window” is used to navigate in large data sets to allow smooth data handling. To open this dockable dialog box, click the item “View > Open Pan Window” from the menu bar of the main application window or click the  button from the toolbar.

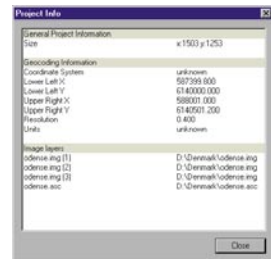
- 1 This field shows the area which has been chosen for panning. Drag the frame with the right mouse button to navigate in the image.



## Project Info

This dialog box provides information about the project data and size, and the geocoding header information.


To open this dialog, choose “Project > Project Info...” from the menu bar of the main application window.

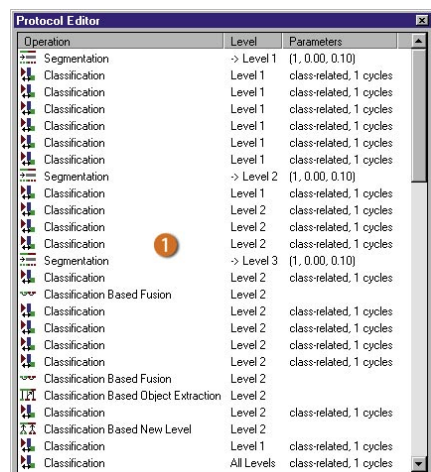


## Protocol Editor

Use this dialog box to execute, create or edit many working steps in the form of a protocol.

To open this dialog box either

- choose the item “Protocol > Open Protocol Editor...” from the menu bar of the main application window or click  in the tool bar.
- choose the item “Toolbars & Dialogs > Protocol Editor” from the menu bar of the main application window.



- 1 This list displays the steps saved in the protocol. Right-click to open the context menu.

Items of the “context menu” (right-click):




**Start Recording** Record new protocol entries.

**Load Protocol** Load a protocol file with defined working steps into the project.

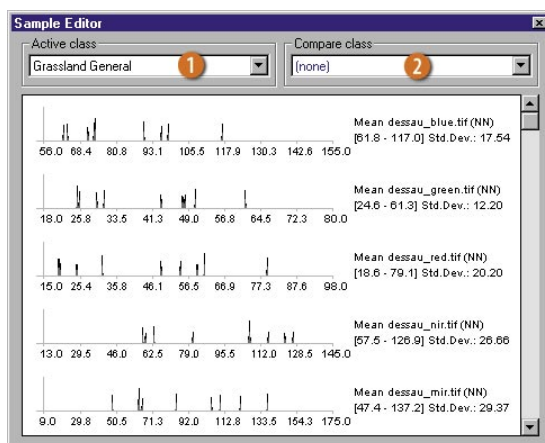
**Stop recording** Stop the recording process.

## Sample Editor

Use the “Sample Editor” to compare image object and sample histograms, to view the range of the image and sample histograms and to compare with other classes.

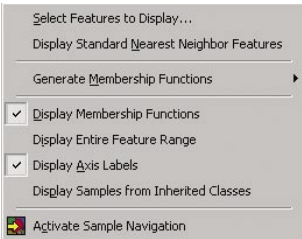
To open this dockable dialog box choose the item “Samples > Open Sample Editor...” from the menu bar or click  in the tool bar and change the input mode in the menu bar and combo box.

- 1 Select the class for which you want to edit or collect sample objects.

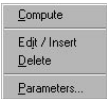


- 2
- Select a class for comparison with the active class.

Menu items in the context menu “Sample Editor”:

<b>Select Features for Display...</b>	Select features for display in the “Sample Editor” dialog. All image object features can be used (form and texture also).	
<b>Generate Membership Functions</b>	See below.	
<b>Display Membership Functions</b>	Switch between display of automatically computed membership functions and no display.	
<b>Display entire Feature Range</b>	Switch display between whole feature range and sample feature range.	
<b>Display Axis Labels</b>	Switch on or off axis labels.	
<b>Display Samples from Inherited Classes</b>	Shows also samples from sub-classes if there are any.	
<b>Activate Sample Navigation</b>	Activates the sample navigation mode. By clicking on a slot you become navigated automatically to the first sample of the slot.	
<b>Display Samples from Inherited Classes</b>	Shows also samples from sub-classes if there are any.	
<b>Activate Sample Navigation</b>	Activates the sample navigation mode. By clicking on a slot you become navigated automatically to the first sample of the slot.	


Menu items in context menu “Generate Membership Function”:

<b>Compute</b>	Automatic calculation of membership function from a selected sample feature.	
<b>Edit / Insert</b>	Edit or insert a membership function manually.	
<b>Delete</b>	Delete the membership function of the selected feature.	

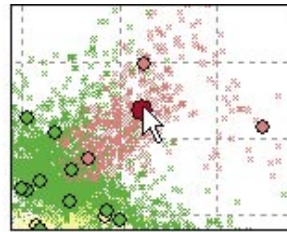
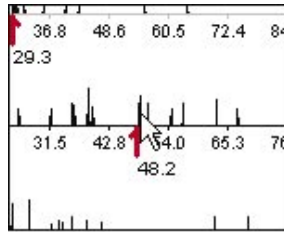


## Sample Navigation

Use „Sample Navigation“ to navigate to samples selected in the „Sample Editor“ or the „2D Feature Space Plot“ with a single click on the corresponding sample.

To activate the sample navigation click the  button. Once a sample is selected the view navigates to the corresponding sample that will be highlighted in the image view.

If in the „Sample Editor“ or in the „2D Feature Space Plot“ there are two or more samples so close together that it is not possible to select them separately in these dialogs, it is possible to switch between the samples with the blue arrows or the pull-down menu.



## Sample Selection Information

If samples were already selected for the corresponding classes you can assess the quality of a new sample with the „Sample Selection Information“ dialog. This way it is easier to decide if an object contains new information for the description of a class, or if it rather belongs to another class. For more information see [Functional Guide > Classification Advanced](#)

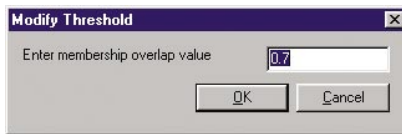
Class	Membership	Minimum Dist.	Mean Dist.	Critical Samples	Number of Samples
rural	0.920	0.259	9.871		19
impervious surface	0.710	1.063	70.592	1	8
agriculture	0.531	1.965	10.078	0	4
water	0.019	12.355	16.431	0	4

- 1 „Class“ describes the name of the class to which the values in the row belong to.
- 2 „Membership“ shows the membership value of the nearest neighbor classifier for the selected object.

- 3 „Minimum Dist.“ shows the distance in feature space to the closest sample of the appropriate class.
- 4 „Mean Dist.“ shows the average distance to all samples of the concerned class.
- 5 „Critical Samples“ shows the number of samples that are in a critical distance to the selected class in the feature space.
- 6 „Number of samples“ shows the number of samples selected for the corresponding class.

In the first line all values for the selected are class colored in yellow. All classes to which the selected sample is situated in a critical distance are red. All other classes that are not in a critical relation are colored green.


It is possible to change the critical sample membership value with a right click in the window. Select “Modify critical sample membership overlap” from the pop-up menu. The default value is 0.7, which means all membership values higher than 0.7 are critical.

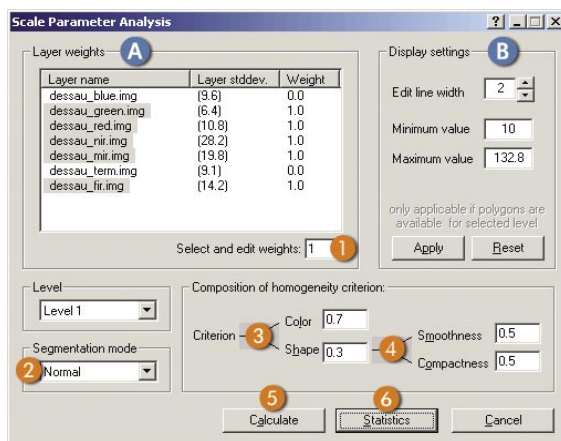


It is also possible to select the classes you want to display. In this case right click in the dialog and choose “Select classes to display” from the pop-up menu.

## Scale Parameter Analysis

With this dialog you control the scale parameter analysis.

To open this dialog box select the item “Image Objects > Scale Parameter Analysis ...” from the menu bar of the main application window or click  in the tool bar.

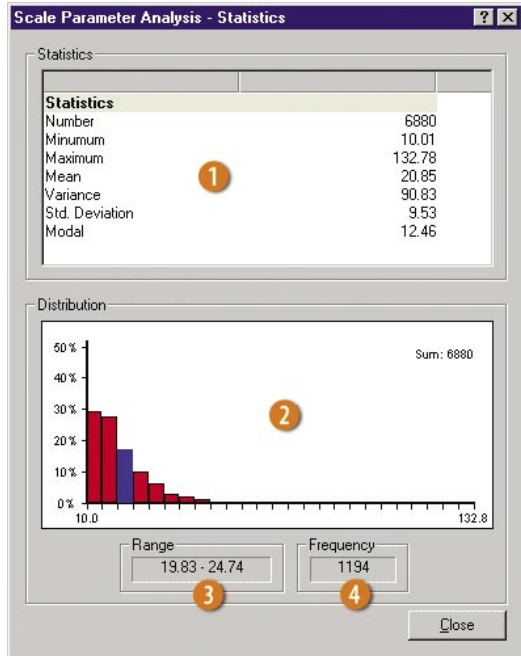


- A** Displays image layers in the scene and their weights for segmentation. The weights are internally standardized to 1.
- B** Set displays settings for visualization of merging scale parameters. The settings are only applicable, if there are polygons present.
- C** Select the level in the image object hierarchy on which the analysis is to be performed.
- 1** Use this field to define the layer weight for the selected image layers. Choose 0 to deactivate the use of a layer during analysis.
- 2** Select the desired segmentation algorithm (in most cases “Normal”).
- 3** Weight color homogeneity against form homogeneity here. The sum of the weights is 1. Edit the value for color or shape. The counterpart is fitted automatically.
- 4** Use smoothness and compactness to define how shape homogeneity is described. Increase smoothness to achieve smoother edges of image objects, increase compactness to create image objects of a more compact form.
- 5** Calculates the scale parameters with current settings.
- 6** Shows statistics of calculated scale parameters.
- 7** Exports the statistics.

### “Scale Parameter Analysis - Statistics”

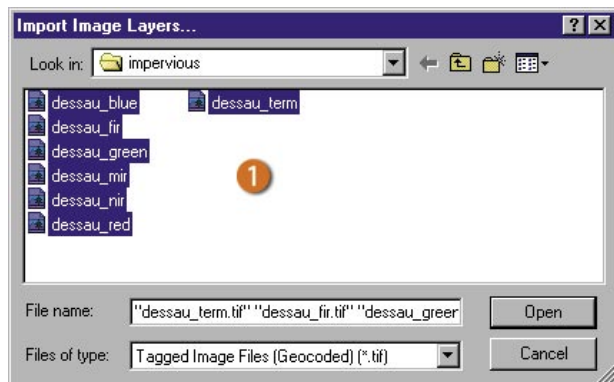
This dialog gives information about the statistics of the potential scale parameters for all object pairs.

- 1 Shows the statistical parameters: Number of scale parameters, minimum value, maximum value, mean value, variance of values, the standard deviation and the modal of the most frequently occurring value range.
- 2 Shows the distribution of potential scale parameter values. Click on a slot of the histogram to see the slots value range and the frequency of this value range.
- 3 Shows the range of the currently selected slot.
- 4 Shows the frequency of entries in the currently selected slot.



## Select Layer

This dialog box pops up every time you have to select image layers, e.g., when you want to create a new project, or load an additional image or thematic layer into the project.



- 1 To select a new layer mark the file name and click “Open.” It is also possible to select multiple files at once.

## Select Level

This dialog box pops up every time you have to make a selection of image object levels, e.g., when you want to delete an image object level or create a training and test area mask from an image object level or create polygons starting from a base level.

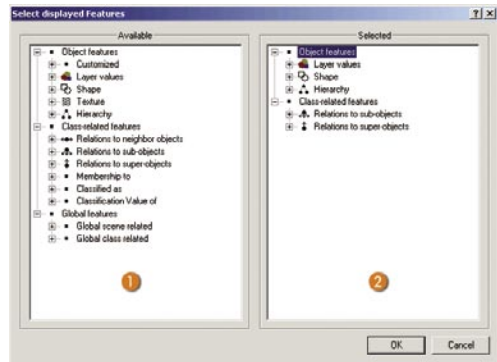
- 1 To select a level mark it and click “OK.”



## Select Displayed Features

Use this dialog box to make a selection of one or more features for display of feature values in the image object information dialog. Open the dialog by right-clicking in the dialog “Image Object Information”, register tab “Features.”

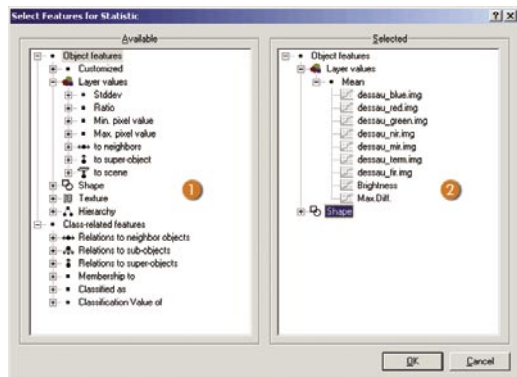
- 1 This is the list of all selectable features. Double-click a feature to select it.
- 2 This list displays the selected features. Double-click a feature to deselect it.



### Select Features for Statistic

Use this dialog box to make a selection of one or more features for the purpose of display of feature values, statistical analysis or export.

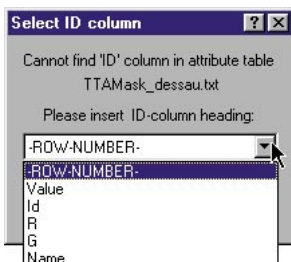
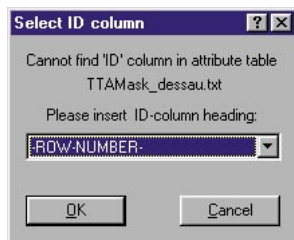
- 1 This is the list of all selectable features. Double-click a feature to select it.
- 2 This list displays the selected features. Double-click a feature to deselect it.



### Select ID Column

If you import a thematic layer into your project and eCognition does not find an appropriate column with the caption ID in the respective attribute table, use this dialog box to determine the column containing the polygon IDs.

Select the caption of the column containing the polygon ID from this drop-down menu.

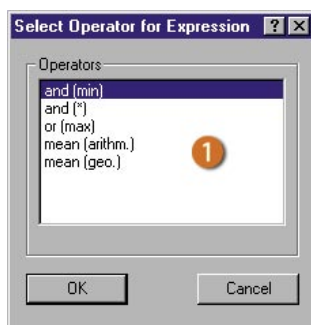


### Select Operator for Expression

Use this dialog box to edit a logical operator in the “Class Description.”

Right-click on a logical expression in the class description and choose the item “Edit Expression” in the context menu.

- 1 List of all available logical operators. To select an operator, mark it and click “OK.”

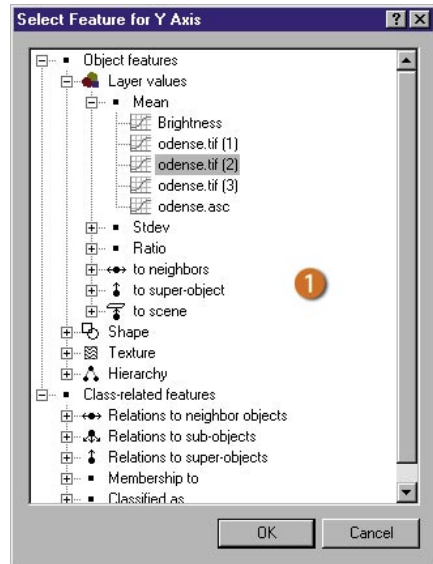


## Select Single Feature

Use this dialog to select a single feature from the feature list.

This dialog is opened when you have to select a single feature, e.g., “2D Feature Space Plot” or if a new expression should be included in the class description.

- 1 To select a feature from this feature list, double-click it or mark it and click “OK.”



## Set Nearest Neighbor Function Slope

The basic effect of the function slope is to increase or decrease the distance an object may have from the nearest sample in feature space while still being classified.



- 1 Degree of membership function at the distance of 1 from image object to sample object. Larger values result in more classified objects.

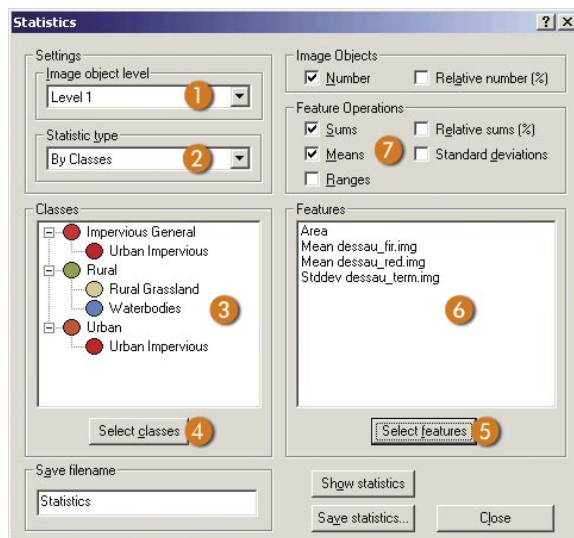


## Statistics

Use this dialog box to calculate statistics of classes using the image object features.

To open the dialog box choose the item “Tools > Statistics...” from the menu bar of the main application window.

- 1 Select the image object level for which the statistics have to be calculated.
- 2 Select the statistics type.
- 3 Check these buttons to select the operations that will be performed on the feature values.
- 4 This field lists the classes for which feature statistics will be calculated.



- 5 Click here to edit the class list above.
- 6 This field lists the features used for the calculation of statistics.
- 7 Click here to edit the feature list above.
- 8 In this field write the name of your statistics export file.
- 9 Click this button to display the statistics.
- 10 Click this button to save the statistics to an ASCII file.

## Subset Selection

This dialog enables the graphical and numerical selection of image subsets. Only during project creation can this subset definition be made. To invoke this dialog chose “Subset selection” in the “Import Raster Layer: Create Project” dialog.

- 1 Display the image and graphical subset selection by clicking and holding the left mouse button at the upper left corner of the subset and dragging to the lower right corner of the subset.

- 2 Minimum X value of subset.

- 3 Maximum X value of subset.

- 4 Minimum Y value of subset.

- 5 Maximum Y value of subset.

- 6 Resolution of active image layer.

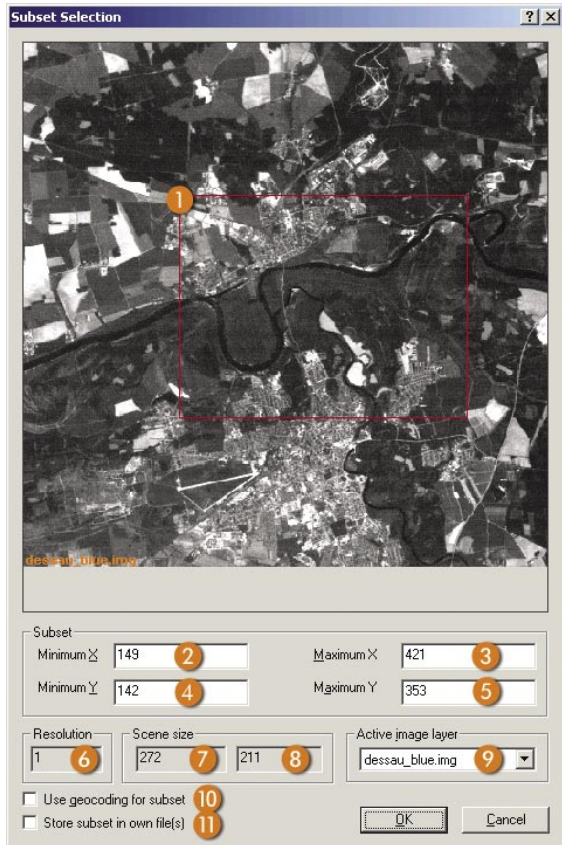
- 7 Scene size of subset in X direction.

- 8 Scene size of subset in Y direction.

- 9 Name of the active image layer.

- 10 Check to use geocoded coordinates.

- 11 Check to store the subset in an own file



## System Info

This dialog box provides information about the hardware and software status of the computer system.

To open this dialog choose “? > System Info...” from the menu bar.

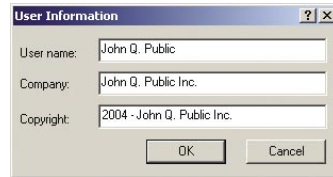
- ① This field provides information about the hardware and software of your system.
- ② Edit the folder for storing temporary files. This can only be done before loading any data.



## User Information

This dialog box provides information about the current user, company and copyright.


To open this dialog choose “Project > User Information” from the menu bar

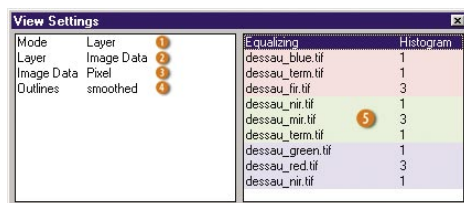


## View Settings

This dialog box is eCognition’s central visualization tool.

To open this dockable dialog box either

- click the button  from the tool bar.
  - choose the item “Toolbar & Dialogs > View Settings” in the menu bar of the main application window.
- ① Select the view mode with right-click on “Mode”. Choose among: “Layer, Samples, Classification, Classification Stability, Best Classification Result.”



- 2 Select layer with right-click on “Layer.” Choose among “Image Data, TTA Mask and Thematic Layer.”
- 3 Select image data. Switch between “Object mean and Pixel.”
- 4 Before displaying outlines you have to create polygons first. Select the outline display. Choose between outlines (raster/smoothed) or no outlines.
- 5 Color display-equalizing mode (Double-click to change values in the “Edit Layer Mixing” dialog).

Layer color mixing (Double-click to change values in the “Edit Layer Mixing” dialog).

Double-click: opens dialog “Edit Highlight Colors”.

Double-click: opens dialog “Edit Layer Mixing”.



# 7 GUIDED TOURS

This section of the user guide serves as an introduction to the handling of the software by means of different examples. You will find different types of data showing the multitude of applications eCognition can be used for. The guided tours cover all the important features of the software. Links to the “Concepts & Methods” chapters will be provided for you along the way. Following them will give you more information on the new terms and techniques you encounter.

## **Tour 1: Landsat TM subset of Orange County, California..... 379**

Keywords: Multiresolution segmentation, sample objects, nearest neighbor classification, training and test areas mask, feature space optimization

## **Tour 2: Analysis of the degree of urban impervious surface ..... 402**

Keywords: Training and test areas mask, classification-based segmentation, multilevel classification, utilization of scale-dependent information, accuracy assessment, export of thematic layers

## **Tour 3: High resolution aerial scan..... 432**

Keywords: Membership function, digital surface model, thematic layer, class-related features, border optimization

## **Tour 4: Radar image of a tropical rain forest in Kalimantan, Indonesia ..... 460**

Keywords: Sub-object line analysis segmentation, line features based on sub-objects

## **Tour 5: Aerial Photo and LIDAR surface model of Odense (Denmark) ..... 470**

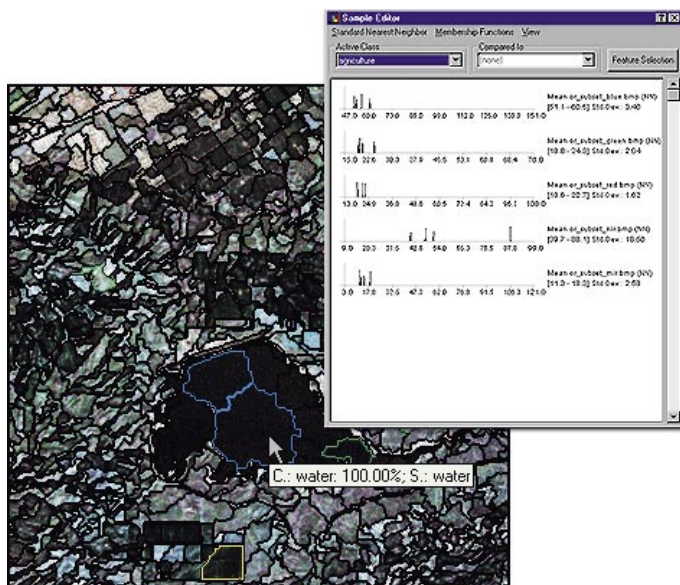
Keywords: Customized features, automation of operations, multiple window functionality, classification-based segmentation

If you have problems working your way through one of the guided tours, remember that cross-reading between the explanatory chapters and the tour you are working through is a helpful way to make yourself more familiar with eCognition's features.

In this exercise you will perform a nearest neighbor classification on a LANDSAT TM database. Nearest neighbor classification makes it easy to achieve classification results quickly without much effort: click and classify!


In this exercise you will learn how to:

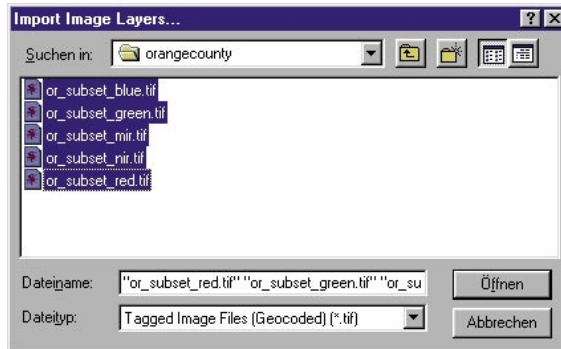
- load and display raster data,
- perform an image segmentation,
- create a simple class hierarchy,
- insert the nearest neighbor classifier into the class descriptions,
- classify,
- perform classification quality assessment.



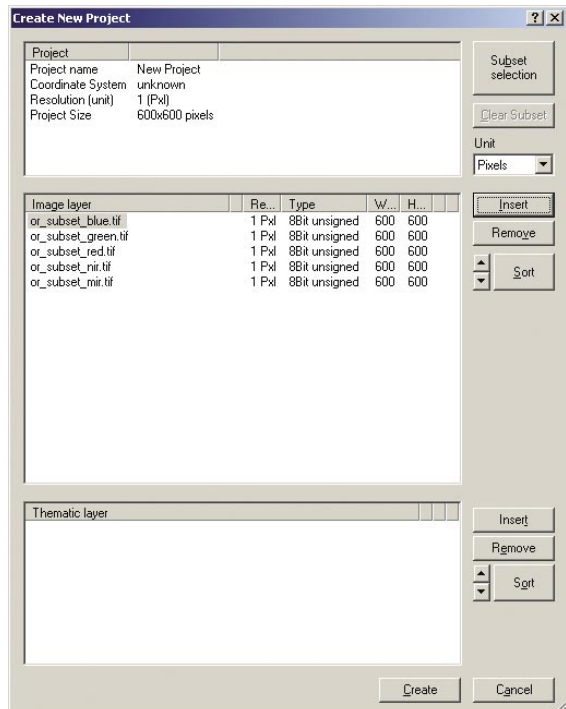
Data courtesy of PCI Geomatics, Canada

## Loading the raster data


1. Start eCognition and choose “Project > New...” or click  in the tool bar.
2. Navigate to the directory “...\data\orangecounty\,” select the five tif-files at once by holding down the SHFT-key, and then click “Open.”

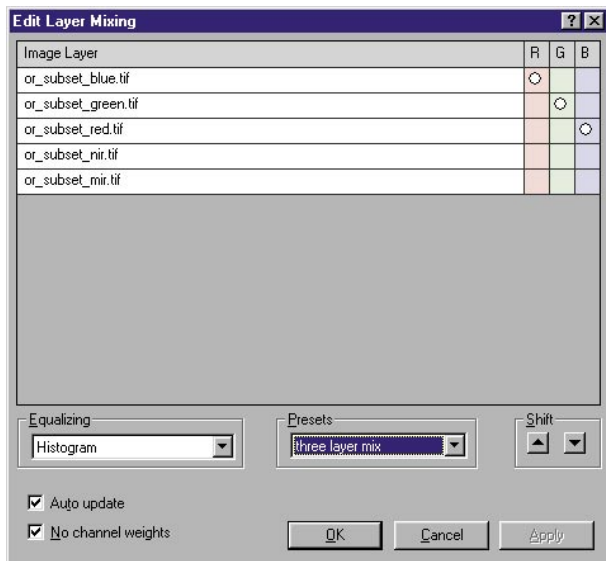


3. Change the order of the tif-files according to their sequence in the spectrum.
4. Click “Create” to import the raster layers into a new project.






5. Open the “Edit Layer Mixing” dialog (in the view menu “View > Layer Mixing...” or click  in the tool bar), apply a histogram stretch and a three layer mix as shown below.



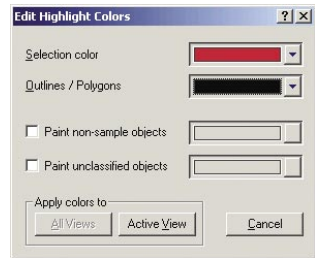
6. Click “OK.” The following view is displayed.




7. Open the “Edit Highlight Colors” dialog in the “View” menu or click  and change the “Selection color” to red and the “Outlines” color to black.

8. Click “OK.”

The raster data is now imported and visualized. Right now only the information provided by the single pixels can be used. To create image objects by grouping pixels, the segmentation process must be started now.



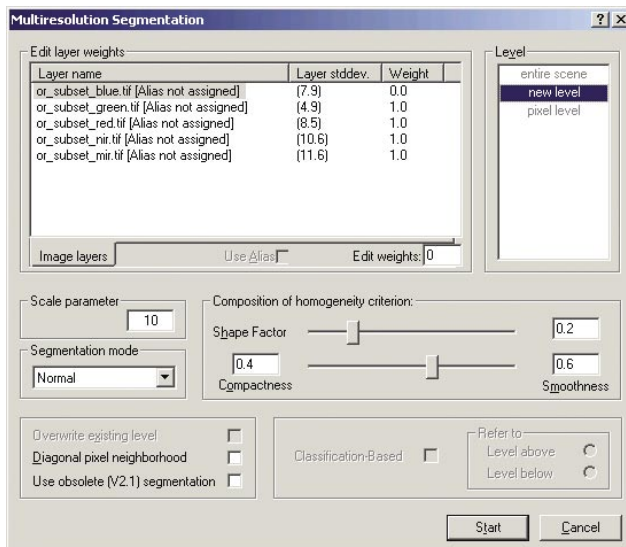
## Creating image objects




1. From the “Segmentation” menu choose “Multiresolution Segmentation...” or click  in the tool bar.

**Note!** Remember one basic rule when segmenting an image: Create image objects as large as possible and at the same time as small as necessary.

2. Edit the segmentation parameters as shown below. Note the layer weights of the bitmap files indicated to the right of them.

3. Click “Start.”




4. Create polygons to view the outlines of the image objects (“Polygons > Create Polygons” or click  and select the level the polygons are to be created on).
5. Adjust the “View Settings” dialog to display the segmented image as “Image Data” showing the “Object mean” values with “smoothed” outlines.
6. To toggle the display of the image data between object mean values and pixels or to show or hide outlines, click  or  respectively.



Now you have created a simple image object hierarchy consisting of one image object level. A great variety of information can be derived from each object for classifying the image.

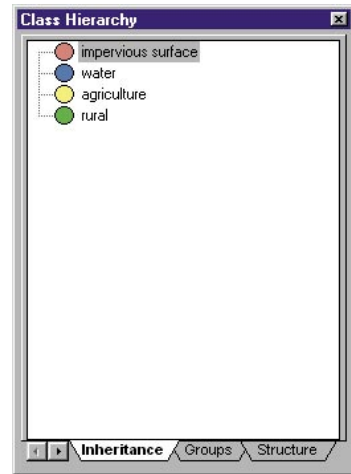
### Creating a knowledge base by means of the class hierarchy

1. Select “Open Class Hierarchy...” from the “Classification” or “Class Hierarchy” from the “Toolbars & Dialogs” menu or click .

Four classes will be distinguished in this exercise: *impervious surface*, *water*, *agriculture* and *rural*. The first thing to do is to define the class names and class colors.

2. Be sure you are in the “Inheritance” register.

3. Create the classes *impervious surface*, *water*, *agriculture* and *rural* as shown here. Select “Classification > Edit Classes > Insert Class” in the menu or right-click in the dialog box and select “Insert Class.”



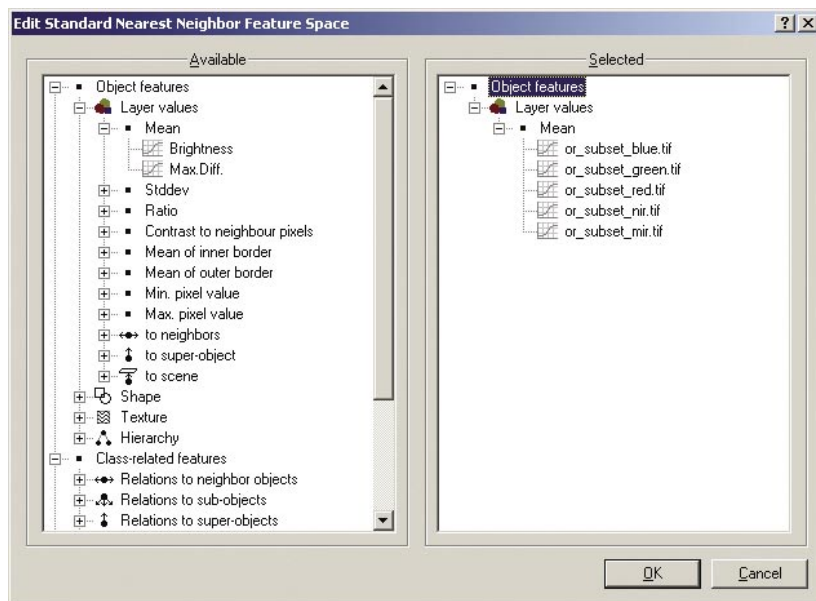
### Inserting the classifier

eCognition offers two different classifiers: nearest neighbor and membership functions.

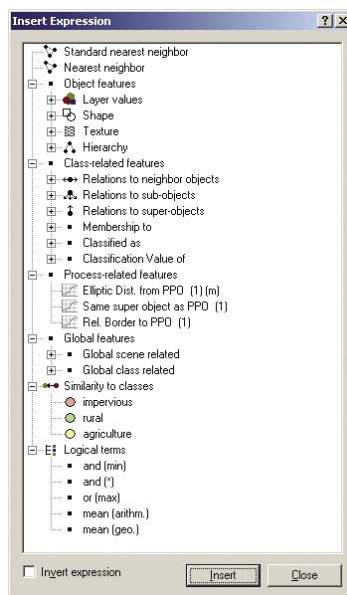
This example is a nearest neighbor classification. Before you insert the standard nearest neighbor expression into the class descriptions, define the feature space in which the distance between image objects is calculated. In contrast to the normal nearest neighbor, the standard nearest neighbor exists only in one definition in a project and therefore uses an identical feature space in all class descriptions in which it is contained. If you change the feature space of the standard nearest neighbor at any place, it will be changed in all the other places where it is used, too.

1. Select the menu item “Classification > Nearest Neighbor > Edit Standard NN Feature Space...” in the menu bar.
2. You will notice that in the right window the mean values of the five layers have already been selected by default. Since this feature space will be adopted, just click “OK.”

**Note!** Always try to describe a class with as few features as possible. The use of too many features in one class description causes an immense increase of overlaps in the feature space, complicates classification and reduces transparency significantly. However, if you have to use a larger number of features, the use of nearest neighbor as classifier is advisable. The correlations in a multidimensional feature space can be handled much better by nearest neighbor than by membership functions.

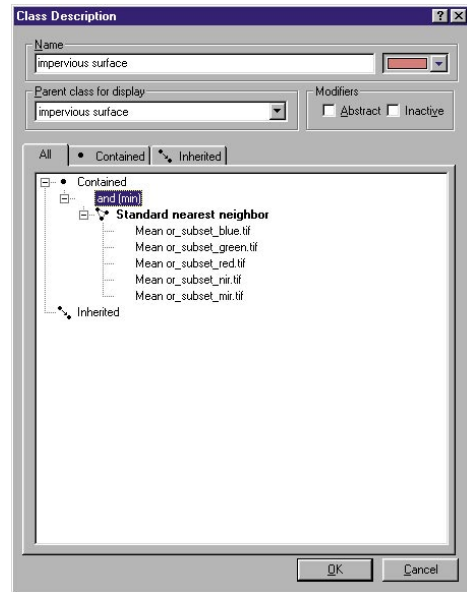


3. Open the “Class Description” dialog of the class *impervious surface* by double-clicking it.
4. Double-click the logical expression “and (min).”
5. Navigate to the expression “Standard nearest neighbor” and insert it into the class description.
6. Close the “Insert Expression” dialog if necessary.



7. Click “OK” to close the “Class Description” dialog.
8. Repeat the last step to insert the standard nearest neighbor into the remaining class descriptions.

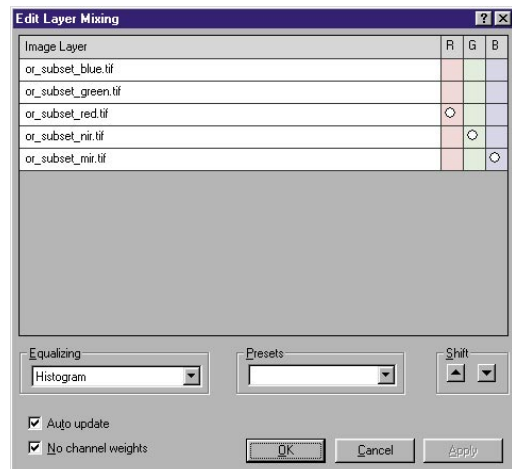
As an alternative, the standard nearest neighbor can also be inserted by choosing the menu item “Classification > Nearest Neighbor > Apply Standard NN to Classes....”.



## Declaring sample objects

Nearest neighbor classification in eCognition is similar to supervised classifications in common image analysis software. You have to declare training areas, which are typical representatives of a class. In eCognition such training areas are referred to as samples or sample objects.


1. To make identification of image objects easier, change the view from “Object mean” to “Pixel” and activate the “Outlines” in the “View Settings” or with the corresponding buttons.
2. Change the layer mixing as shown here.







In this view, most of the objects can easily be identified. Objects representing agricultural areas appear bright green, impervious objects appear purple and violet and are characterized by a high degree of texture. Water objects appear dark, but do not mix them up with shadow objects caused by the mountainous terrain in the lower left part of the image. All of the remaining objects represent rural areas. Rural areas consist of land coverage with very diverse spectral information. Consequently, *rural* is a very heterogeneous class. Nevertheless, eCognition's nearest neighbor classification can handle such heterogeneities.

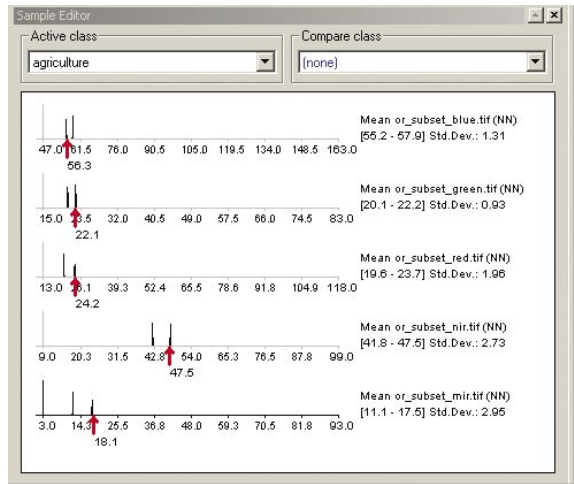
3. From the "Samples" menu select "Open Sample Editor..." or click  in the tool bar.
4. Choose "Select Samples" from the "Input Mode" menu. If you do this step first, the "Sample Editor" opens automatically.

The sample editor displays five diagrams for feature values. The features are the mean values for each channel. Thus, the feature space displayed in the sample editor is identical to the standard nearest neighbor feature space. The "Active class" selection box in the upper left corner of the sample editor lets you choose the class you want to enter samples for. The sample feature values can be compared to those of another class by choosing a different class in the "Compared to" selection box.


5. Change the “Active class” to *agriculture* by selecting the class in the respective selection box or by clicking on the class in the “Class Hierarchy” editor.
6. Click a sample object for the class *agriculture*.


With a single-click, the sample editor marks the object's values with red arrows for each feature. Simultaneously, the content of the „Sample Selection Information“ changes.

7. Mark the object as a sample object by double-clicking or SHFT-clicking on it.



The feature values of the sample object are now displayed by additional marks in the histograms.

When collecting samples, start in the first step with only one or a few samples for each class, covering the typical range of the class in the feature space, especially when it is heterogeneous. Otherwise, its heterogeneous character will not be fully considered. If a chosen sample is in feature space critically close to samples of other classes it is marked red in the “Sample Selection Information”. To locate critical samples in the image switch on the “Sample Navigation” by clicking . You become automatically navigated to the appropriate sample in the image view as well as in the “Sample Editor”. Use the blue arrows of the “Sample Navigation” bar to switch between a critical or a closest sample and the currently selected object. In the “Sample Editor” also the samples of the critical class are shown as the comparison class. To reselect the currently selected object, click on the gray marked sample in the “Sample Selection Information” and choose “Current”.

Nearest	↑	↓	
Nearest			
Current			

Class	Membership	Minimum Dist.	Mean Dist.	Critical Samples	Number of Samples
agriculture	0.434	2.591	4.491		2
rural	1.000	0.000	3.652	8	11
impervious	0.335	3.402	4.303	0	3
water	0.016	12.909	13.068	0	2

When training a nearest neighbor classification manually, start in the first step with only one or a few samples for each class, covering the typical range of the class in the



feature space, especially when it is heterogeneous. Otherwise, its heterogeneous character will not be fully considered. In this chapter you will learn how to improve the classification in further steps.



Note that the distribution of a class does not need to be continuous when using nearest a neighbor classifier! This makes it, for instance, possible to summarize all different heterogeneous appearances of agriculture in one class.

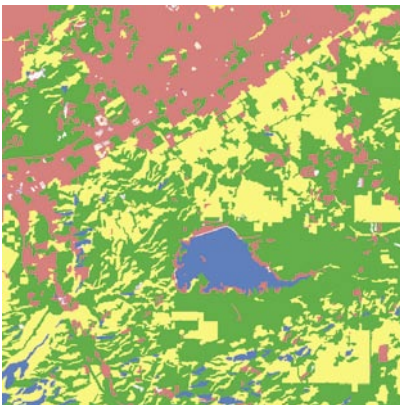
9. Repeat the declaration of samples for the remaining classes *water*, *impervious surface* and *rural*. Insert two or three sample objects for each class. Do not forget to select the appropriate class as active class.

Training areas for the nearest neighbor classification are determined by assigning sample objects to them. Now the first classification will be performed.

### Classifying the image objects in the scene

In the previous step you have defined the feature space and an initial set of samples for the nearest neighbor classification. You have thereby completed the knowledge base. The following step is to classify the image objects in the scene.

1. You could now edit the parameters for the classification process. As there currently exists only one image object level and there are no class-related features used in the class descriptions yet, you can classify without class-related features.
2. Click  in the tool bar to start the classification process.
3. Change the view to display the classification result (click ) and deactivate the outlines.



Since the outcome of the classification depends on the chosen samples, your modified image might look slightly different from the one shown above.

The result so far looks promising, but there is certainly a need for further improvement. Notice that on the one hand there are objects that are not assigned to any of the classes, and on the other hand there is a relatively large number of objects that have been classified incorrectly, especially the objects classified as water in the lower left part of the image. In addition, too many objects are classified as *impervious surface*.

These inaccuracies will now be corrected in iterative steps by assigning typically wrongly classified image objects as samples to the right class.

4. Change to the class you want to work with.

Make sure that “Select Samples” in the “Input Mode” menu is still active.

5. Change the view back to the samples, pixels and outlines. To facilitate your work it could be helpful to open a second window that shows the classification result and is linked with the first window (“Window > New Window” and “Window > Link all Windows”).

6. Assign one or two unclassified image object samples to the class they belong to by double-clicking them. Do this for every class it is necessary for.

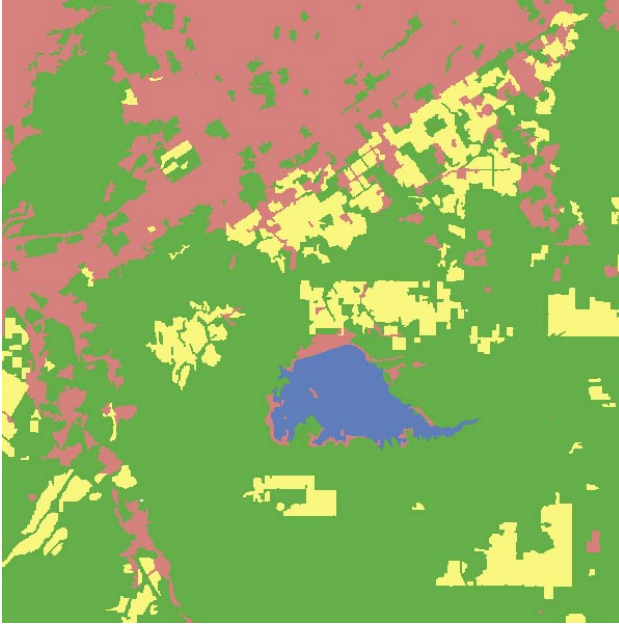
7. Assign one or two incorrectly classified image objects samples of the correct class. Again, do this for each class it is necessary for.

8. Reclassify and display the classification result without outlines.



The new classification result looks better, but there is still room for improvement. Thus, continue to edit sample objects.

9. Repeat inserting unclassified objects as samples and declaring incorrectly classified segments objects of the correct class. Classify again and check the result.
10. Repeat the cycle of sample assignment and classification until you achieve a satisfying classification result.



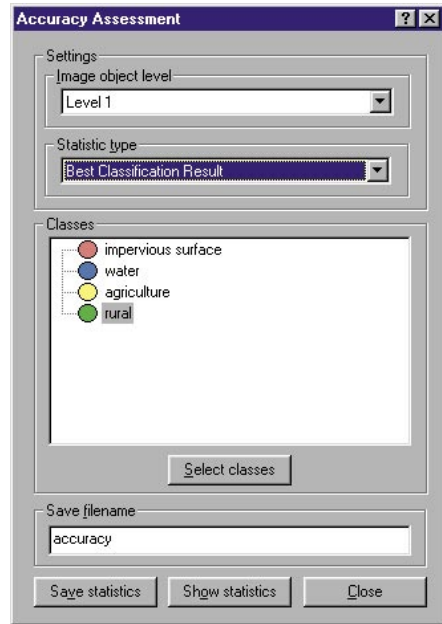
In fact, this procedure of iterative improvement of the nearest neighbor classification leads to a differentiation of the borders of the class distributions in the multidimensional feature space. Starting with a few samples and adding only necessary samples in subsequent steps is a very efficient way to come up with a successful classification. This is supported by the pleasing characteristics of the nearest neighbor classifier of not relying on a continuous, Gaussian distribution and of being able to detect even complex shaped distributions in the feature space exactly.

In turn, let us assess the classification accuracy with objective methods for testing.

## Checking for each image object the best class evaluation result (best classification result)

1. Select “Accuracy Assessment...” from the “Tools” menu.
2. Select “Best Classification Result” as statistic type.

In eCognition, classified image objects are not only assigned to one class or not; you also get a detailed list with the membership values of each of the classes contained in the class hierarchy. An image object is assigned to the class with the highest membership value, as long as this highest membership value equals at least the minimum membership value, which you can edit under “Classification > Advanced Settings > Minimum Membership Value...”



It is significant for the quality of a classification result that the highest membership value of an image object is absolutely high, indicating that the image object attributes are well suited to at least one of the class descriptions.

Using nearest neighbor classification, a high membership value indicates a close distance to one of the given samples. For each image object, the best classification result could be increased for nearest neighbor classification by increasing the nearest neighbor function slope. However, the following section describes how this would decrease classification stability.

3. Click “Show statistics.”

Class	Objects	Mean	StdDev	Minimum	Maximum
impervious surface	608	0.801	0.172	0.103	1
water	30	0.98	0.0327	0.831	1
agriculture	273	0.898	0.105	0.252	1
rural	2137	0.872	0.118	0.156	1

The statistics are displayed as a matrix. To see the graphical presentation you have to open the “View Settings” dialog and click on “Mode” to change the view mode. The graphical presentation indicates the assignment value of each object on a color palette ranging from red (low value) to green (high value). Your results may differ from the ones above, since you most likely selected different sample objects.



4. Move the mouse over an image object to get information about the best classification result.

As you can see, the best classification value for most of the objects is significantly high. There is only a small number of objects with a low assignment value, the minimum in this example is 0.103. The information can be used to check these particular image objects. The class mean values and standard deviations show that only a very small number of objects were classified with such low membership values. All in all, the class assignments are significant.

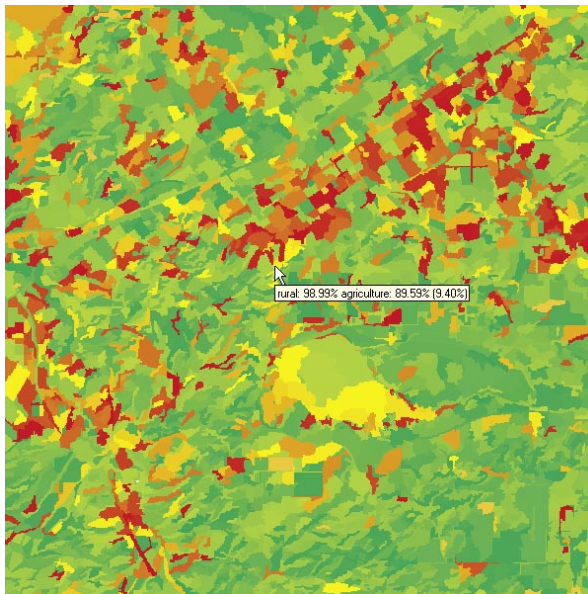
5. If you want to export these statistics as an ASCII file, click “Save statistics.”

## Checking the classification stability

Membership values are not membership probabilities, i.e., they do not add up to 100 %. For this reason, a high membership value to a certain class does not necessarily indicate definite membership in this class. If there is only a low difference between the best and the second best membership value, the classification result is relatively unclear. This difference can be checked for all image objects of one level under “Classification Stability” in the dialog box “Accuracy Assessment.”

1. Open the dialog “Accuracy Assessment.”
2. Select “Classification Stability” as statistic type.
3. Click “Show statistics.”

Class	Objects	Mean	StdDev	Minimum	Maximum
impervious surface	608	0.229	0.209	0.000207	1
water	30	0.0709	0.0335	0.0176	0.147
agriculture	273	0.241	0.192	0.0014	1
rural	2137	0.242	0.16	0.000508	1



For each class, the classification stability is statistically computed over all image objects in the scene.

4. Move the mouse over image objects to achieve information about their exact classification stability.

Most image objects have a clear difference between the best and the second best class evaluation. They are therefore assigned distinctly.

When using nearest neighbor as a classifier, image objects often have high membership values for more than one class, depending on the nearest neighbor function slope, which calculates the fuzzy values depending on the distance of an image object vector to the nearest sample vector in the feature space. Many objects' second-best assignment is nearly as good as their best due to the fact that the feature description in the classes overlap. The reasons are low distances of samples for different classes in the features space. For these objects the classification stability is not very high. The average classification stability can be easily improved by using a lower nearest neighbor function slope.

However, a change in nearest neighbor function slope only influences the absolute membership values and the relative distance between them (classification stability), but not the sequence of the membership to classes depending on the membership values. Classification stability is, therefore, not of decisive significance for the evaluation of a nearest neighbor classification result.

A better differentiation in terms of classification stability can be achieved by using membership functions as a classifier. However, classification based on membership functions is not nearly as able to perform such a differentiation between classes in a multidimensional feature space as nearest neighbor classification is able to do.

## Checking the classification results against predefined test areas

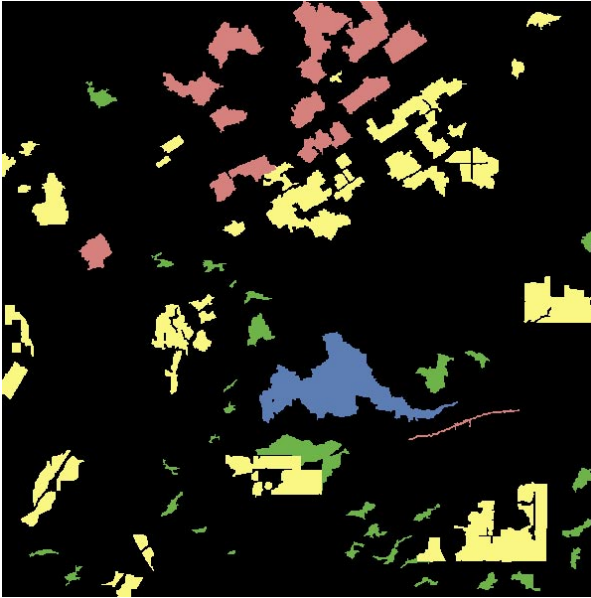
### Loading test areas

1. Select the menu item "Samples > Load TTA Mask..." (training and test areas mask).

In eCognition the term "test areas" is equivalent to TTA mask.

2. Choose the file "TTAMask\_oc.asc" as the actual TTA mask.
3. Choose the file "TTAMask\_oc.txt" as the conversion table.
4. Do not create classes from this conversion table (a class hierarchy has already been created).

The (predefined) TTA mask is now displayed in the document window.

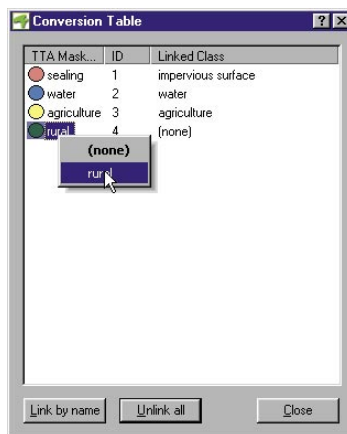


**Note!** If you load a TTA mask into your project you do not automatically create samples from it. The declaration of samples from the TTA mask has to be done explicitly by choosing the menu item “Samples > Create Samples from TTA Mask....”



### Linking classes with test areas

1. Select “Edit Conversion Table...” in the “Samples” menu.
2. If there is no or no correct connection between TTA mask entry and linked class, unlink with the corresponding button and afterwards right-click on the TTA mask entry to choose the class to be linked with.
3. Repeat these steps for the classes *water*, *agriculture*, *rural* and *impervious surface*.



### Comparing the test areas with the actual classification

1. If necessary, open the “Accuracy Assessment” dialog box from the “Tools” menu.
2. Select “Error Matrix based on TTA Mask” as the statistic type.
3. Click “Show statistics.”

Error Matrix based on TTA Mask

User \ Reference Class	impervious surface	water	agriculture	rural	Sum
<b>Confusion Matrix</b>					
impervious surface	11306	8	183	0	11497
water	0	6572	0	0	6572
agriculture	0	0	22726	0	22726
rural	485	6	3225	6636	10352
unclassified	0	0	0	0	0
Sum	11791	6586	26134	6636	
<b>Accuracy</b>					
Producer	0.959	0.998	0.87	1	
User	0.983	1	1	0.641	
Heiliden	0.971	0.999	0.93	0.781	
Short	0.944	0.998	0.87	0.641	
KIA Per Class	0.947	0.998	0.765	1	
<b>Totals</b>					
<b>Overall Accuracy</b>	<b>0.924</b>				
<b>KIA</b>	<b>0.887</b>				


reduce

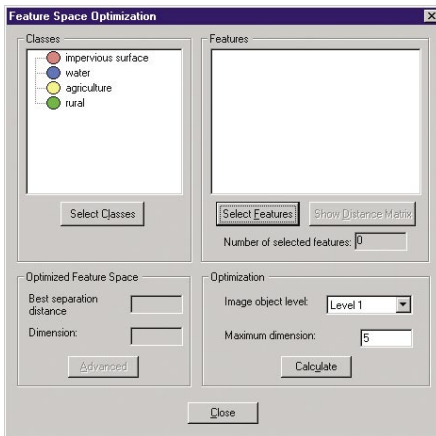
expand

Close

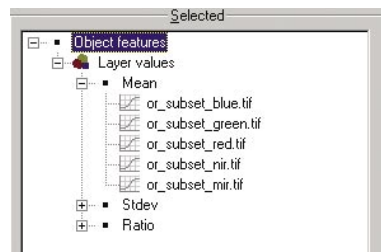
The error matrix provides class-specific statistics. The number of pixels the training and test area mask (TTA mask) contains is given for each class. For the class *water*, the recognition rate is 100 %. There are objects among the class *rural* which have been classified incorrectly (relative to the TTA mask). A total of 6636 rural objects were assigned to *rural*. 485 were assigned to *impervious surface*. The objective of the classification quality assessment has been met when the classification results can be regarded as significant. There is always room for improvement by repeating the process of declaring new samples and classifying. For more information see [Functional Guide > Accuracy Assessment](#).

## Define an optimal feature space

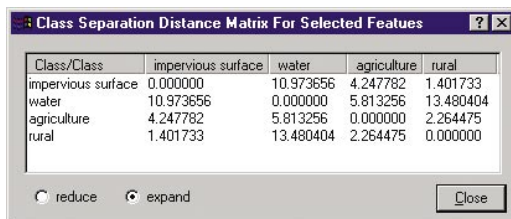
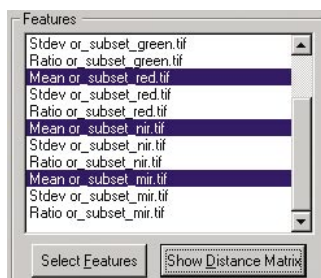
Besides the pure layer mean values, eCognition offers far more features to define classes or feature spaces. To find a well suited feature combination to separate your classes in conjunction with a nearest neighbor classifier, you can use the feature space optimization tool by clicking the  button. Make sure that you have collected samples for each of your classes to separate by scrolling through the “Active class” scroll bar in the “Sample Editor”. You should see several samples for each class. Than call the “Feature Space Optimization” dialog.



Create the feature space to be reduced to an optimum dimension by clicking the “Select Features” button. Choose from the upcoming dialog all layer mean values, except “brightness” and “Max.Diff.”, all layer standard deviations and all ratios:



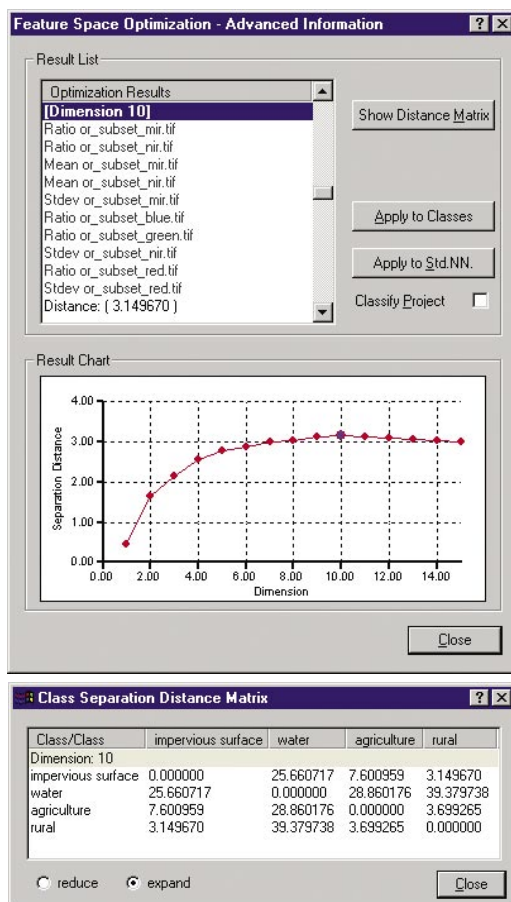
You will see, that your initial feature space has a dimension of 15. In order to take all feature permutations into account enter in the edit control “Maximum dimension” a value of 15 and click “Calculate”. Compare the old feature space described by the layer mean values to the newly created: Select in the “Features” window all mean values and click on “Show Distance Matrix”. The matrix shows the distances of all samples of all classes within feature space.



You can see that some classes are hardly to distinguish by only regarding the spectral mean values. Press “Advanced” to get more information about the optimized feature space:

By default the best separating feature combination is shown. Click on “Show Distance Matrix” to check the distances of the samples within this feature space. Now compare this distance matrix to the matrix of the layer mean values. You will see, that the closest samples in this feature space are more distant than in the pure layer mean feature space. This indicates, that the optimized feature space is better suited to distinguish the desired classes, than the pure layer mean feature space.

Now you can update the feature space of the standard nearest neighbor by clicking on “Apply to Std.NN”. Check “Classify Project” to automatically classify the project with the new standard nearest neighbor. You will notice, that several objects now are unclassified. But at the same time the classification stability increases. If the classification result is still unsatisfying, add and/or remove samples and optimize and classify again.



## Summary

In this exercise we

- loaded raster data into a project,
- performed an image segmentation,
- created a class hierarchy and inserted the standard nearest neighbor as classifier,
- declared sample objects as initial points for a nearest neighbor classification and feature space optimization,
- performed a nearest neighbor classification: click and classify,
- examined the accuracy of the classification result using different methods.
- and created the best separating feature space

If you are comfortable with using these tools, move on to the next exercise. Repeat steps in this part of the tour if you do not feel too sure about them.

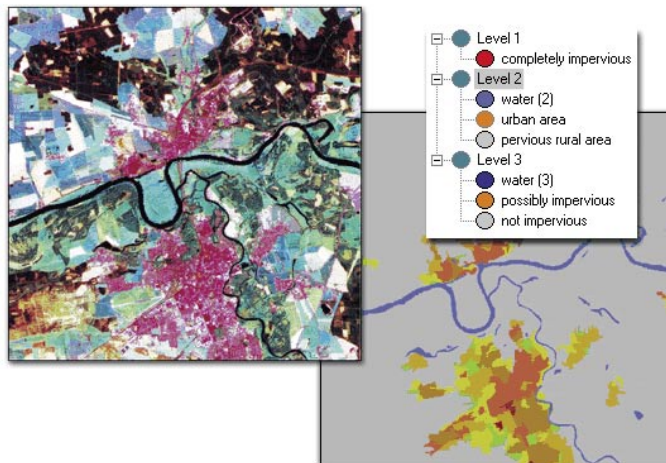


## Tour 2: Analysis of the degree of urban impervious surface

The focus of this exercise lies on the use of multiscale information extracted from different image object levels which differ in resolution. The database is the same as used in the chapter “Take the Plunge.” The goal is to create a map which shows the degree of urban impervious surface of the town of Dessau.


In this exercise you will learn how to:

- perform a classification-based fusion of image objects,
- build an image object hierarchy consisting of several levels of different resolutions,
- prepare sub-scale and super-scale information for a final classification,
- classify using sub-scale and super-scale information,
- use eCognition’s statistics tool,
- export objects as a thematic layer.




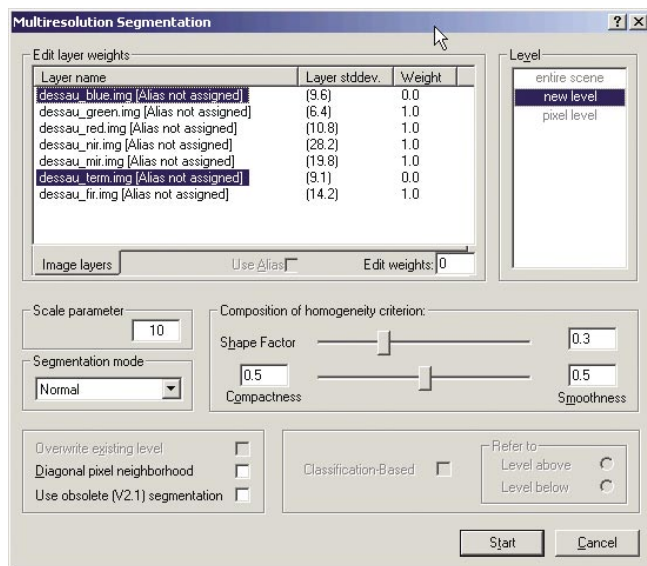
This exercise will begin with the recovery of the classification created in the chapter “Take the Plunge.” As the same class hierarchy and identical samples are needed to achieve an identical classification result, the appropriate class hierarchy will be loaded along with the samples.

## Loading raster layers

1. Start eCognition and choose “Project > New...” from the “Project” menu or click  in the tool bar.
2. In the Dessau sample folder open and visualize the respective raster layers as described in “Take the Plunge.”


## Creating image objects

1. Choose “Multiresolution Segmentation...” from the “Segmentation” menu or click  in the tool bar.
2. Edit the segmentation parameters as shown below. They are identical to those used in “Take the Plunge.”



Your view window should now display a segmented image which is identical to that in “Take the Plunge.”


## Loading the class hierarchy

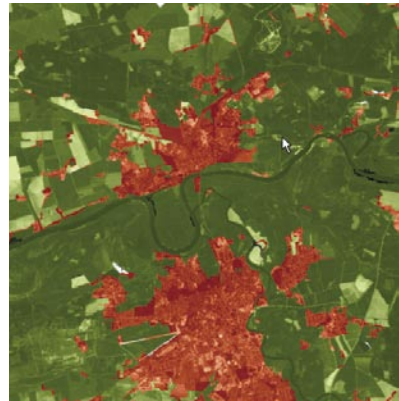
1. Open the class hierarchy window by clicking the  icon or selecting “Open Class Hierarchy...” from the “Classification” menu or “Class Hierarchy” from the “Toolbars & Dialogs” menu.
2. Select “Load Class Hierarchy...” from the “Classification” menu (alternatively right-click in the “Class Hierarchy” dialog and choose “Load Class Hierarchy”).
3. Select “...\data\impervious\dessau.dkb” as your class hierarchy file and open it.

The class hierarchy created in “Take the Plunge” has now been imported along with the identical samples into this project.

## Classifying the image

Edit the classification parameters as shown in “Take the Plunge” (class-related features enabled, five classification cycles) and click .

In the pictures below, the classification of “Take the Plunge” has been recovered. Navigate through the hierarchy of semantic groups using the green arrows  in the tool bar.



Our main interest is the urban area, which is where urban imperviousness has to be expected. In contrast to the previous examples, not only will a thematic map be created, but a map showing the degree of urban imperviousness. This time, information from different segmentation levels has to be used. The first step is creating a reasonable hierarchy of image objects. Each of the image objects created knows its neighbors as well as its super- and sub-objects along with all their attributes and class assignments.



## Editing the structure groups

The structure groups have already been defined in the imported class hierarchy. The following steps explain how to edit them manually.

1. Switch to the “Structure” register in the “Class Hierarchy” editor.

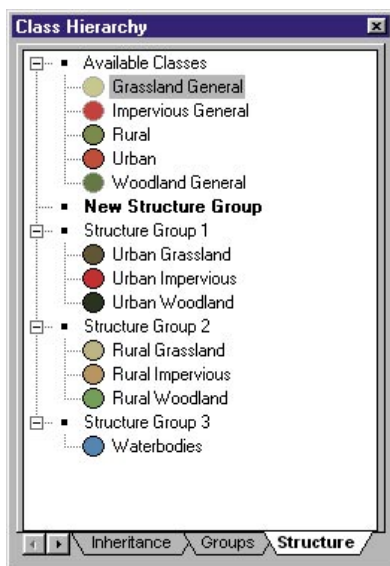
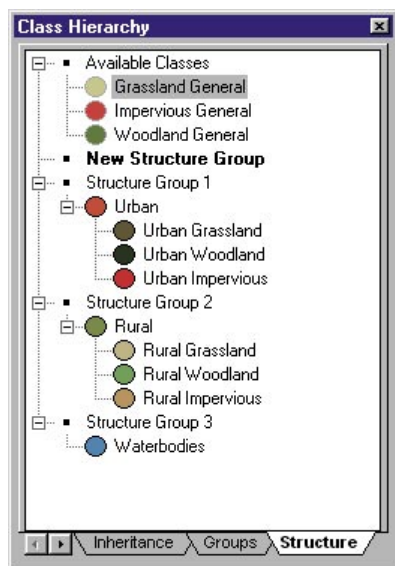
If necessary, execute the three following steps:

2. Create a structure group consisting of all urban classes. To create a new structure group, drag a class and drop it over the text “New Structure Group.” To add a class to an existing structure group, drag and drop it over an existing structure group.
3. Create a new structure group consisting of all rural classes.
4. Create a new structure group consisting only of the class *Waterbodies*.


The following images show two equivalent definitions of these structure groups.

## Performing a classification-based fusion

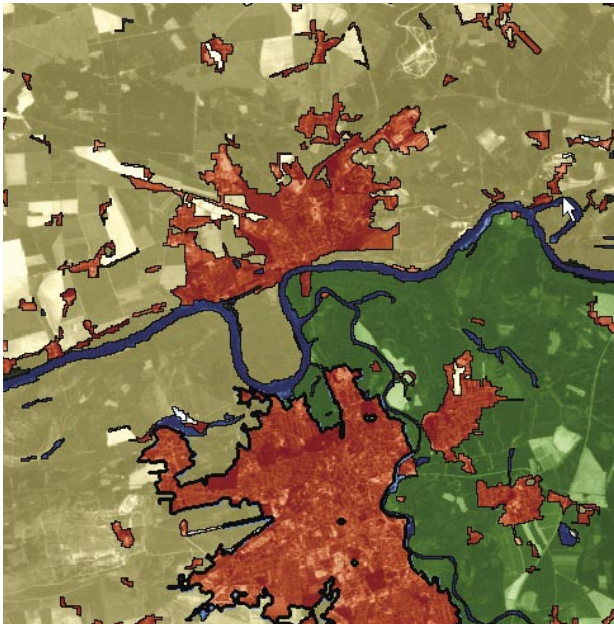
In classification-based fusion, all neighboring objects that belong to the same structure group are combined to larger objects. In this example, lengthy objects can be expected




for water bodies, very large objects for rural areas, and medium sized objects for the the urban areas.

1. Select “Classification-based Segmentation” from the “Segmentation” menu or click  in the tool bar.
2. Activate the check box “Merge image objects in selected level” since the old segmentation is not needed any more.
3. Click “OK” to start the fusion.

The view above shows the merged image objects. They are still classified, since the new image objects are based on the former classification. However, the old class rules cannot be applied to the new segmentation any more.



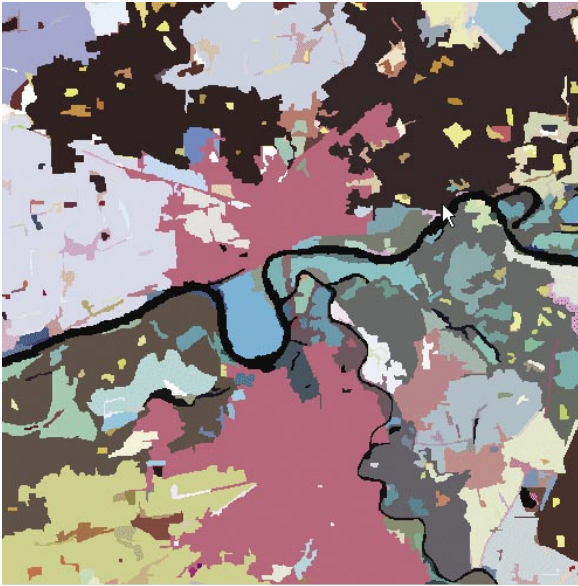
4. To get an impression of the new image objects' sizes, mark an arbitrary object or create polygons by clicking the  icon or choosing “Create Polygons...” from the “Polygon” menu to display the object outlines.

### Deleting the classification

The latest classification provides the knowledge base for the fusion of the image objects to larger, meaningful objects. After performing that fusion, the old classification is no longer needed.

Delete the old class hierarchy (“Classification > Delete Hierarchy”).

The result is displayed in the image below.




### Creating the new image object hierarchy

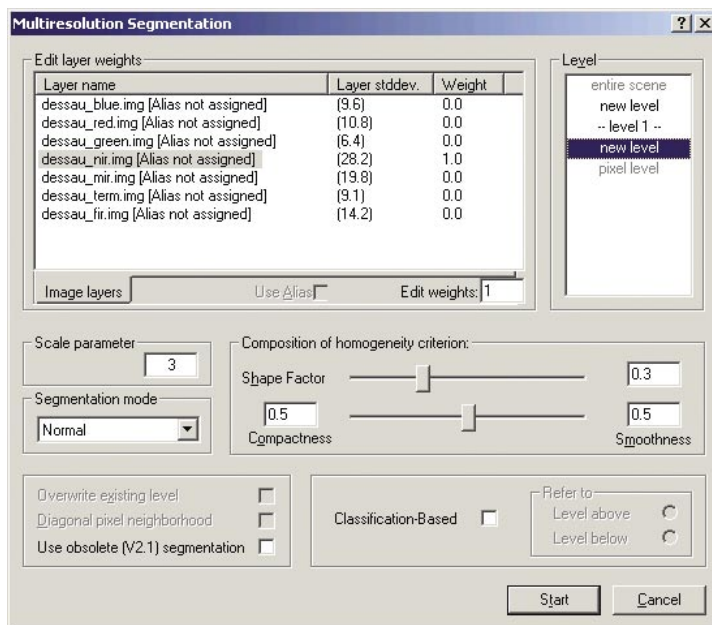
It is reasonable to create an image object hierarchy consisting of three levels: On the highest level only three classes will be distinguished: urban areas, which are the areas of interest, rural areas that are not impervious, and water. This highest image object level already exists due to the classification-based fusion. The objects in this level will become super-objects of all the objects in the two lower levels. Their classification will be used for the classification of the sub-objects by determining whether they belong to a possibly impervious or not impervious area.

The lowest level should be segmented in a very high resolution. The objects should be small enough to represent areas that are fully impervious. The level in the middle

should be segmented in a way that its objects represent areas that appear homogeneous in their degree of urban imperviousness.

### Creating the lowest level of the image object hierarchy (level 1)

1. Open the “Multiresolution Segmentation” dialog (“Segmentation > Multiresolution Segmentation...” or click ).
2. Edit the segmentation parameters as shown below.



Note that the segmentation was only performed on the near infrared layer (“dessau\_nir.img”) of the subset, which is best for displaying impervious structures in this case. You can check this by changing the view setting to one layer.

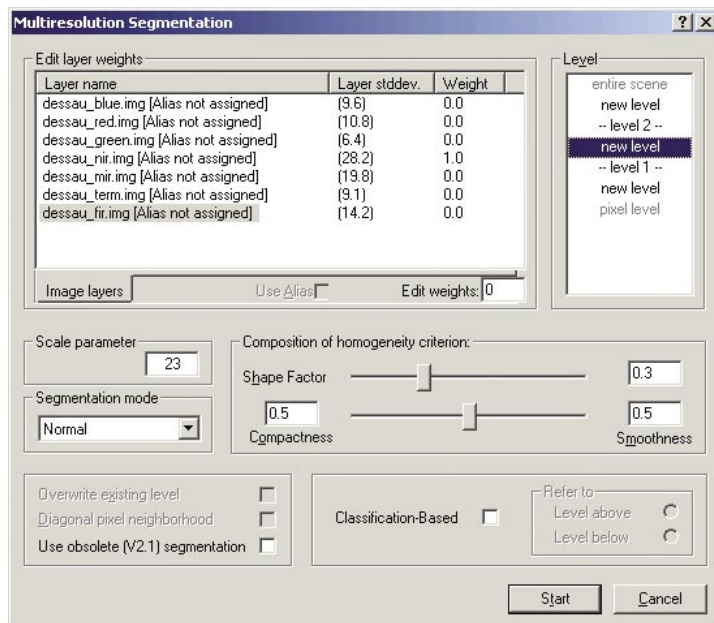
3. Click “Start” to get the segmentation process going.

The last segmentation yielded some image objects which can be classified as completely impervious.

## Creating image object level 2

Once again, only the near infrared layer is used for the segmentation.

1. Open the “Multiresolution Segmentation” dialog and edit the segmentation parameters as shown below. You are creating a new level, so be sure to highlight “new level” instead of “level 2.”



2. Start the segmentation process.

This segmentation yields objects which are relatively homogeneous in their degree of imperviousness.



Together with the classification-based fusion, you have now created the three image object levels of different resolution shown below. You can navigate through them using the vertical black arrows   in the tool bar.



Image object level 1 (unclassified)



Image object level 2 (unclassified)

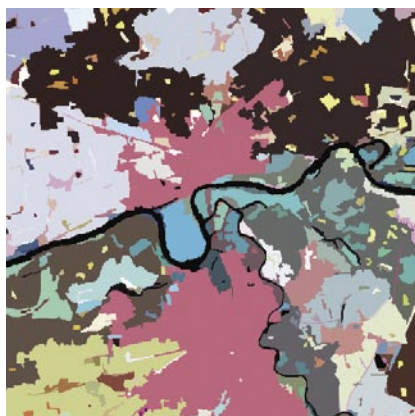


Image object level 3 (unclassified)

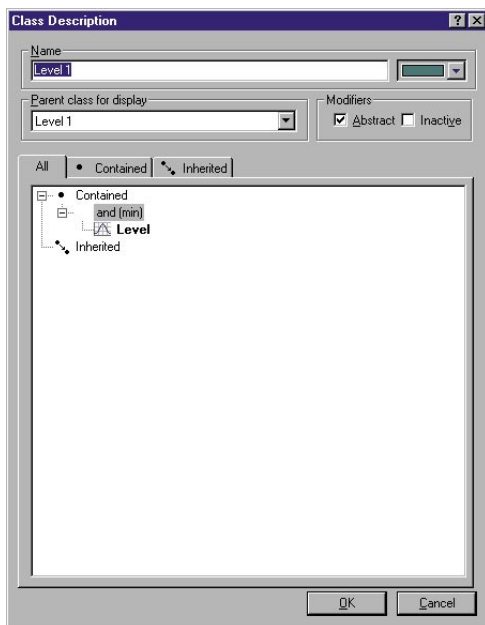
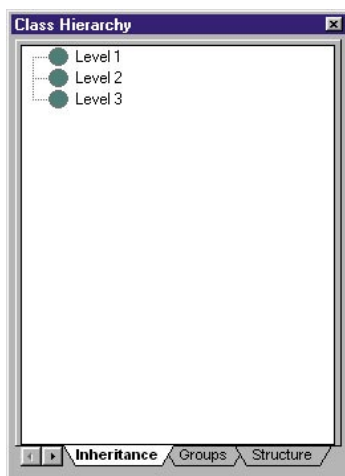
## Creating abstract base classes

In this exercise different classifications will be applied to different image object levels. Thus, it is necessary to ensure that the class hierarchy is kept transparent so that the work can still be surveyed. To achieve this, three abstract classes will be introduced, each inheriting the number of the image object level for which a class will be defined.

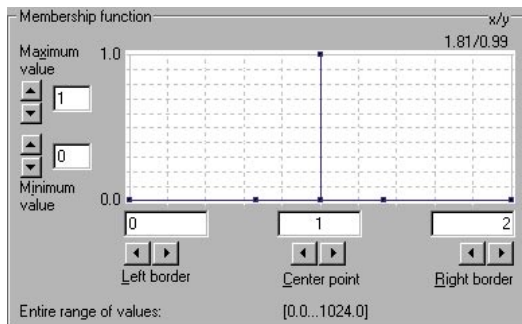
1. Open the “Class Hierarchy” editor. Go to the “Inheritance” register.

In this exercise the hierarchy of semantic groups will be identical to the inheritance hierarchy.

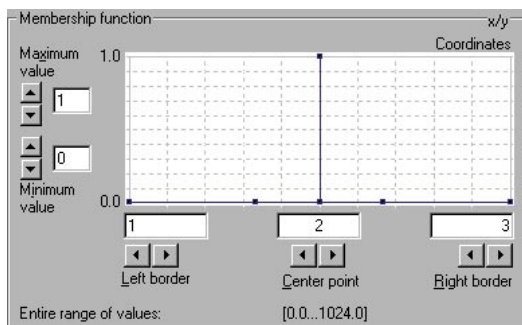
2. Delete the old class hierarchy (menu item “Classification > Delete Class Hierarchy”).
3. Create three new classes and call them *Level 1*, *Level 2* and *Level 3*.
4. For all three classes, insert the feature “Object features > Hierarchy > Level.”



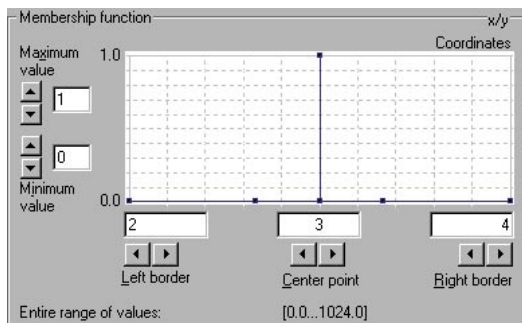
5. Edit the membership functions as shown below.



Level 1



Level 2



Level 3

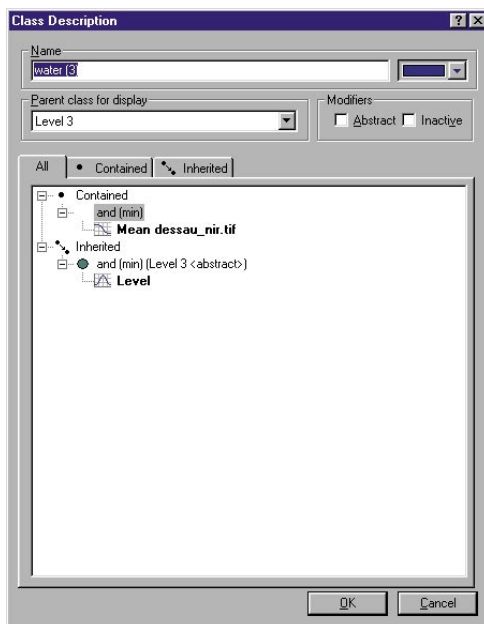
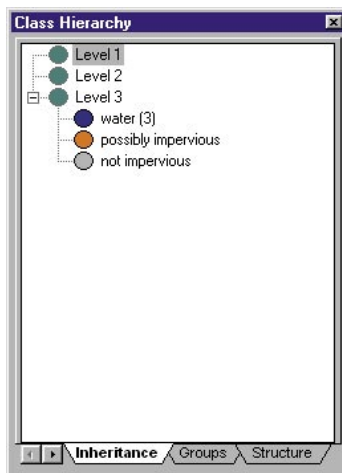


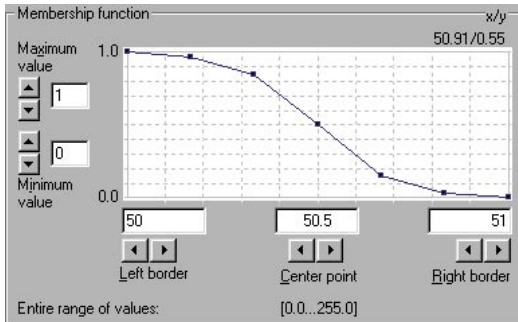
## Classifying image object level 3

1. Navigate to image object level 3.
2. Create three new classes and call them *water (3)*, *possibly impervious* and *not impervious*.
3. Make these three classes child classes of *Level 3* in the inheritance and in the groups hierarchy. You have to do this since the inheritance and the groups hierarchy are independent of each other and in this example both information is needed. For further discussion about the class hierarchy please refer to the chapter “Functional Guide - Classification Advanced” (page 250).
4. Open the class description dialog of class *water (3)*.

On image object level 3, water objects can easily be described using a spectral feature, the near infrared mean value (“Object features > Mean > dessau\_nir.tif”). To fully assign an object to *water (3)*, its near infrared mean value must be 50 or smaller.

5. Edit the class description and membership function of *water (3)* as shown below.



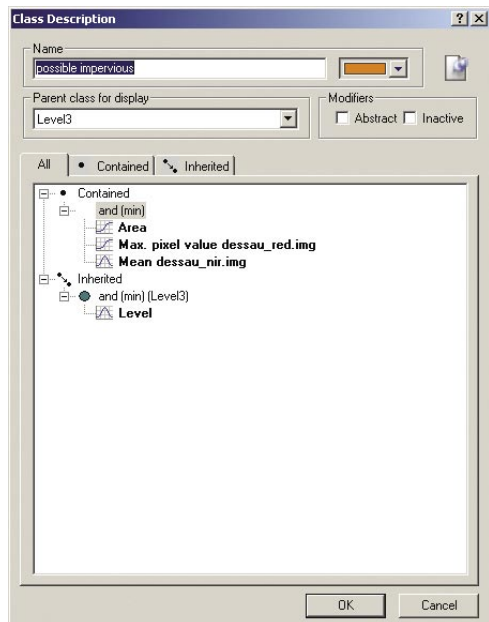


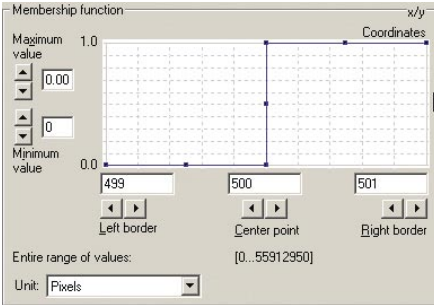
6. Open the “Class Description” dialog of the class *possibly impervious*.

All objects that urban imperviousness can occur in are assigned to *possibly impervious*. It is known from the former classification and fusion that these are objects that have been created out of the former urban objects. With that knowledge, the following class rules can be built. Objects of the class *possibly impervious*

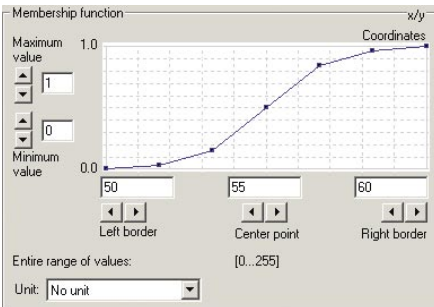
- must be of 500 pixels or larger in size
- must have a red maximum pixel value larger than 60.
- must have a near infrared value larger than 60 and smaller than 75.

7. Edit the class description and the appropriate membership functions as shown here.

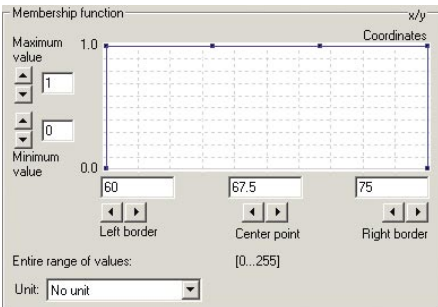




possibly impervious: "Area"



possibly impervious: „Max. pixel value ssau\_red.img“

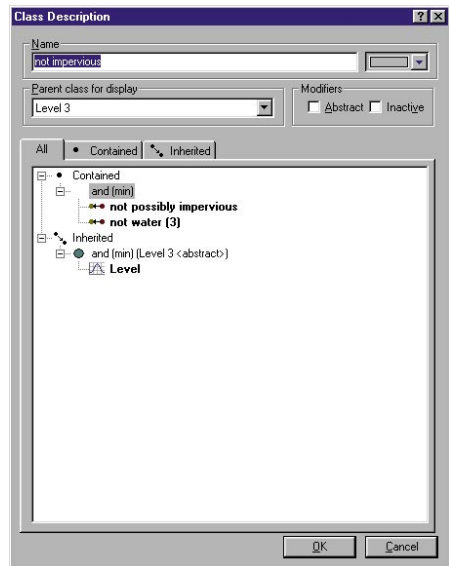


possibly impervious: „Mean dessau\_nir.img“

8. Open the “Class Description” dialog of the class *not impervious*.


All remaining unclassified objects of image object level 3 will be assigned to this class, so the definition of the class rules is simple: objects of the class *not impervious* are neither objects of the class *possibly impervious* nor objects of the class *water* (3).

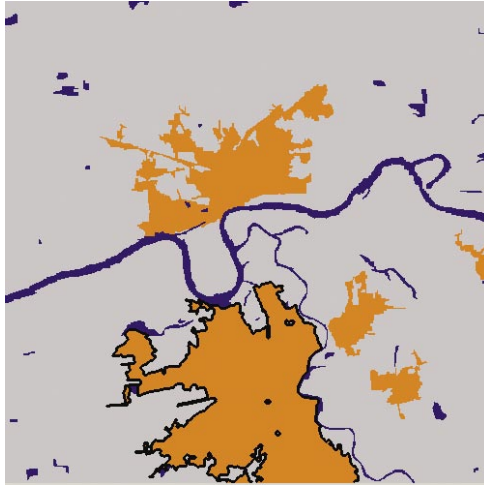
9. Edit the class description of *not impervious* as shown here. Insert the expression „similarity to classes“ *possibly impervious* and *water*. For the „not“ statement you have to invert the expression.




10. Navigate to level 3 and edit the classification parameters as shown below



11. Click  to start the classification.



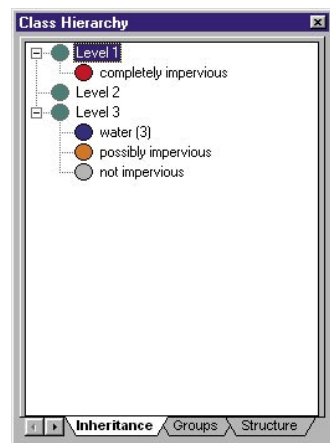
The classification result should look like this. To verify your classification with the image below, make sure you are in the correct level of the groups hierarchy. Use the green arrows  to move to the pertinent level.

This classification provides super-scale information for the classification of the lower levels: Each sub-object in a lower level now knows if it belongs to a possibly impervious area, a nonimpervious rural area or water. This way, the super-scale classification can aid us in future classifications.

## Classifying image object level 1

In order to create image objects that represent single structures like rows of houses or streets in urban environments, a high resolution will be chosen for the segmentation in level 1. Now the objective is to determine the objects among them that represent completely impervious or dense spots.

1. Navigate to image object level 1.
2. Create a new class called *completely impervious*.
3. Make the new class a child of the class *Level 1*.

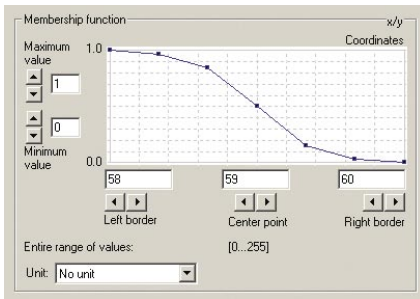
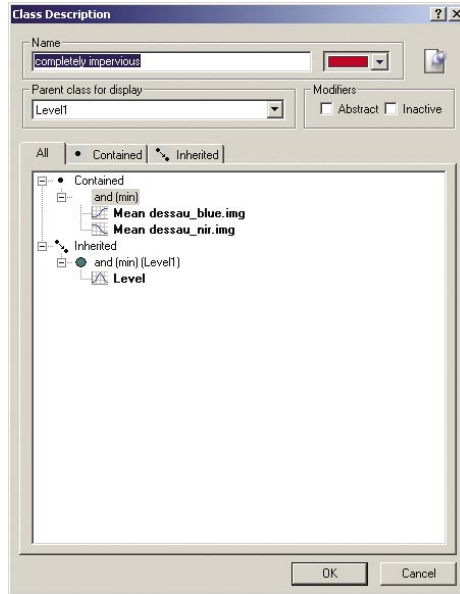


4. Open the class description dialog of the class *completely impervious*.

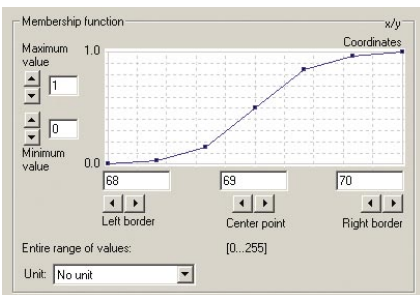
It has already been ascertained that the near infrared layer is best for analyzing urban structures. It is advisable to extract features from it for the classification of dense areas as well. As a class rule the following can be formulated:

- Objects of the class *completely impervious* have a near infrared mean value less than or equal to 59.
- Objects of the class *completely impervious* have a mean value in layer “dessau\_blue.img” of 70 or higher.

5. Edit the class description and the respective membership functions as shown below.



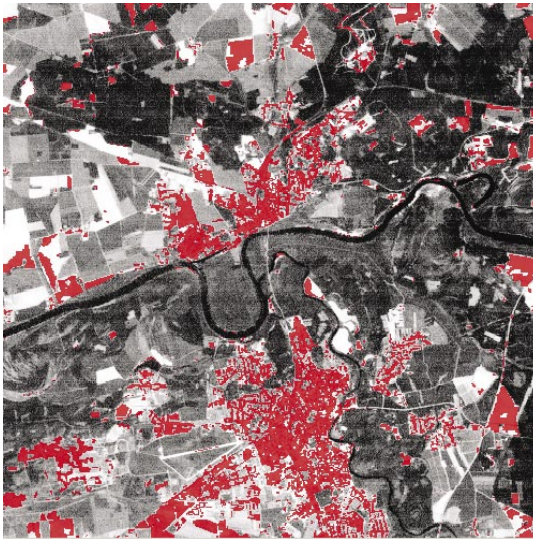
completely impervious: "Mean dessau\_nir.img"



completely impervious: "Mean dessau\_blue.img"

## 6. Classify level 1.

The result should look like the following image (layer mixing: one layer gray).



Urban structures are well represented, but on the other hand, too many objects have been assigned to *completely impervious*. Especially coniferous areas show low values in the near infrared part of the spectrum. Two options are available to tackle this inconsistency: Super-scale information could be applied, thus only classifying urban areas. The second option is to wait until classifying level 2 to do this, since it has a set of class rules which are easier to survey.

For now, this classification will be kept and you will move on to the next step.

## Classifying the actual degree of urban imperviousness

### 1. Navigate to image object level 2.

It is now necessary to define classes in a way that the previously gained information can be used. If you look at the image below, you will notice that the information provided by level 3 (super-scale) tells you for which objects you must expect impervious areas and for which the information provided by level 1 (sub-scale) must therefore be used. Information provided in level 1 can be used by calculating the relative area of sub-objects classified as *completely impervious*.



Image object level 1 (classified)

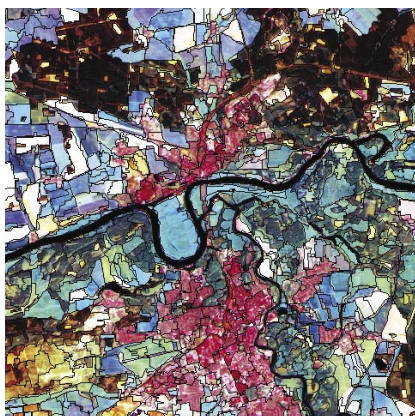


Image object level 2 (unclassified)

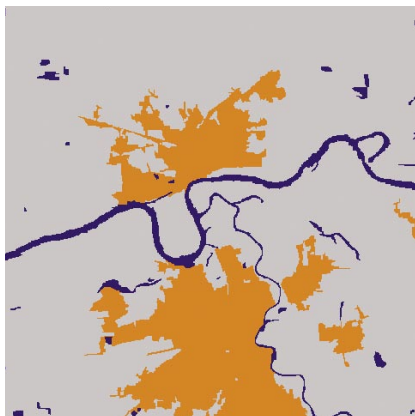


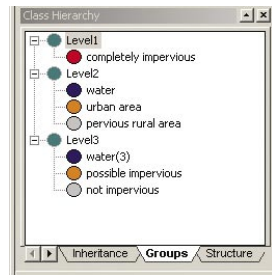
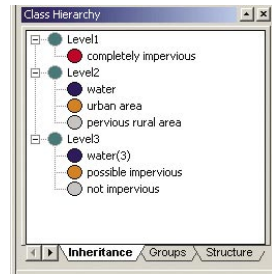
Image object level 3 (classified)



## Making use of super-scale information

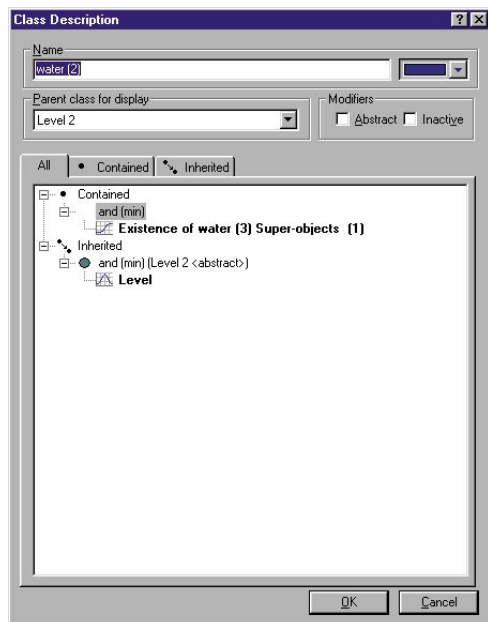
In which objects of image object level 2 urban imperviousness can occur can be determined by using super-scale information.

1. Create three new classes and name them *water (2)*, *urban area* and *pervious rural area*.
2. Make them child classes of the class *Level 2*.
3. Open the class description dialog of the class *water (2)* in the inheritance and in the groups hierarchy. You have to do this since the inheritance and the group hierarchy are independent of each other and in this example both information is needed. For further discussion about the class hierarchy please refer to the chapter “Classification Advanced” (page 250).

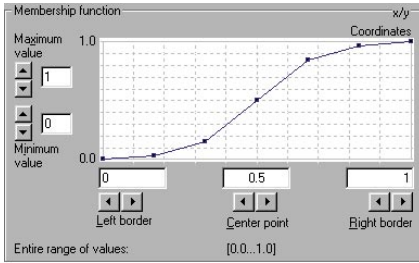


Now use can be made of super-scale information. All objects whose super-object is classified as *water (3)* will be assigned to *water (2)*.

4. Insert the feature “Class-related features > Relations to super-objects > Existence of *water (3)* (1).”
5. Edit the membership function as shown below.





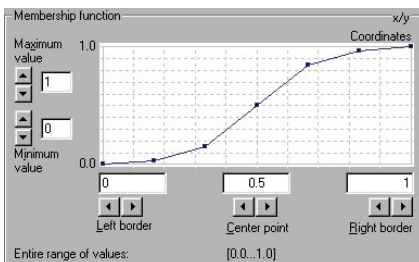
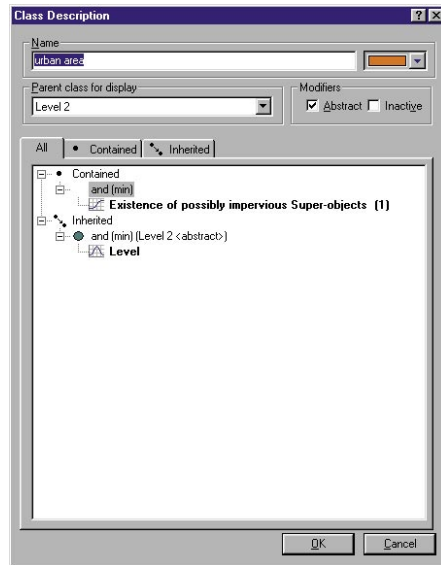


water [2]: "Existence of water [3]  
Super-objects (1)"

The selected feature is a boolean feature, i.e., only two values are being returned. If the value is 0 (false), the object does not have a super-object of the given class. If the value is 1 (true) the object has a super-object of class *water* (3) and is assigned to *water* (2).

6. Edit the class description and respective membership function of the class *urban area* in an analogous way.

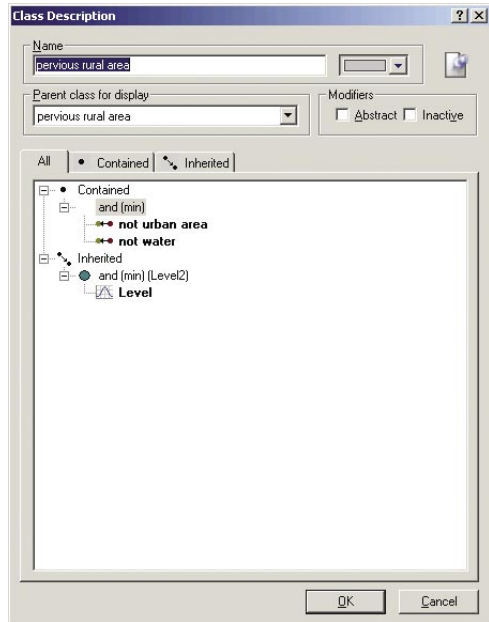
An object will be assigned to *urban area*, if it has a super-object classified as *possibly impervious*.



urban area: "Existence of possibly impervious super-objects (1)"

By analogy, an object is assigned to the class *pervious rural area*, if it is *not urban area* and *not water*.

7. Edit the class description and the appropriate membership function of the class *pervious rural area*.



### Making use of sub-scale information

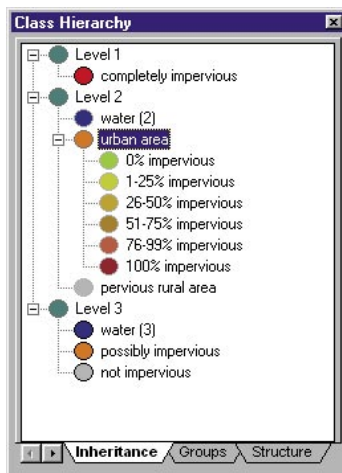
Urban imperviousness only occurs in areas represented by objects classified as *urban area*. In the next step classes will be created which represent the actual degree of imperviousness in these areas, determined by the use of sub-scale information.

For each object classified as *urban area* the part of its sub-objects' area classified as *completely impervious* will be calculated.

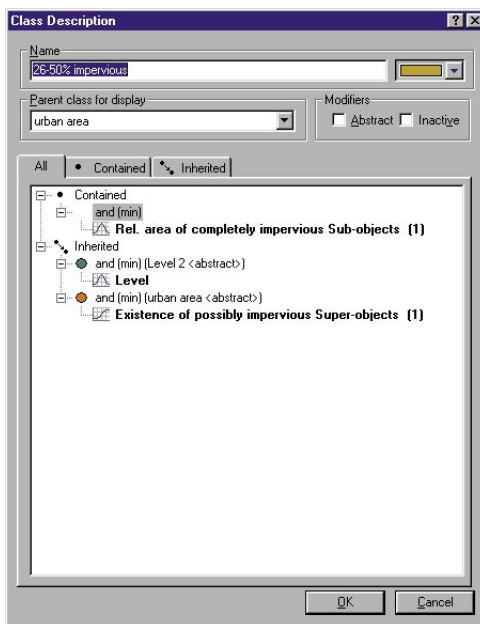
1. Create six new classes and call them

- 0 % impervious
- 1-25 % impervious
- 26-50 % impervious
- 51-75 % impervious
- 76-99 % impervious
- 100 % impervious

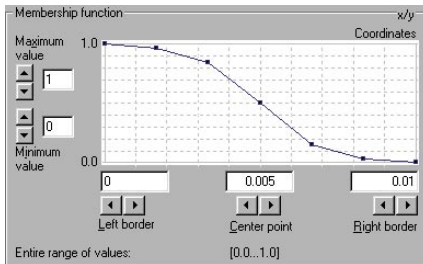
2. Make these classes children of *urban area*.



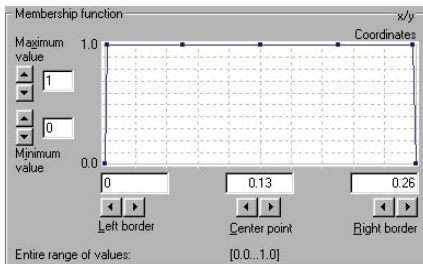
3. Insert the feature “Class-related features > Relations to sub-objects > Rel. area of completely impervious (1)” into the class description of each of these classes.



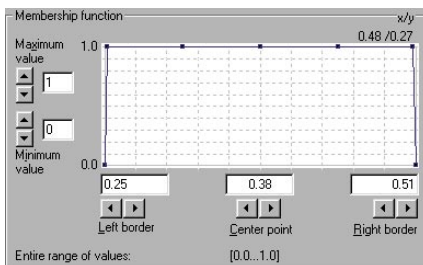
4. For each class edit the membership function according to the degree of imperviousness they represent.



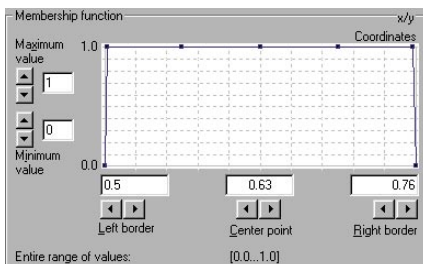
0% impervious: "Rel. area of completely impervious sub-objects [1]"



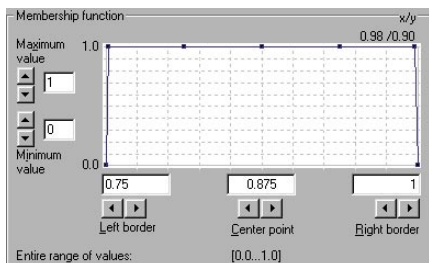
1-25% impervious: "Rel. area of completely impervious sub-objects [1]"



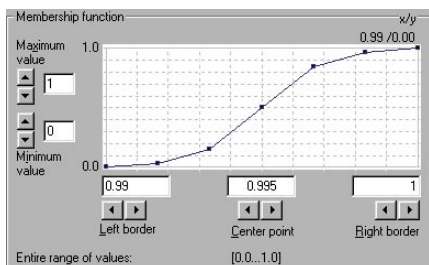
26-50% impervious: "Rel. area of completely impervious sub-objects [1]"



51-75% impervious: "Rel. area of completely impervious sub-objects [1]"



76-99 % impervious: "Rel. area of completely impervious sub-objects [1]"




100 % impervious: "Rel. area of completely impervious sub-objects [1]"

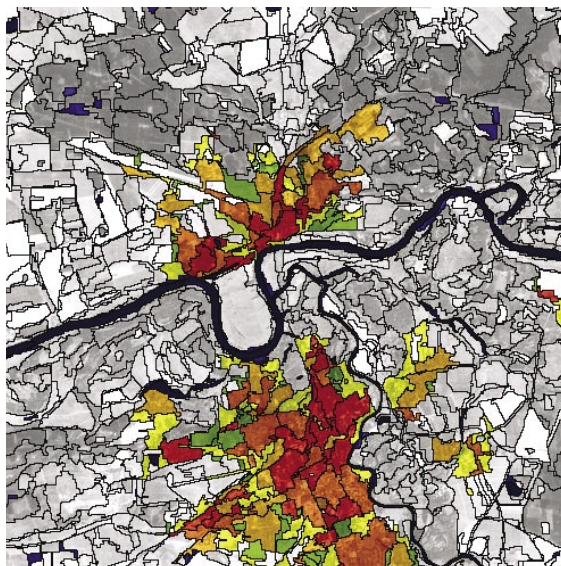
You have now created and edited all new classes.

## Classifying image object level 2

1. Enable classification with class-related features (five cycles). The tool bar should look this:



2. Click  to get the classification process going.





Here is an overview of how information from different segmentation levels was used:

Image object level 1 [classified]

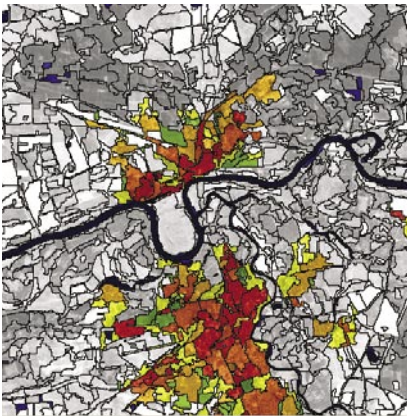


Image object level 2 [classified]



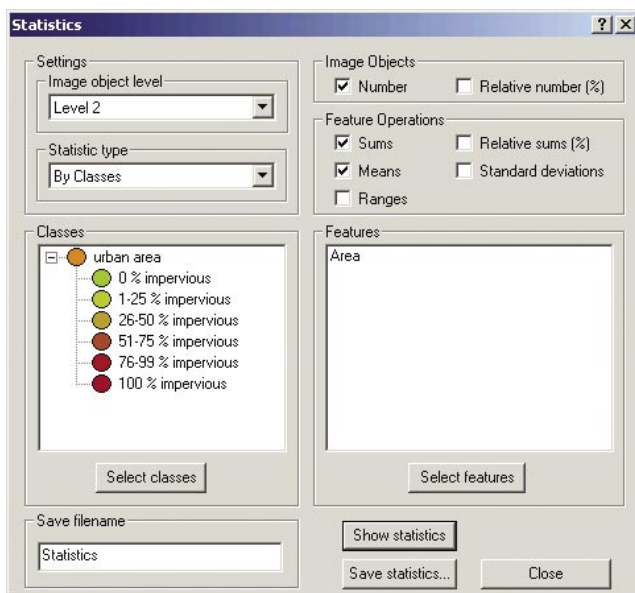
Image object level 3 [classified]

The classification of the degree of imperviousness in urban areas has now been completed. Now move on to the analysis of this classification result. eCognition provides a powerful statistics tool for tasks like this.

### Analyzing the classification result

As an example, you can examine what share of the urban area and of the entire subset the single impervious degrees cover.

1. Select “Statistics...” from the “Tools” menu.
2. Level 2 is the level of interest. Select it under “Image object level.”
3. Click “Select classes,” then select the classes *Level 2, urban area* and the classes representing the different degrees of imperviousness.
4. Click “Select Features” and select the feature “Object features > Shape > Area.”  
Double-click the desired feature to move it from “Available” to “Selected.”
5. Click “Show statistics.”



A matrix is presented showing the number of objects of each class along with the chosen feature statistics. The first column names the class, the second the number of objects classified as such, the third gives you the total area in pixels covered by objects of the given class and the fourth column shows you the average size of the so classified objects (also in pixels).

Class	Objects	Sum: Area (Px)	Mean: Area (Px)
Level2	1196	248502	207.7776
urban area	167	44320	265.3892
0 % impervious	39	4606	118.1026
1-25 % impervious	44	10632	241.6364
26-50 % impervious	31	10260	330.9677
51-75 % impervious	23	9578	416.4348
76-99 % impervious	28	9090	324.6429
100 % impervious	2	154	77

Now you can easily solve the problem by dividing the total areas of the impervious degree classes by the total areas (*Level 2* or *urban area*). These are the results:

Class	share of urban area [%]	share of entire subset [%]
0 % impervious	10.4	1.9
1-25 % impervious	24.0	4.3
26-50 % impervious	23.1	4.1
51-75 % impervious	21.6	3.9
76-99 % impervious	20.5	3.7
100 % impervious	0.4	0.1

This is only a simple example of how the statistics tools of eCognition can be used. eCognition provides a powerful statistics tool for image analysis along with the variety of features that can be calculated and summarized for each class.

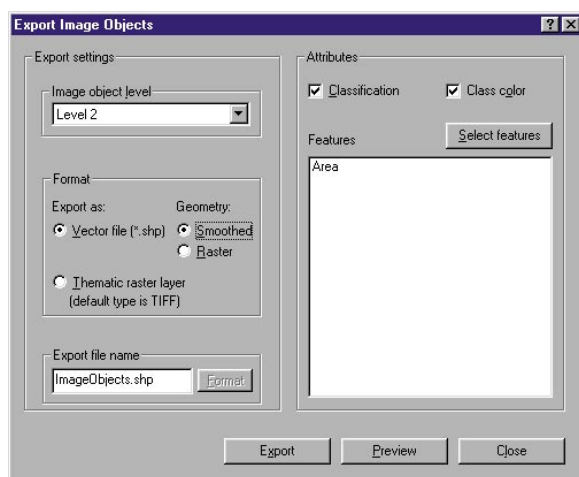
### Exporting image object level 2 as a thematic layer

Image object layers can be exported into new eCognition projects or into a geographic information systems (GIS) as thematic layers.

1. Select “Image Objects...” from the “Export” menu.
2. Select Level 2 under “Image object level.”



3. Choose the format you want to export the layer in. Notice that you can only choose “Vector file (\*.shp)” if you have created polygons before.
4. Choose “Classification” and “Class color.”
5. Click “Select Features” and choose the feature “Object features > Shape > Area” as an additional attribute.
6. Click “Preview” to display the attribute table.



**Attribute table**

ID	R...	G...	Bl...	Best Class	B...	B...	Second Class	S...	S...	Third Class	T...	T...	Area
445	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	168
446	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	480
447	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	68
448	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	152
449	206	171	49	26-50% impervious	12	1	unclassified	-1	0	unclassified	-1	0	149
450	186	138	48	51-75% impervious	13	1	unclassified	-1	0	unclassified	-1	0	731
451	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	214
452	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	249
453	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	417
454	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	534
455	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	198
456	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	91
457	209	103	67	76-99% impervious	14	1	unclassified	-1	0	unclassified	-1	0	237
458	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	22
459	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	84
460	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	528
461	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	155
462	0	0	255	water (2)	7	1	unclassified	-1	0	unclassified	-1	0	10
463	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	576
464	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	128
465	192	192	192	pervious rural area	9	1	unclassified	-1	0	unclassified	-1	0	35
466	0	0	255	water (2)	7	1	unclassified	-1	0	unclassified	-1	0	18
467	192	192	192	51-75% impervious	13	1	unclassified	-1	0	unclassified	-1	0	420

reduce expand Close

Each object of level 2 received an object ID, which is shown in the first column. It is followed by the RGB color values (of the actual view in the view window). The next columns contain the three best class assignments along with their percentage as attribute (–1 is the value for no assignment). The features chosen as attributes are at the far right.

7. Click “Export.”

For more information see [Functional Guide > Importing and Exporting](#).

## Summary

In this exercise you

- imported an existing class hierarchy,
- performed a classification-based fusion of image objects,
- created an Image objects hierarchy consisting of three levels of different resolution,
- performed classifications on the highest and lowest level of the image objects hierarchy,
- classified the degree of urban imperviousness with the use of sub- and super-scale information,
- performed an imperviousness analysis using feature statistics,
- exported an image object level as a thematic layer.



## Tour 3: High Resolution Aerial Scan

This exercise will demonstrate the ability of eCognition to work with high resolution data. The subset is an aerial scan and shows the village of Offenberg in southern Germany. It consists of three spectral layers and an 8 bit surface model. In addition a thematic layer produced in a Geographic Information System (GIS) will be used. This thematic layer shows plots of land in and around Offenberg.

In this exercise you will learn how to:

- import a thematic layer and utilize it for segmentation,
- classify high resolution data,
- use a surface model for classification,
- perform analysis using the thematic layer,
- perform classification-based segmentation (border optimization and image object extraction).



## Loading the raster data

1. Start eCognition, choose “Project > New...” and switch to the directory “...\data\airalscan\.”

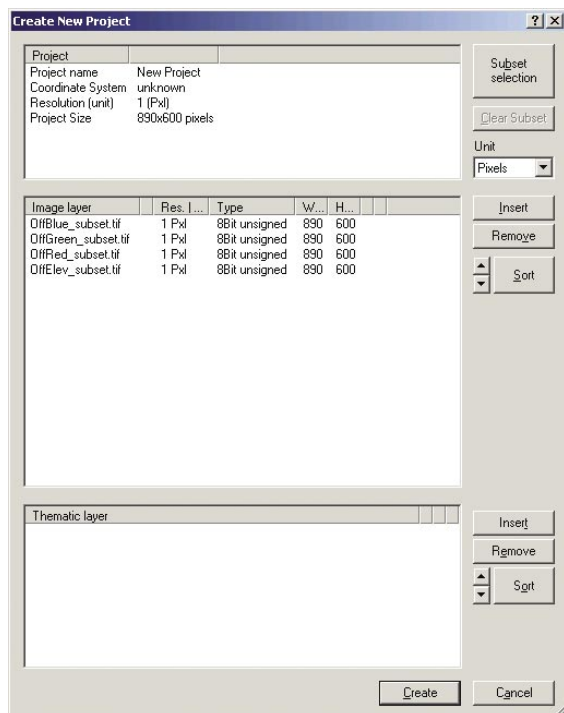
2. Select the image layers

- OffBlue\_subset.tif
- OffGreen\_subset.tif
- OffRed\_subset.tif
- OffElev\_subset.tif

and open them.

3. Change the order of the image layers according to their sequence in the spectrum.  
Move the surface model to the last position as shown below.

4. Do not click “Create” just yet, because the thematic layer first has to be loaded.




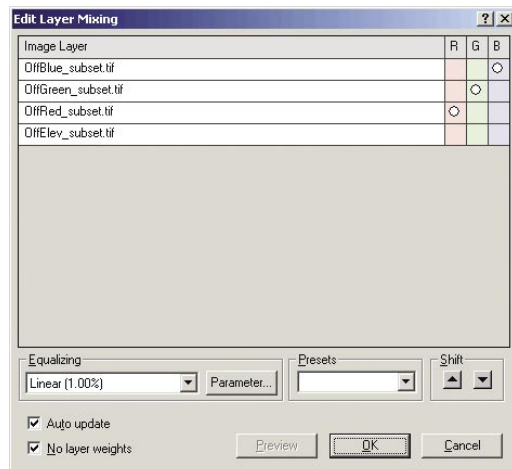
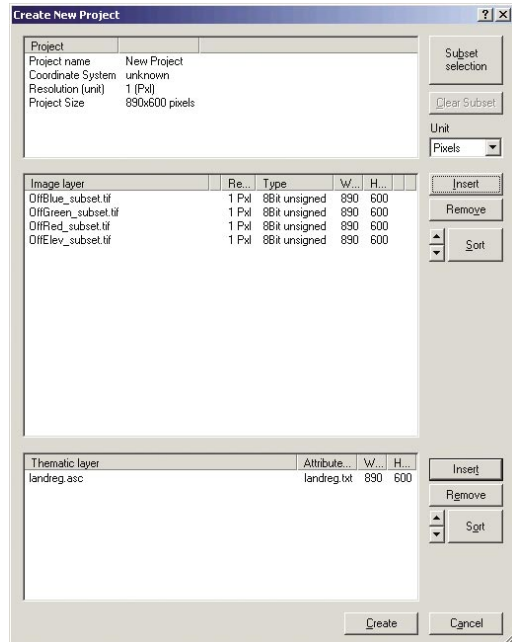
## Loading the thematic layer

Keep in mind that it is also possible to load a thematic layer later. But to use the information of the thematic layer it is necessary to perform a new segmentation afterwards.

1. Click the lower “Insert” button to the right of the list window for thematic layers.
2. Select the file “landreg.asc” as thematic layer and click “Open” (You may have to change the file format selection to \*.asc).
3. Select the file “landreg.txt” as the complementing attribute table and open it.

The thematic layer is now displayed in the “Create Project” dialog.


4. Click “Create.”
5. Choose “Layer Mixing...” from the “View” menu, double click in the right column of the “View Setting” dialog or click  to change the color composition.
6. Combine the image layers and the colors as shown here.
7. Click “OK.”



The data will be shown in the view window in true colors.



### Visualizing the thematic layer

1. Open the “View Settings” dialog (“Toolbars & Dialogs > View Settings” or click ).
2. Change the view to the thematic layer “landreg.asc” by a left mouse-click on “Layer,” choose “landreg.asc” by a left mouse-click on “Mode,” and choose “Layer” if it is not yet chosen.




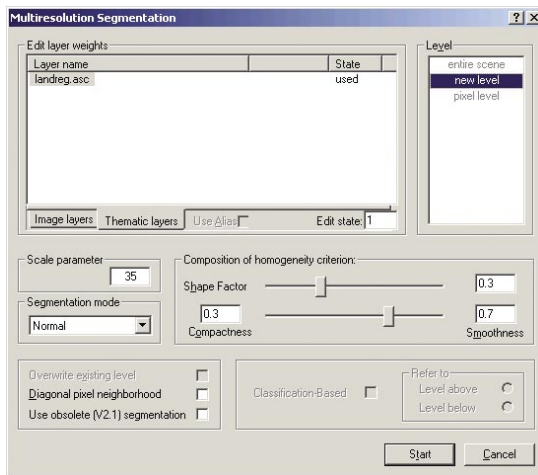
If this thematic layer is activated during the segmentation process, its polygons will form the super-objects of the image objects created by multiresolution segmentation.

3. Change the view back to the pixel view.

### Creating image objects

In this exercise a thematic layer containing cadastral information is imported. The segmentation dialog will look slightly different: an additional tab register in which imported thematic layers can be activated and deactivated for segmentation will appear.

1. Open the “Multiresolution Segmentation” dialog (menu item “Segmentation > Multiresolution Segmentation...” or click ).
2. Change to the tab register “Thematic layers.”
3. Make sure that the thematic layer “landreg.asc” is used for the segmentation. If it is not applied, insert 1 in the field “Select and edit state.”
4. Change back to the “Image layers” tab register.



Three different image object levels will be created. The actual classification will be carried out on image object level 2. Image object level 3 will consist of image objects that are identical to the polygons of the thematic layer. Image object level 1 will form sub-objects that will be suitable for demonstration purposes. The thematic layer will be used for the creation of each of the image object levels.

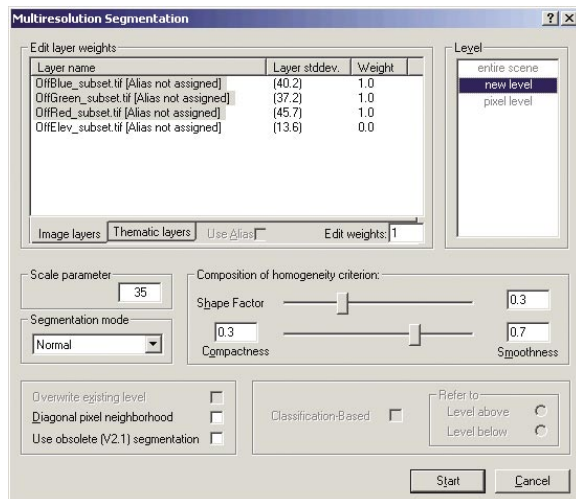
### Creating the first image object level


In this level, the actual classification will be performed. So this level should be created first.

**Note!** Since it is possible to create levels in any position within the image object hierarchy at any time, it is recommended that you create the level in which to perform the classification first and the levels used to help with the classification afterwards. By doing so, you will not run the risk of having the most important object level negatively influenced by sub- or super-objects.



1. Edit the segmentation parameters as shown below.
2. Click “Start.”



3. Display the segmented image with outlines. To display outlines it is first necessary to create polygons (Click  or select “Create Polygons...” in the “Polygons” menu).



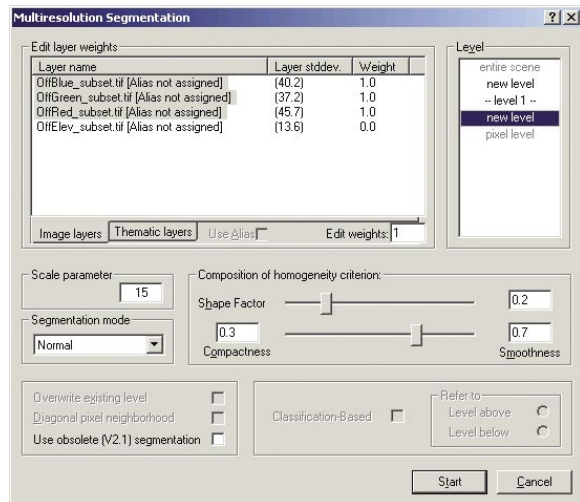
Note that the buildings in particular are well represented by the image objects. Some of the object outlines form straight lines. This is due to the application of the thematic layer. If a thematic layer is used during the segmentation process, no objects that reach beyond the border of a polygon or are larger than such a polygon will be created.

## Creating the second image object level

As mentioned above, this image object level will be needed to perform a classification-based segmentation of the level created before.

**Note!** Up to now you have created only one image object level. Accordingly this level is named level 1. By creating this new level as a sub-level of level 1, the former level 1 will be named level 2 and the new level will take the position of level 1.

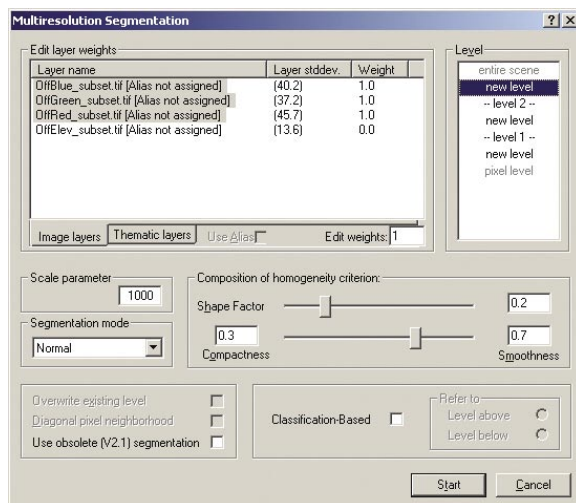
1. Open the “Multiresolution Segmentation” dialog again.
2. Edit the segmentation parameters as shown below.
3. Click “Start.”



## Creating the third image object level

This level will be used later on to perform analysis based on the cadastral information. The image objects have to completely match the polygons of the thematic layer. No image objects larger than the polygons of the thematic layer should be segmented. You have to choose a very high scale parameter to achieve this.

1. Once again, open the “Multiresolution Segmentation.”
2. Edit the segmentation settings as shown below.
3. Click “Start.”



The view window, displayed in pixel mode with outlines, shows an image segmented just like the thematic layer.



Summary: You now have an image object hierarchy containing three image object levels. Level 1 is used as a basis for the classification-based segmentation performed later on. Level 2 is the level in which the actual classification is to be performed and level 3 is the level in which the imported thematic layer is to be analysed regarding land cover.

Firstly the classification of image object level 2 will be performed. The first step is to assign objects to three base classes. One class will represent all objects that appear green (vegetation), the second will represent all objects that appear red (roofs and yards) and

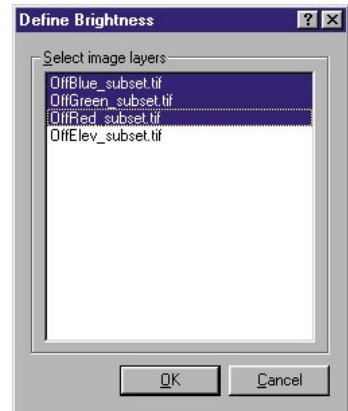
the last will represent all objects that appear grey (roofs, yards and roads). Some objects appear very dark due to shadows and cannot be assigned to any of the classes mentioned. For those objects, a fourth base class will be created.

After all objects have been assigned to one of the base classes, the classes will be further distinguished. The surface model will be used for distinguishing roofs and yards.

### Defining image layers for the calculation of brightness


The brightness of an image object is calculated using all image layers. In this example, however, one image layer (the surface model) does not contain spectral information and thus is of no use for the computation of brightness. Therefore, you have to define the layers which contain spectral information.

1. To define the brightness choose “Classification > Advanced Settings > Image Layers for Brightness...” from the menu.
2. Mark the image layers containing spectral information.
3. Click “OK.”



### Loading the class hierarchy

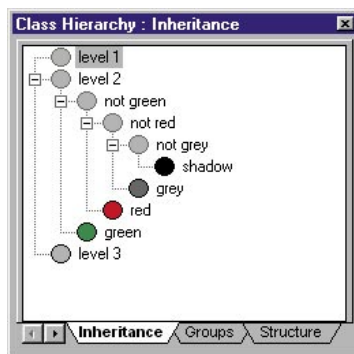
Since this exercise focuses on the use of a surface model for classification, the application of a thematic layer, and the possibilities of classification-based segmentation which eCognition offers, no class hierarchy will be created in this section. Rather, the one consisting of the aforementioned base classes will be loaded.

1. Open the “Class Hierarchy” dialog (“Toolbars & Dialogs > Class Hierarchy...” or click ).
2. Choose the menu item “Classification > Load Class Hierarchy...” or right-click in the “Class Hierarchy” window to load the class hierarchy.
3. Select the file “as\_baseclasses.dkb.”

- If the imported class hierarchy does not match the one displayed below (but would deliver the identical classification), you can always sort the classes manually. To do so choose “Classification > Sort Classes,” activate “Manually” and “Enable Sort Mode.” Just drag the classes to the pertinent position. Remember to deactivate the “Enable Sort Mode” to make classes child classes of any other class.

There are classes which determine the image object level on the second level of the “Inheritance” hierarchy for which they and their child classes are valid. There are no child classes for image object levels 1 and 3 yet.

Membership functions are used throughout the whole example as a classifier. In this case the ratio of the respective color classified is the feature of choice in the class descriptions. To classify *green*, for example, it would be straightforward to use the feature “Mean OffGreen\_subset.tif,” but this causes problems with objects representing shadows. This problem can be avoided by using the feature “Ratio OffGreen\_subset.tif”.



- Select “Feature View” from the menu item “Tools” and compare the feature “Object features > Layer values > Mean > OffGreen\_subset.tif” and “Object features > Layer values > Ratio > OffGreen\_subset.tif”. You will notice that all green areas are represented very well by a high ratio value.

*Red* and *grey* are classified the same way, but in addition a minimum brightness is defined for these two classes. All darker objects are assigned to the class *shadow*.

- Open the class descriptions of the classes *green*, *red*, *grey* and *shadow* and check how they are built.


First, all green objects are classified and all remaining objects are assigned to *not green*. These remaining objects are then further classified as red and not red objects, and so on.

## Classifying the image

After the base classes have been defined, you can run the first classification.

1. Navigate to image object level 2.
2. Choose “without class-related features.”



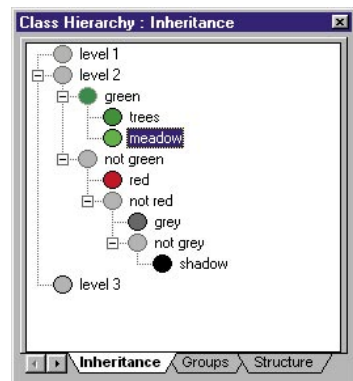
3. Click  to run the classification.
4. Display the classification result without transparency and object outlines.



## Discriminating objects representing trees and grassland

The next step is to assign all objects classified as *green* to either a class called *trees* or a class called *meadow*. First use could be made of the surface model at this point as trees are high standing objects as opposed to meadows. However, the standard deviation of the image layer “OffGreen\_subset.tif” is still a better feature to fulfill the task of discriminating these two classes.

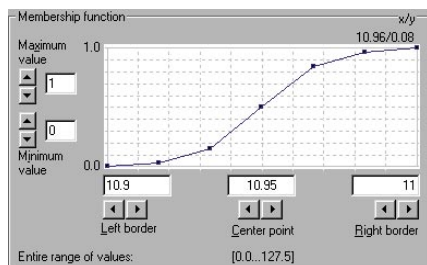
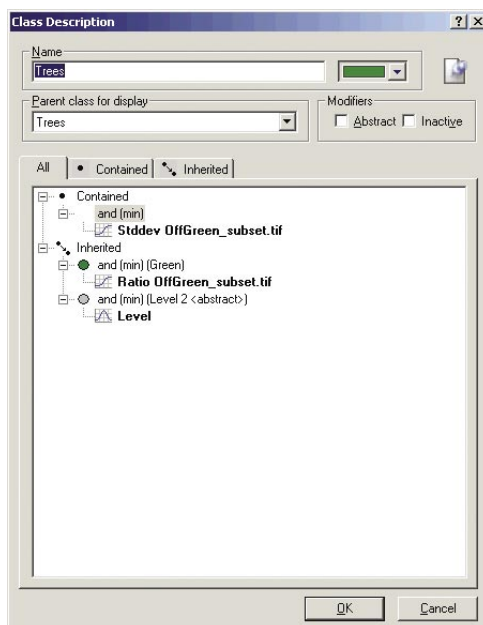
1. Create two new classes and call them *trees* and *meadow* in the “Inheritance” dialog of the “Class Hierarchy” window.
2. Make these two classes child classes of the class *green*.



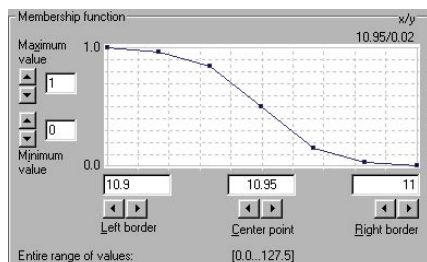
3. Insert the feature “Object features > Layer values > Stdev > OffGreen\_subset.tif” into the class descriptions of *trees* and *meadow*.

A green object represents trees if its standard deviation of layer “OffGreen\_subset.tif” is larger or equal to 11.

4. Edit the membership functions as shown below.




trees: “Stddev OffGreen\_subset.tif”

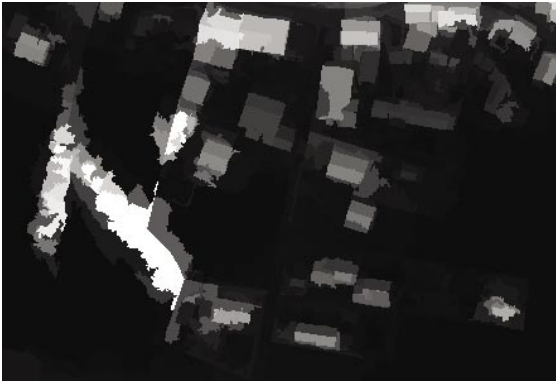


meadow: “Stddev OffGreen\_subset.tif”

## Separating objects representing roofs, yards and roads

The classes *red* and *grey* contain objects that represent roofs, yards and roads. For grey objects in particular, it is hardly possible to distinguish these classes with the use of spectral features like layer mean values or standard deviations. You have to apply the surface model and the fact that roofs are generally higher than roads and yards.

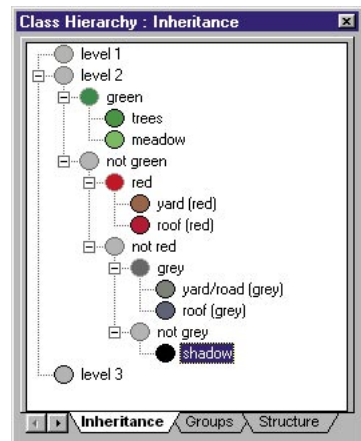
1. Open the “Feature View” (“Tools > Feature View...” or click ) if it is not open yet.
2. Select the feature “Object features > Layer values > to scene > Mean diff. to scene > OffElev\_subset.tif”.



In this view, the elevations of the single objects are displayed. Notice that the houses are quite clearly recognizable. This feature will be used to distinguish between high and low objects.

## Creating the new classes

1. Create four new classes and call them *roof (red)*, *yard (red)*, *roof (grey)* and *yard/road (grey)*.
2. Make the classes *roof (red)* and *yard (red)* child classes of the class *red*.
3. Make the classes *roof (grey)* and *yard/road (grey)* child classes of the class *grey*.

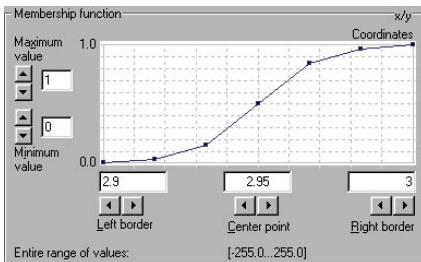
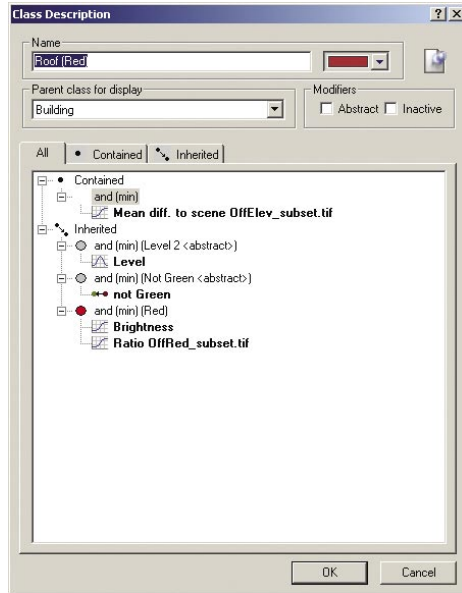




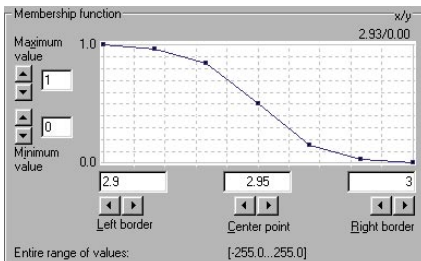
## Inserting the class descriptions for roof (red) and yard (red)

A red object is classified as a red roof if its mean difference in elevation according to the whole scene is larger than 3. Otherwise the object is classified as a red yard.

1. Insert the feature “Object features > Layer values > to scene > Mean diff. to scene > OffElev\_subset.tif” into the class descriptions of *roof (red)* and *yard (red)*.
2. Edit the membership functions of these two classes as shown below.



roof (red): "Mean diff. to scene OffElev\_subset.tif"



yard (red): "Mean diff. to scene OffElev\_subset.tif"

3. Close the class descriptions again.

## Inserting the class descriptions for roof (grey) and yard/road (grey)

The distinction using the elevation was quite simple for the red objects. However, the larger amount of grey objects leads you to consider more possibilities. An additional feature will be introduced for the description of low and high grey objects.

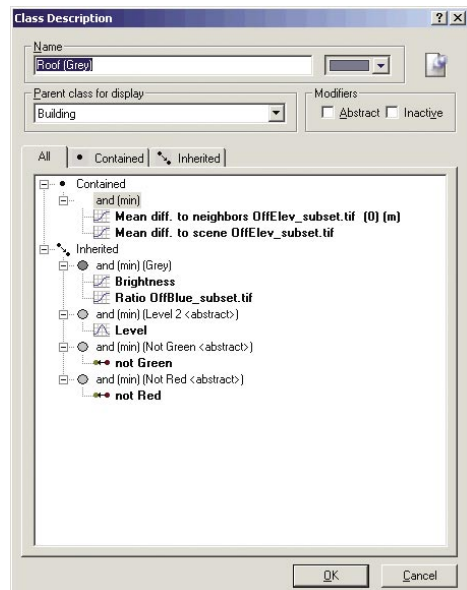
An object is classified as *roof (grey)*

- if its elevation mean difference to direct neighbors is larger than or equal to 0.01 and
- if its elevation mean difference to the entire scene is larger than or equal to 4.

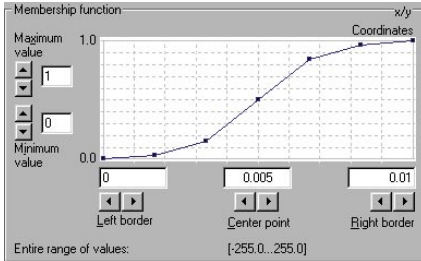
Consequently, a grey object is classified as *yard/road (grey)*

- if its elevation mean difference to direct neighbors is less than 0 or
- if its elevation mean difference to the entire scene is less than 3.9.

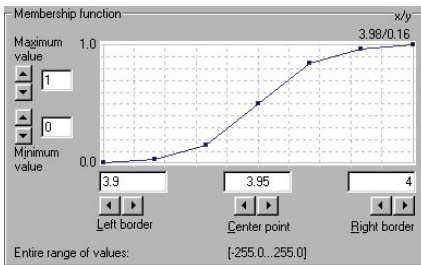
1. Insert the feature “Object features > Layer values > to scene > Mean diff. to scene > OffElev\_subset.tif” and “Object features > Layer values > to neighbors > Mean diff. to neighbors > OffElev\_subset.tif” into the class description of *roof (grey)*.



2. Edit the membership functions as shown below.



roof (grey): "Mean diff. to neighbors OffElev\_subset.tif"



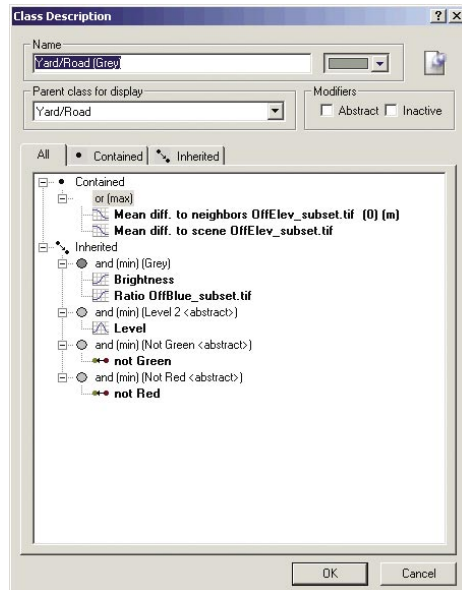
roof (grey): "Mean diff. to scene OffElev\_subset.tif"

3. Close the "Class Description" dialog again.

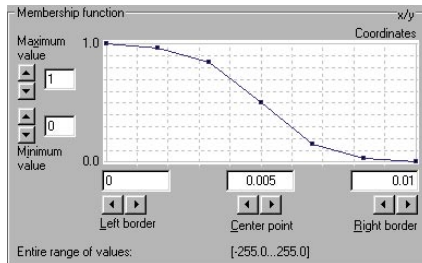
4. Open the "Class Description" dialog of the class *yard/road* (grey).

5. Insert the feature "Object features > Layer values > to scene > Mean diff. to scene > OffElev\_subset.tif" and "Object features > Layer values > to neighbors > Mean diff. to neighbors > OffElev\_subset.tif" into the class description of *yard/road* (grey).

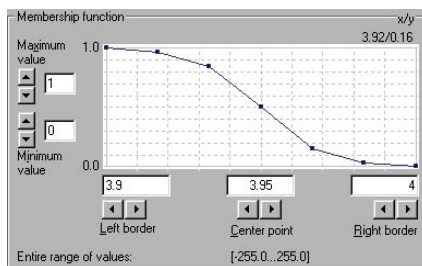
6. Change the logical expression to "or (max)" by right-clicking it and selecting "Edit Expression."



7. Edit the membership functions as shown below.



ard/road (grey): "Mean diff. to neighbors OffElev\_subset.tif"



yard/road (grey): "Mean diff. to scene OffElev\_subset.tif"

8. Close the "Class Description" dialog again.

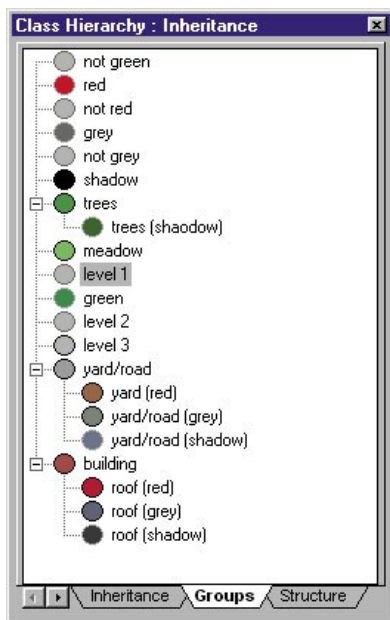
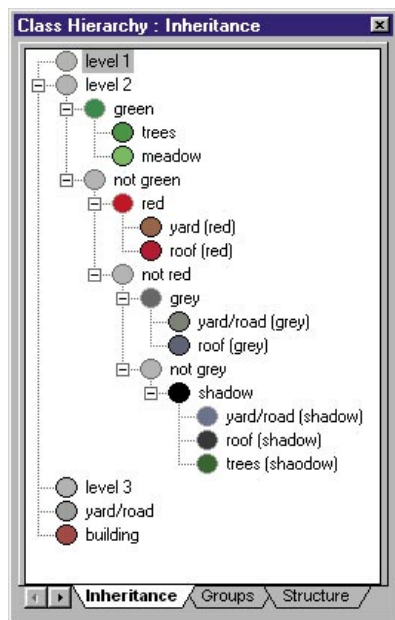
9. Classify the image!

### Loading the final class hierarchy

The next step is to classify the shadow objects. This is done using neighborhood relations: if a shadow object has a high relative border to objects classified as yard or road, it is regarded a yard or a road as well. Since the use of such class-related features has been explained in previous exercises, the final class hierarchy will simply be loaded.

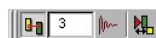
1. Choose the menu item "Classification > Load Class Hierarchy" or right-click in the class hierarchy window to load the class hierarchy.
2. Select the file "as\_level2.dkb" and open it.
3. Have a look at the class descriptions of the child classes of *shadow*.

There are two other new classes beside the new child classes of *shadow*: *yard/road* and *building*, which form semantic groups.




## Classifying the image

1. Enable classification with class-related features.
2. Select image object level 2 as the level to be classified.
3. Set the number of classification cycles to 3.



4. Click .

If your classification result looks different from the one shown here, navigate to the exact level of the semantic groups hierarchy using the green arrows  in the tool bar.

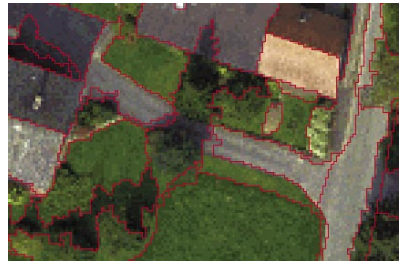
The classification is just about finished. However,



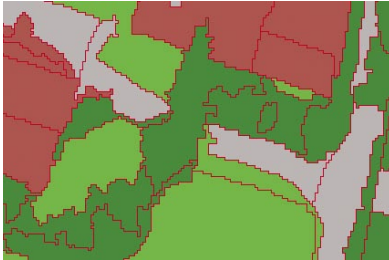
if you look at the borders of the objects classified as buildings, you will notice, that they do not represent meaningful parts of the image just yet, due to the segmentation. A technique for correcting this inconsistency is introduced in the next step.

5. Display the classification result in pixel mode with outlines.

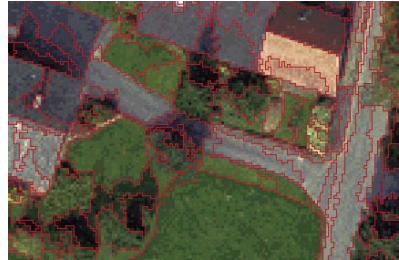
### Performing border optimization



This is the image object in question in a magnified view



This is the classification of the magnified view



This is the same subset on image object level 1

As explained in the theoretical parts of the user guide, multiresolution segmentation yields image object primitives. Shape and size are determined by heterogeneity criteria respective to their color and form. These objects are not meaningful objects. Have a look at the image objects classified as *trees* on image object level 2 for a good example.

If the aerial scan is segmented with a lower scale parameter, i.e., with higher resolution, this special subset can be displayed better. On the other hand there are a lot more objects than necessary, e.g., for the representation of roofs. There is the general problem of different aspects in the same image being best represented at different scales of resolution.

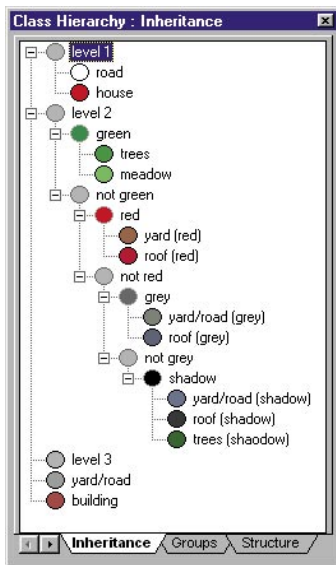
To change such image object primitives into more meaningful image objects, eCognition provides a tool called classification-based segmentation. The knowledge base is

provided by an existing classification of the image. One aspect of classification-based segmentation, classification-based fusion of image objects, was already demonstrated in the exercise [Analysis of the Degree of Urban Impervious Surface](#). Another aspect, so-called border optimization, is shown here.

For border optimization, classified sub-objects are needed. Introducing a complete set of rules for the classification of image object level 1 would go too far, so border optimization is demonstrated on the selected image object using manual classification.

### Creating new classes

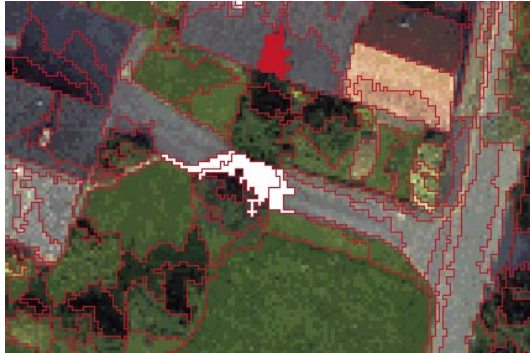
1. Navigate to image object level 1.
2. Open the “Class Hierarchy” dialog and create two new classes called *house* and *road*.
3. Make these new classes child classes of the class *level 1*.



## Classifying sub-objects using manual classification

1. Choose “Manual Classification...” from the “Input Mode” menu.
2. Choose the class *house* in the “Class Hierarchy” window.
3. Assign the shown image objects to the class *house*.

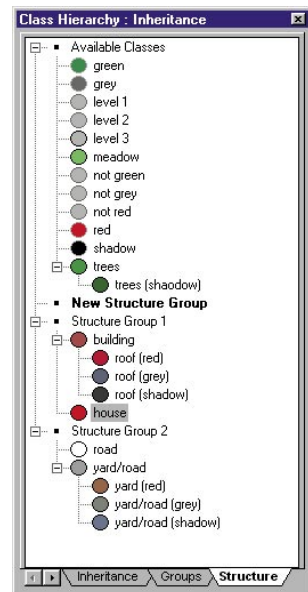
4. Select the class *road* in the “Class Hierarchy” dialog.
5. Assign the shown image objects to the class *road* by clicking them. Deactivate the “Input Manual Classification” in the “Input Mode” menu.



## Editing structure groups


As you may know from the chapter “Concepts & Methods,” classification-based segmentation is based on the definition of structure groups.

1. In the “Class Hierarchy” dialog switch to the “Structure” tab register.
2. Create a structure group consisting of the classes *roof (shadow)*, *roof (grey)*, *roof (red)* and *building* by dragging them onto “New Structure Group.” Add the class *house* to the same structure group by dragging it onto the corresponding structure group.
3. Create a second structure group consisting of the classes *road*, *yard (red)*, *yard/road (grey)* and *yard/road (shadow)* by dragging *yard/road* to “New Structure Group.” Add *road* to the same structure group by dragging it onto the corresponding structure group.





## Performing the actual border optimization

1. Choose “Classification-based Segmentation...” from the “Segmentation” menu or click  in the tool bar.
2. Mark image object level 2 in the field “Level.”
3. Enable the check box “Border optimization of selected level.”
4. Click “OK.”

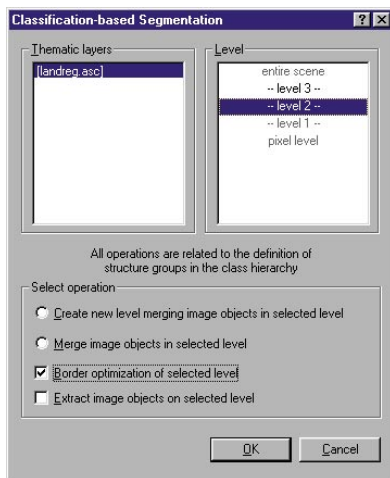


Image object level 2 before border optimization

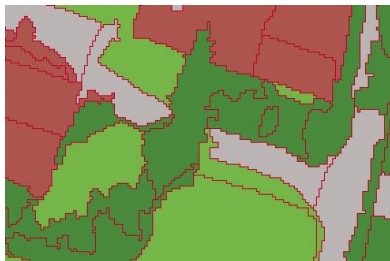
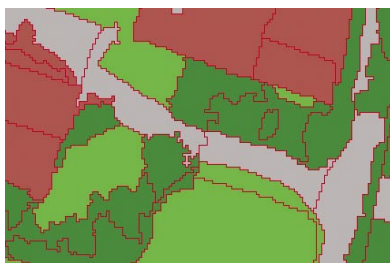


Image object level 2 after border optimization



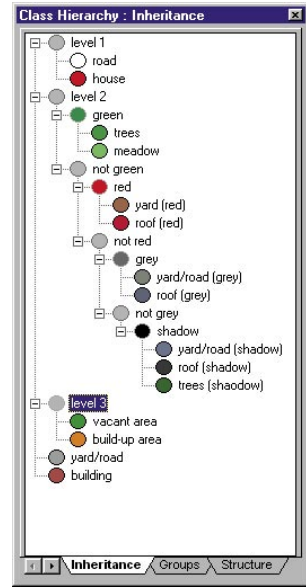
With the help of the classified sub-objects, the image object primitives yielded by multiresolution segmentation have been transformed to more meaningful image objects.

## Classifying built-up and vacant plots of land

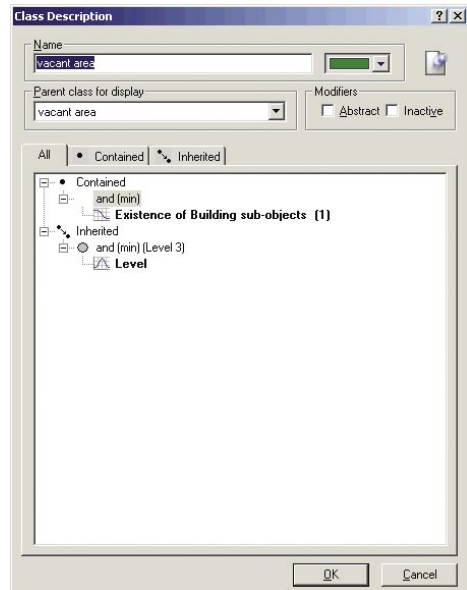
Finally, the imported thematic layer is to be used for analysis. For a start, a classification distinguishing built-up and non-built-up plots on image object level 3 will be performed. Since this level was segmented in such a way that each polygon of the imported thematic layer is now identical with an image object, each image object of this level now represents a plot of land.

1. Switch to the “Class Hierarchy” dialog, create two new classes called *built-up area* and *vacant area* and make them children of *level 3*.

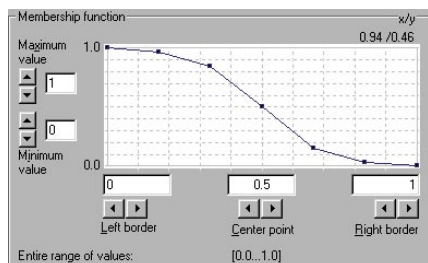
A plot of land is classified as vacant if it does not have any sub-object on image object level 2, which is classified as *building*. eCognition offers the feature “Class-related features > Relations to sub-objects > Existence of > *Building* (1)” to check this.



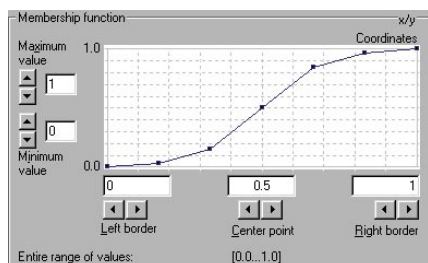
2. Insert the feature “Class-related features > Relations to sub-objects > Existence of > *Building* (1)” into the class description of each of the new classes.



3. Edit the membership functions as shown below.



vacant area: "Existence of building sub-objects"



built-up area: "Existence of building sub-objects"

4. Close the "Class Description" dialog again.

The following classification can only be executed successfully if image object level 2 has been classified. If this is not the case, you can make up for it now.

5. Enable classification with class-related features. One classification cycle is enough this time around.

6. Select image object level 3.

7. Click .

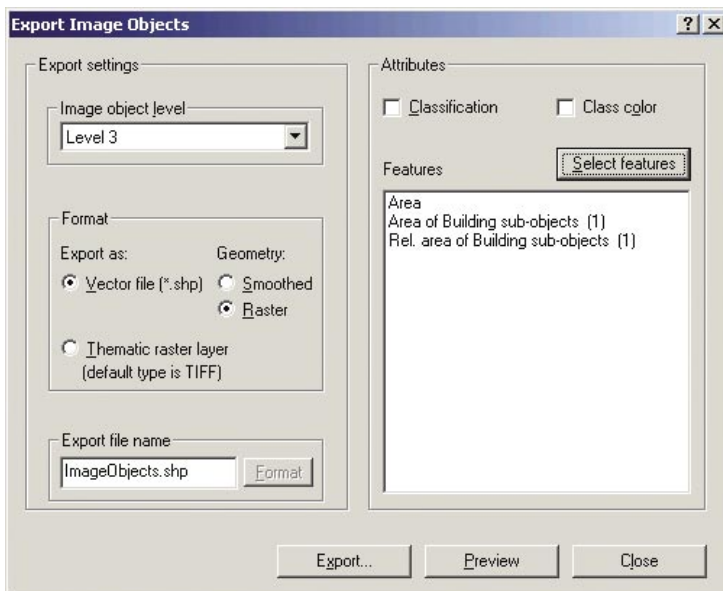
The classification result now shows the built-up and vacant properties.



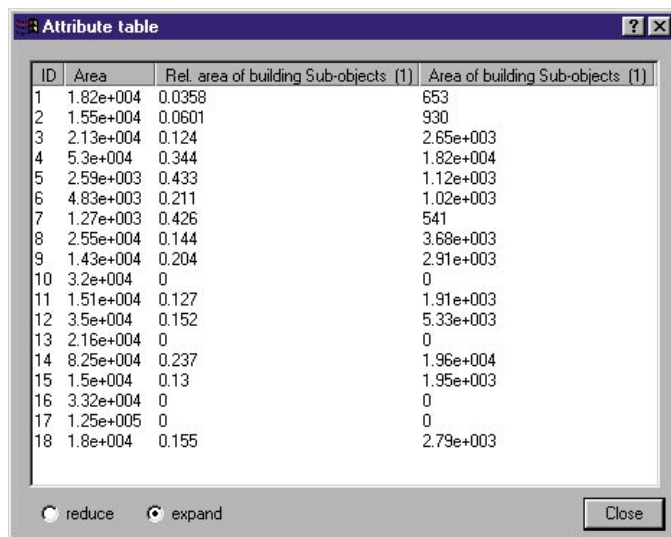
### Analyzing the area covered by buildings for each property

Another way of using the information given by the thematic layer is to find out how much area of a plot of land is covered by buildings.

1. From the “Export” menu choose “Image Objects....”
2. As image object level, select level 3.
3. Under “Attributes” deactivate “Classification” and “Color.”
4. Select the following features as attributes:
  - “Object features > Shape > Area” to calculate the area of the properties,
  - “Class-related features > Relations to sub-objects > Rel. area of > *building* (1)” to calculate the share of area in a plot of land covered by buildings,
  - “Class-related features > Relations to sub-objects > Area of > *building* (1)” to calculate the absolute area in a plot of land covered by buildings.
5. Choose the format you want to export the data in. You can select between thematic raster layer, smoothed vector file or vector file with raster geometry.



6. Click “Preview” to view the attribute table.



ID	Area	Rel. area of building Sub-objects (1)	Area of building Sub-objects (1)
1	1.82e+004	0.0358	653
2	1.55e+004	0.0601	930
3	2.13e+004	0.124	2.65e+003
4	5.3e+004	0.344	1.82e+004
5	2.59e+003	0.433	1.12e+003
6	4.83e+003	0.211	1.02e+003
7	1.27e+003	0.426	541
8	2.55e+004	0.144	3.68e+003
9	1.43e+004	0.204	2.91e+003
10	3.2e+004	0	0
11	1.51e+004	0.127	1.91e+003
12	3.5e+004	0.152	5.33e+003
13	2.16e+004	0	0
14	8.25e+004	0.237	1.96e+004
15	1.5e+004	0.13	1.95e+003
16	3.32e+004	0	0
17	1.25e+005	0	0
18	1.8e+004	0.155	2.79e+003

Attributes are displayed here for each object chosen.

7. To export the objects as a thematic layer with their attribute table, click “Export.”

As a condition of the protocol function it is not possible to choose the name and the folder for the export file. Those of the other data are always used.

## Summary

In this exercise you

- used eCognition for classifying a high-resolution image,
- imported a thematic layer and used it during the segmentation process,
- discriminated classes using a surface model,
- changed image object primitives into more meaningful image objects using classification-based border optimization,
- performed a thematic analysis using the land register imported as a thematic layer.



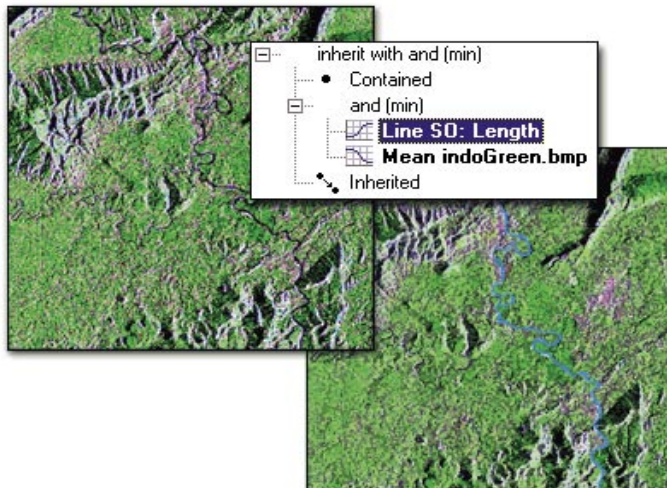
## Tour 4: Radar image of a tropical rain forest in Kalimantan (Indonesia)

This exercise gives you a short example of how line features based on sub-objects can support the classification process.

A three-channel radar image from a tropical rain forest in Kalimantan (Indonesia) is used in this example. The blue and red channels have been filtered for speckle reduction. The highly textured version of the green channel was maintained.

In this exercise you will learn how to:

- generate sub-objects for line analysis,
- insert new classes into a class hierarchy,
- use line features based on these sub-objects to classify their super-objects.



Data courtesy of RSS GmbH/München, Germany



## Load and display the raster data

1. Start eCognition, choose “Project > New...” and change to the directory “...\data\kalimantan\.”

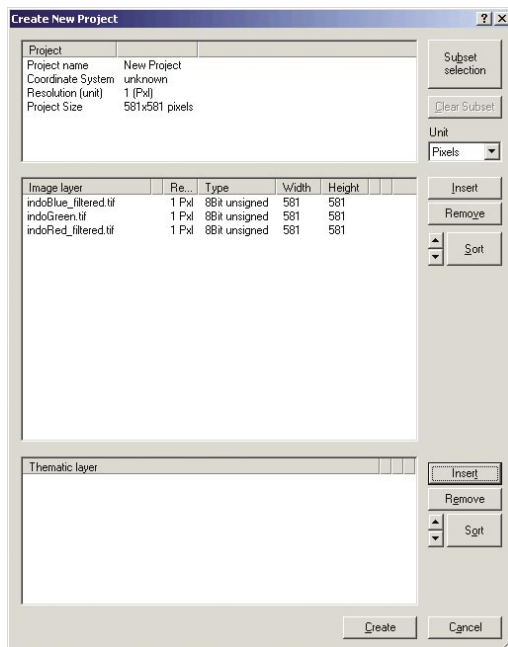
2. Select the image layers


- indoBlue\_filtered.tif
- indoGreen.tif
- indoRed\_filtered.tif

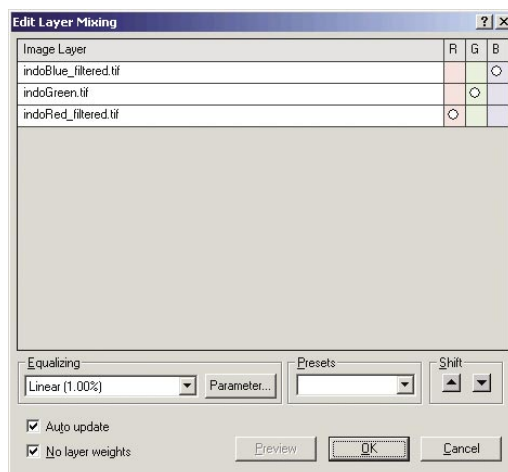
and open them.

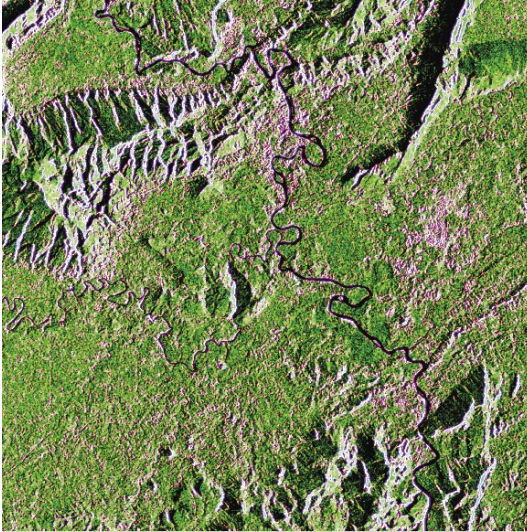
3. If necessary, change the order of the image layers according to their sequence in the spectrum.

4. Click “Create.”




5. Adjust the layer mixing in the “Edit Layer Mixing...” dialog as shown here. To open the dialog choose “Layer Mixing...” from the “View” menu, click  or double-click in the right column of the “View Settings” dialog.

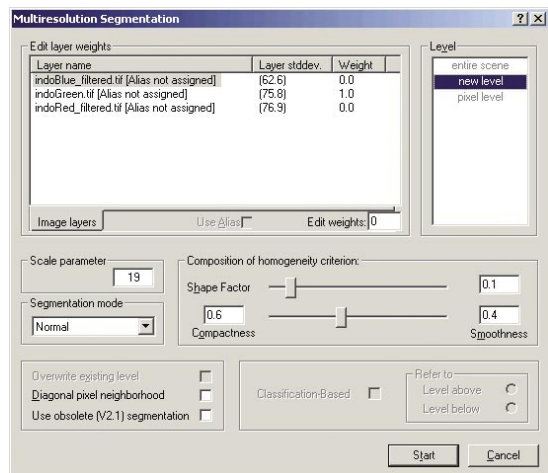




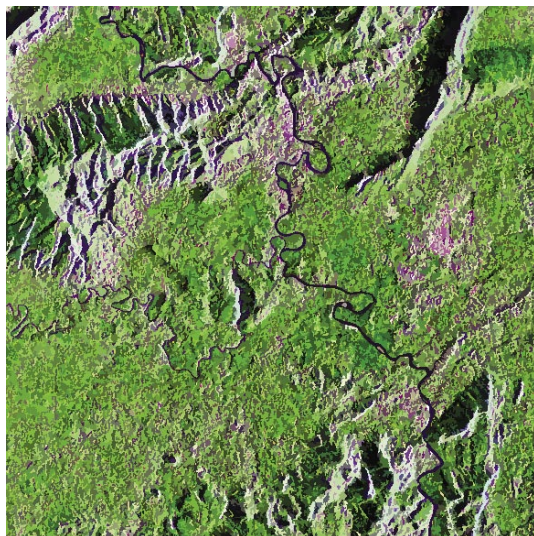
The image shows a dense tropical rain forest with local clearcuts (pink) in a mountainous region (highly reflecting areas appear very bright, radar shadows appear dark) with one major and several smaller rivers running through the scene. In this example the focus lies on the classification of the major river.

## Create image objects


1. Select “Multiresolution Segmentation...” from the “Segmentation” menu or click  in the tool bar.
2. Since the blue and the red channels have been filtered, only the green channel will be used for segmentation (green channel: weight 1; rest: weight 0). Therefore, edit the segmentation parameters as shown below.
3. Click “Start.”



4. The resulting segmented image (in object mean mode) is shown below.



### Obtaining a general overview


1. Select “Feature View...” from the “Tools” menu, the “Toolbars & Dialogs” menu or click  in the tool bar, if it is not open yet.
2. Select the feature “Object features > Layer values > Mean > indo-Green.tif.”

This feature is obviously very useful for at least distinguishing the major river from forest and clearcuts.

3. Move the mouse over the river objects. You will notice layer values well below 3.5.



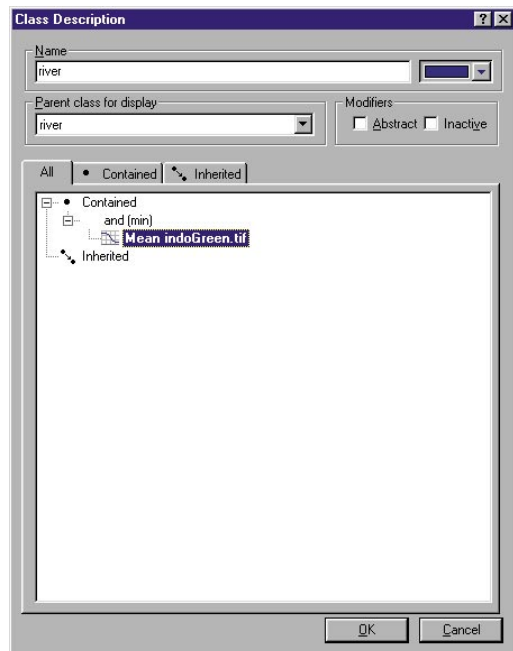
### Creating the class *river*

1. Open the “Class Hierarchy” dialog (menu item “Classification > Open Class Hierarchy...” or click  in the tool bar).
2. Right-click and choose “Insert Class” to create the new class *river*.

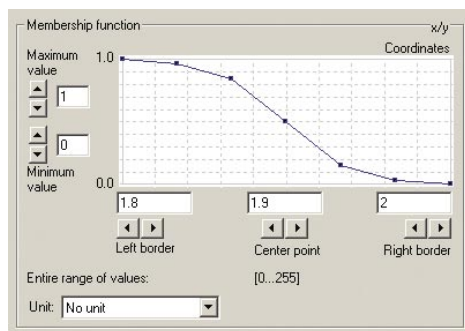


### Building a first class description for *river*

1. Open the class description dialog of the class *river* by double-clicking this class or right-clicking on the class and choosing “edit class.”
2. Insert the feature “Layer Values > Mean > *indoGreen.tif*” into the class description.




### 3. Edit the pertinent membership function.



river: "Mean indoGreen.tif"

## Classifying the image

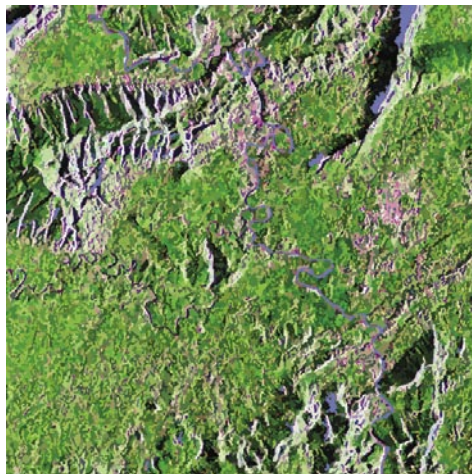
1. Choose "without class-related features" in the tool bar.
2. Click  in the tool bar to start the classification.



## Assessing the classification result


As you can see, the large river has been entirely classified. Some objects representing radar shadows have also been assigned to the class *river*. In this example, it seems that spectral features alone cannot successfully distinguish the large river from radar shadows.

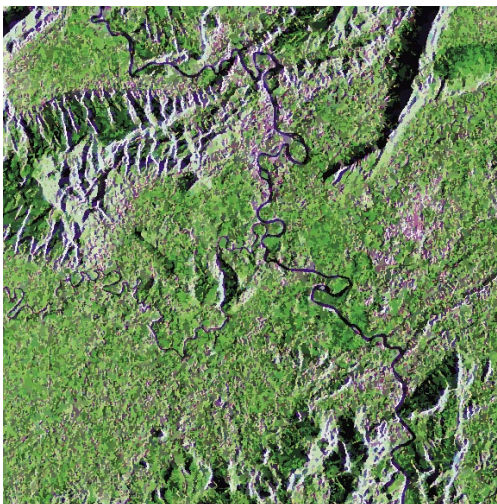
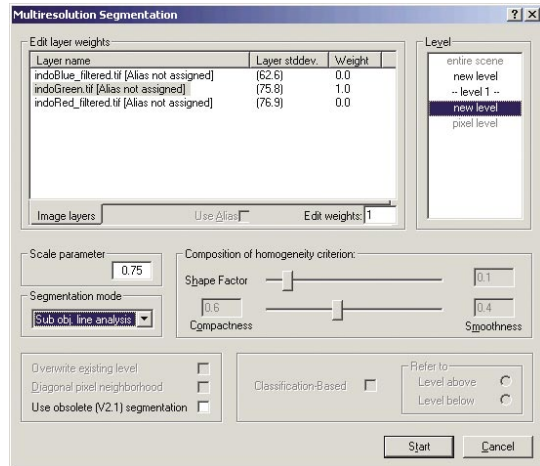
Instead, the qualities of lengthy objects such as rivers will be used with the help of a sub-object line analysis. As a first step, this involves a further segmentation especially suitable for linear objects. The so-called sub-object line analysis segmentation mode will be used to create sub-objects that can be used to apply specific line features based on sub-objects for the class description.





## Sub-object line analysis segmentation

1. Select “Multiresolution Segmentation...” from the “Segmentation” menu or click  in the tool bar.
2. Select “Sub obj. line analysis” as the segmentation mode. Again, use only the green channel for segmentation and set the other layer weights to 0 if necessary. Select **0.75** as “Scale Parameter.” Create the sub-objects on a new, lower level.
3. Click “Start.”





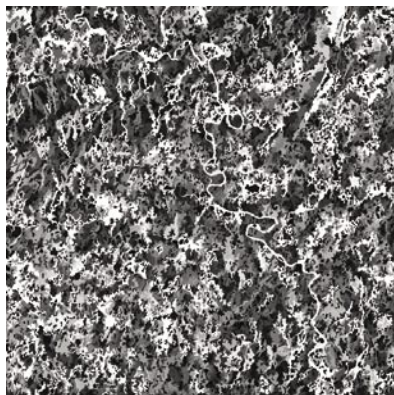
Have a look at the result and notice the new image objects representing the river. The form of their super-objects was regarded during the segmentation process.

## Using line features based on sub-objects

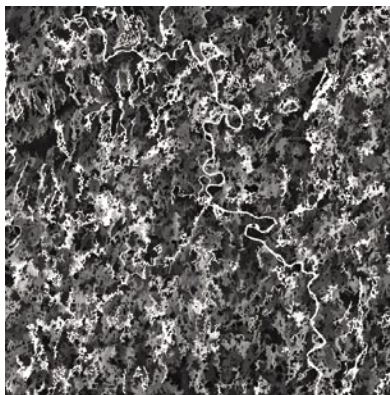
By means of these line-oriented sub-objects it is possible to describe their super-objects by special line features such as width, length and curvature.

Have a look at these features.

1. Navigate to image object level 2. 
2. Select “Feature View...” from the “Tools” menu, the “Toolbars & Dialogs” menu or click , if it is not open yet.
3. Select any or all of the features “Object features > Shape > Line features based on sub-objects” and have a look at the feature qualities for class distinction.
4. Notice that the large river can be distinguished very well from the radar shadow by applying “Length/width (line so)” or “Length (line so).” For both features, the river is emphasized by high layer values. Move your mouse pointer over the image below and you will notice that “Length (line so)” is even better suited than “Length/width (line so).”




“Length (line so)”

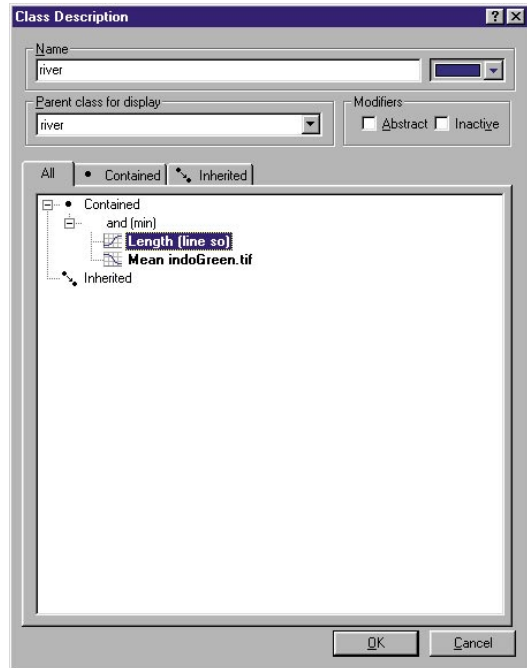


“Length/width (line so)”

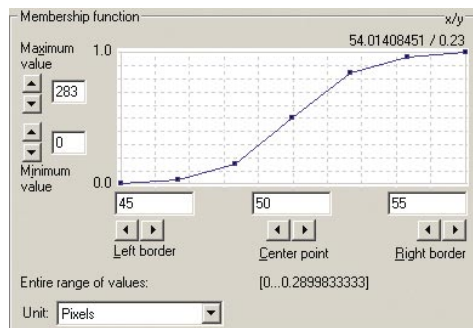
## Completing the class *river*

Use the line feature “Length (line so)” to complete the class description of *river*.

1. Open the “Class Hierarchy” editor (menu item “Classification > Open Class Hierarchy...” or click ).
2. Open the “Class Description” dialog of the class *river* by double-clicking on this class or choosing “Classification > Edit Classes > Edit Class” from the menu bar.
3. Insert the feature “Object features > Shape > Line features based on sub-objects > Length (line so)” into the class description.



4. Edit the appropriate membership function of “Length (line so)” as follows:






## Final classification

1. Change to level 2 in the tool bar.

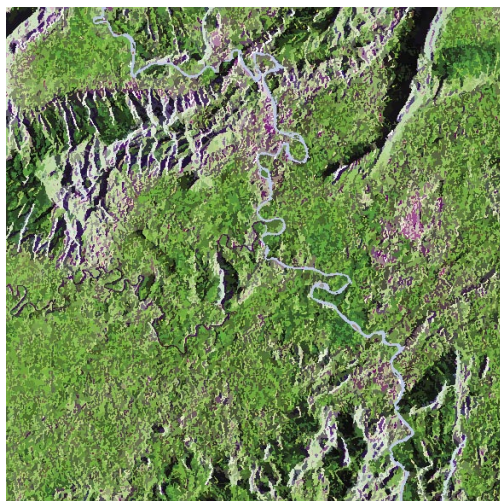


2. Choose “without class-related features.”

3. Click  in the tool bar to start the classification of level 2.



Look at the classification result below: As you can see, the class *river* has been assigned completely and correctly. The classification result has been improved substantially.



The classification of this image will not be continued here. This example serves merely to demonstrate how to classify objects with a specific form, such as rivers, and distinguish them from falsely classified objects.

## Summary

In this exercise we

- created a new class hierarchy,
- applied the sub-object line analysis segmentation mode to create line oriented sub-objects,
- used line features based on these sub-objects to classify their super-objects.

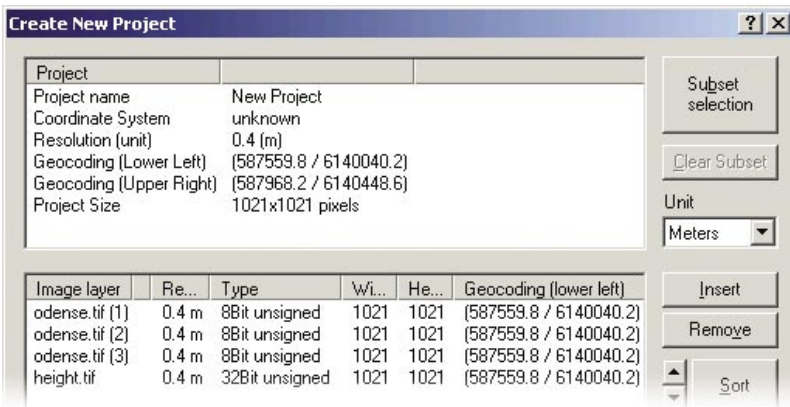
## Tour 5: Aerial Photo and Lidar Surface Model of Odense (Denmark)

This exercise will demonstrate the ability of eCognition to work with a combination of aerial photography and a LIDAR surface model. Based on a simple class hierarchy, high and low lying impervious areas, vegetation and the respective shadows will be separated. Further, the use of customized features to improve a class hierarchy is demonstrated.

### Loading the data

1. Start eCognition and open the import dialog by selecting “Project > New....”


The data to be loaded is an aerial photograph and a LIDAR surface model of the town of Odense in Denmark. Take care that the three layers of the odense.tif files are first in the sequence, so that the class hierarchy which is used addresses the right layer values.

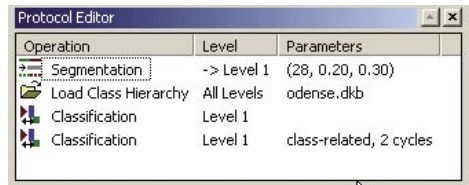


2. After selecting the files and sorting them in the required order, select “Create” to generate the new project.

## Loading a protocol





Since the segmentation should be relatively clear by now and the focus for this guided tour lies on other topics, the first steps are performed by a protocol.

1. Open the protocol editor by selecting “Protocol > Open Protocol Editor.”
2. Right-click the editor and select “Load Protocol.” The protocol is named “odense.dpt.” It contains the segmentation, automatically loads a class hierarchy and performs the classification.
3. To execute the protocol, right-click the protocol editor and select “Execute” .



## Multiple window functionality

Since the view settings may vary depending on the last project which was edited in eCognition, some of the actions described here may not be necessary.

1. To obtain a good overview of the first classification result, open three additional windows, so that four windows are open. Then arrange them horizontally. To link the zoom and pan functionality, select “Window > Link all Windows.”
2. Now each window can be fitted with its own view settings. Open the view settings by selecting “Toolbars and Dialogs > View Settings” or clicking  in the tool bar. Since object outlines are used in the following settings, polygons have to be created first in order to be able to display the outlines. Therefore, select “Polygons > Create Polygons...” from the menu bar or click  in the tool bar. To change the layer mixing, select “View > Layer mixing...” from the menu bar or click  in the tool bar. To change the color of the outlines, open the “Edit Highlight Colors” dialog by selecting “View > Edit Highlight Colors...” or click  in the tool bar.

A good setting for the given example is as follows:

**Window 1**

Mode:	layer
Layer:	image data > set the layer mixing to a three layer RGB mode (“three layer mixing”) for odense.tif
Image data:	pixel
Outlines	off

**Window 2**

Mode:	layer
Layer:	image data > set the layer mixing to a one layer mode (“one layer gray”) for height.tif
Image data:	pixel
Outlines	off

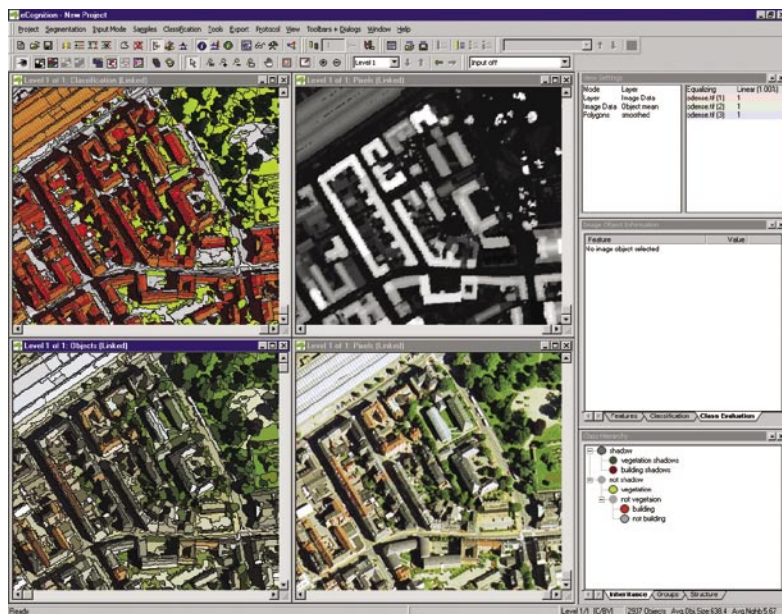
**Window 3**

Mode:	layer
Layer:	image data > odense.tif in a three layer RGB mode (“three layer mixing”)
Image data:	pixel
Outlines	smoothed

**Window 4**

Mode:	classification
Layer:	image data > odense.tif in a three layer RGB mode (“three layer mixing”)
Image data:	pixel
Outlines	smoothed

If you have the “Image Object Information” dialog, the class hierarchy and the view settings open, your screen should look like this:



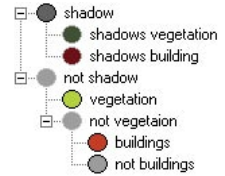
This way an extensive overview of the different input data as well as the classification can be obtained.

Summary of the steps to take in this section:

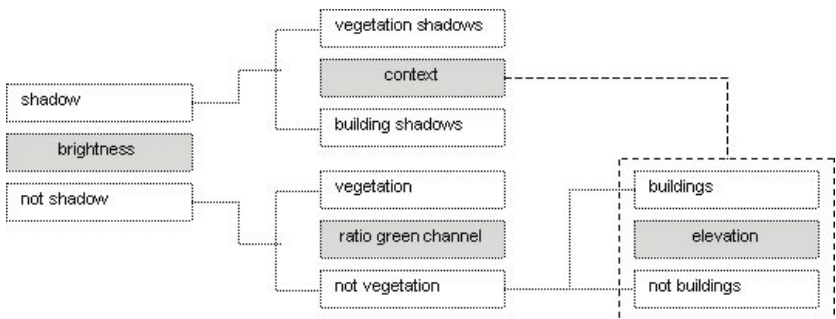
- Open three additional windows
- Arrange the windows horizontally
- Link all windows
- Create polygons
- Open the class hierarchy, the image object information dialog and the view settings
- Edit the view settings as described above

## The class hierarchy

The class hierarchy uses a masking technique. First, shadows are separated from nonshadows. Then the nonshadowed areas are distinguished into vegetation and nonvegetation. The latter is separated into buildings and nonbuildings. Within the shadows a further separation is made between *vegetation shadows* and *building shadows*.



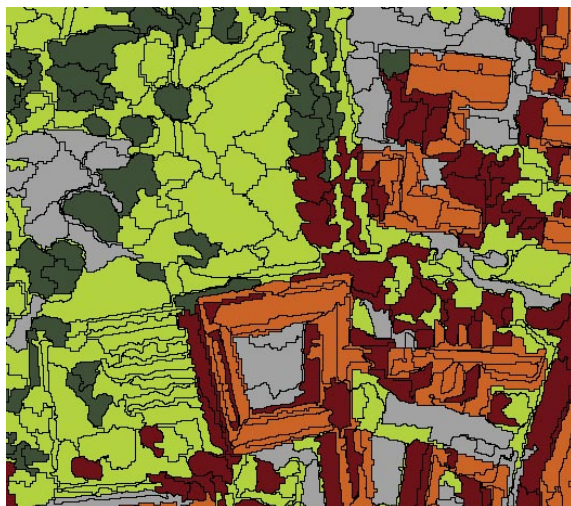
The setup of the class hierarchy and the features used to separate the classes are displayed below.




To get an insight into how the different classes are distinguished, open them in the class hierarchy and take a look at the contained expressions. Alternatively, the image object information dialog can be used to get an overview of the features used.

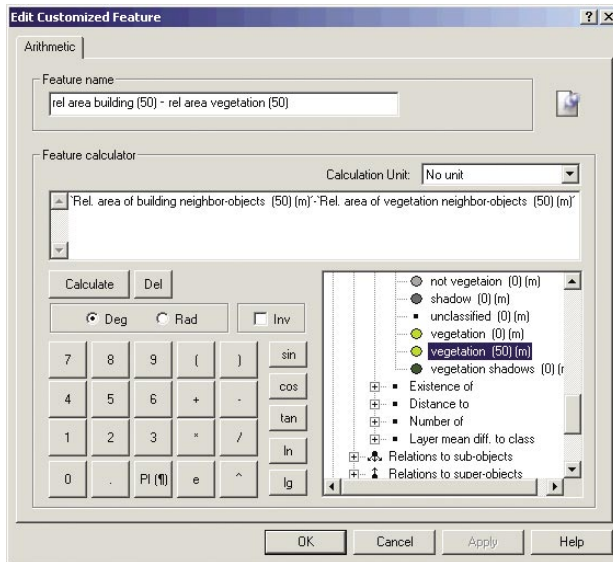
## Defining arithmetic features

The loaded class hierarchy is a clean and simple approach to the task. However in some areas improvements can be made. One such area is the distinction of shadows cast by buildings and the vegetation shadows. In the present form, the relative area of the surrounding objects is used to determine the membership to the classes *building shadows* and *vegetation shadows*. In some places this causes shadows cast by trees to be classified as building shadows due to a high percentage of urban area in the surrounding.



The logic applied at the moment is “if an object classified as shadow is surrounded sufficiently by objects classified as building, it should be classified as *building shadow*.” This rule takes into account only the relative area of neighbor objects classified as building. It would be more accurate however to specify that all objects which are more surrounded by buildings than by vegetation are to be classified as *building shadows*. Therefore, a new feature is created which evaluates both the relative area of buildings and of vegetation. This is done with the help of the “Customize Features” editor.

1. Open the dialog by selecting “Tools > Customize Features...” or using the respective button  from the tool bar.
2. Insert a name into the feature name window. In the example the feature is named “rel area building – rel area vegetation.” This way the feature name already indicates the performed arithmetics.
3. Define the function of the new feature by inserting first the feature “Rel. area of building Neighbor-objects (50)” and then subtracting the feature “Rel. area of vegetation Neighbor-objects (50).” As you can see, the features used have the number 50 attached in brackets. This indicates that the relative area of all objects within the radius of 50 pixels from the respective object is used. When a new project is started, those features exist only with no radius (0) defined. To change this radius, right-click the feature in the feature selection window, choose „Copy Feature“ and edit the number in the opening menu. The feature will be added to the list of features with the new radius defined. It can then be included into the calculation by a double-click.



Summary of the steps to take in this section:

- Open the dialog “Customize Features”
- Name the feature
- Define the radius for the features “Rel. area of building neighbor objects” and “Rel. area of vegetation neighbor objects” as 50.
- Insert the feature “Rel. area of building neighbor objects (50)”
- Insert a minus sign
- Insert the feature “Rel. area of vegetation neighbor objects (50)”

#### Adding the feature to the class building shadow

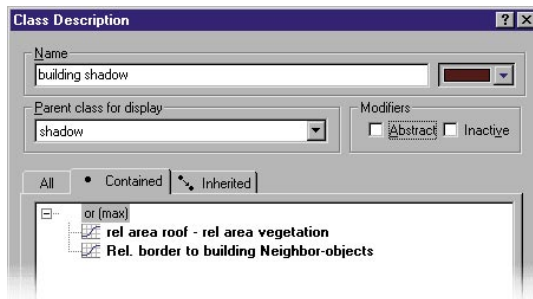
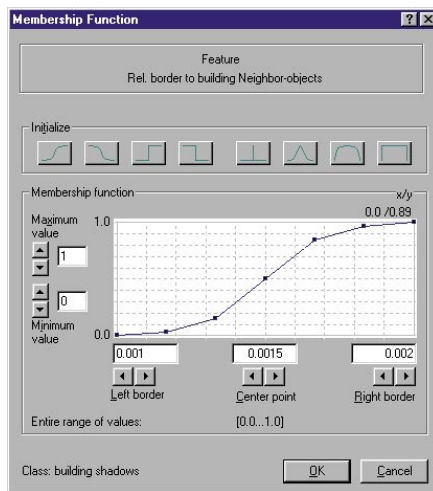
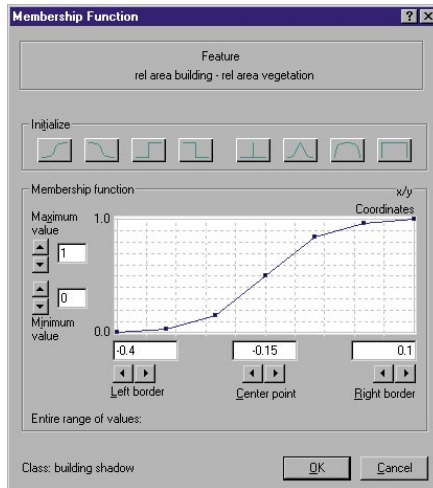
- Now apply the new feature to the class *building shadow*: open the class description, delete the feature “Rel. area of building neighbor objects (120)” by right-clicking it and selecting “delete” and insert the feature “rel area building – rel area vegetation” instead. This feature is listed under “Class-related features > Customized.” The left and right border values can be defined with  $-0.4$  and  $0.1$  respectively.



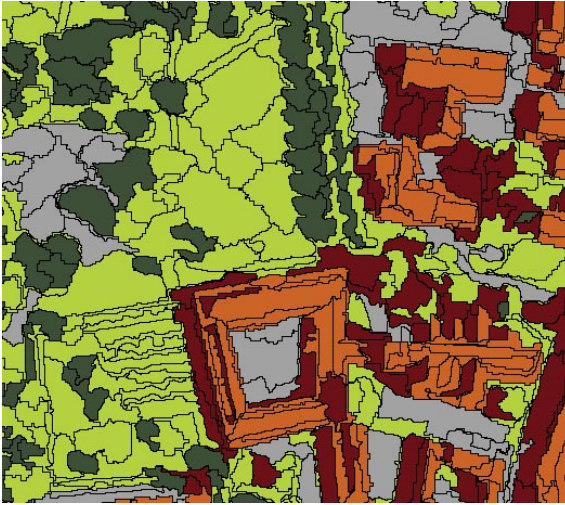
If only this feature is used, some shadows which are highly surrounded by vegetation but nevertheless caused by buildings are misclassified.

This can be avoided by adding the feature “Rel. border to building neighbor objects.” The goal is to have all objects which are either surrounded by a higher portion of buildings than vegetation or all which have a high relative border to buildings classified as *building shadows*. This can be done as follows:

5. Add the feature “Rel. border to building Neighbor-objects” with the border values 0.001 and 0.002 and change the and(min) operator into or(max).



6. If you perform another classification, you will see that the distinction of the shadows was improved.



Summary of the steps to take in this section:


- Open the class *building shadow*
- Insert the newly created feature “rel. area building – rel. area vegetation”
- Define the left and right border values with  $-0.4$  and  $0.1$  respectively
- Insert the feature “Rel. border to building neighbor objects”
- Define the left and right border values with  $0.001$  and  $0.002$  respectively
- Change the and(min) operator into or(max) by right-click and selecting “Edit Expression”

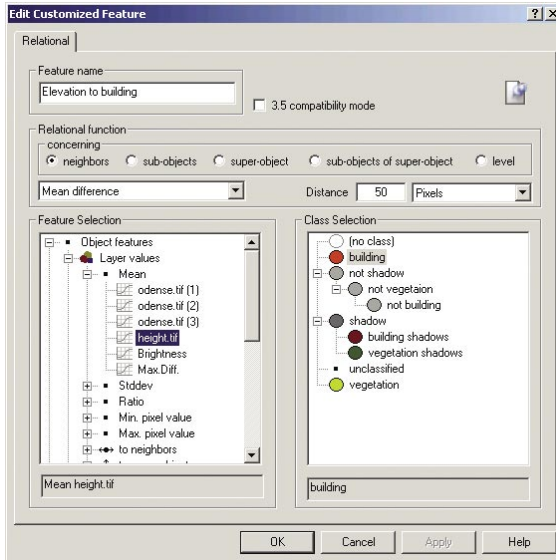
To check whether your results correspond to the description, load the project “odense.dpr”

## Defining relational features

The differentiation of the shadow was improved. Nevertheless there are still further improvements to be made. At the moment all shadows which are cast by buildings are uniformly classified as *building shadows*. Here a further separation into shadows which have the same height as the buildings and shadows which have a lower height is desired. This way shadows which are on the buildings and therefore part of the building, can be identified as such.

The feature required to define whether an object has the same elevation as the surrounding buildings can be easily generated by the “Create Customized Feature” dialog.

1. Open the dialog by selecting “Tools > Customize Features...” or using the respective button from the tool bar . Select “Relational” to get to the definition of relational features. In the relational features register you can create features which compare an object's feature to the features of other objects. You can refer to surrounding objects as well as to sub-objects. Furthermore, you can compare an object's feature to either all surrounding (or sub-) objects or to surrounding (or sub-) objects of a certain class.
2. Define the name of the new feature in the “Feature name” line. The relational function determines what kind of relation is used. In this case the mean difference to neighbor object is needed. Therefore, select “neighbors” and “Mean difference” in the “Relational function” dialog. The distance should be set to 50. This way all objects within a distance of 50 pixels will be included in the calculation.
3. Select the feature “Mean height.tif” from the “Feature Selection” and the class building from the “Class Selection.” Select “OK” to accept the settings for the new feature. This newly defined feature can be explained as follows. For each object the mean difference of its mean value in the height.tif layer is calculated to the mean value in the height.tif layer of all objects of the class building within a radius of 50 pixels. In other words, for each object its difference in height to the surrounding object of the class building is calculated.



Summary of the steps to take in this section:

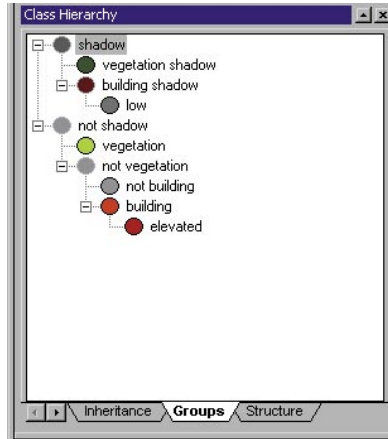
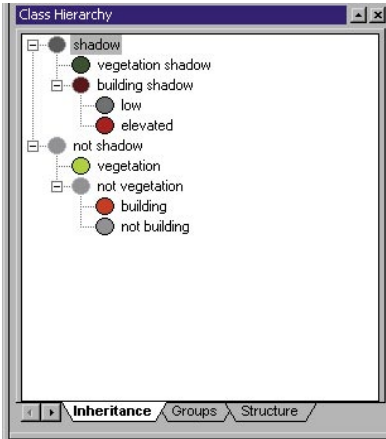
- Open the dialog “Customize Features > Relational”
- Name the feature
- Define the parameters in “Relational function” (“neighbors,” “Mean difference,” “Distance 50”)
- Select the feature “Object features > Mean > height.tif” in the “Feature Selection”
- Select the class *building* in the “Class Selection”

### Adding the new classes to the class hierarchy

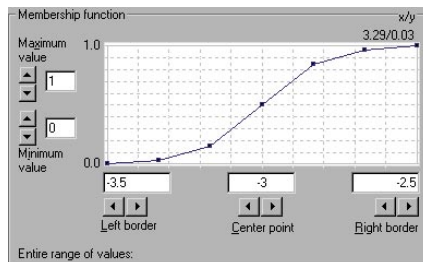
After the new feature is defined, two new classes can be added to further define the class *building shadow*. Those classes could be called *elevated* to describe shadows which are the same height as the buildings, and *low* to describe the shadows besides the buildings.

4. Define those classes as child classes of *building shadow* in the inheritance hierarchy.

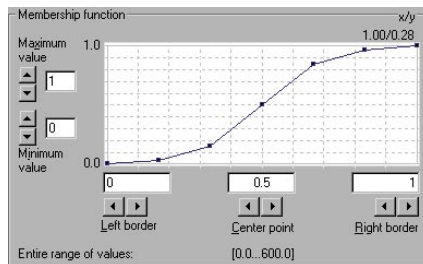
In the groups hierarchy, it makes sense to define the class *elevated* as a child class of *building*. This way, the fact that elevated shadows are a part of the buildings is considered.



5. Insert the newly generated feature into the class *elevated*. The feature range can be determined with  $-3.5$  and  $-2.5$  as left and right border values with an ascending slope.



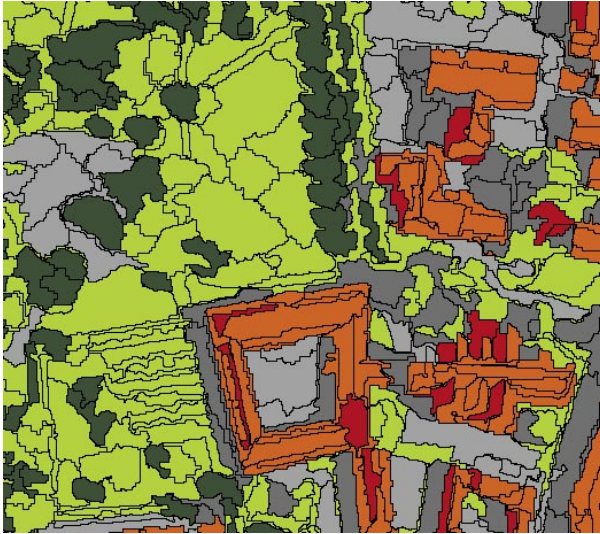
6. To make sure only shadows which are part of buildings are covered by this class and not also shadows on treetops, for example, the additional feature "Border to building neighbor objects" has to be added. The left and right border values can be determined as 0 and 1 with an ascending slope.



The class *low* can be defined with an inverted similarity to the class *elevated*.

If you then perform another classification with class-related features, the differentiation of shadows is further improved.

Summary of the steps to take in this section:

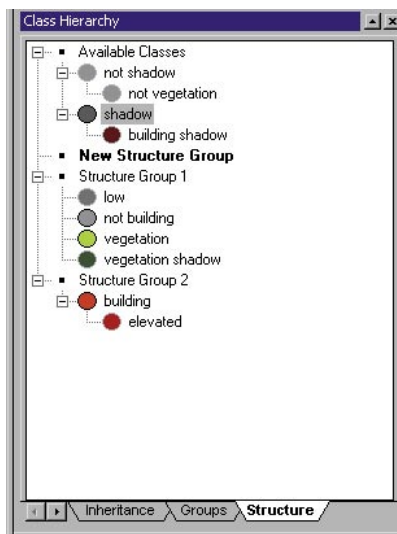



- Add two new classes, *elevated* and *low*
- Define those classes as child classes of *building shadow* in the inheritance hierarchy
- Define the class *elevated* as child class of *building* in the groups hierarchy.
- Insert the newly created feature into the class *elevated*
- Define the left and right border values with  $-3.5$  and  $-2.5$  respectively
- Insert the feature “Border to building neighbor objects” into the class *elevated*
- Define the left and right border values with  $0$  and  $1$  respectively
- Insert an inverted similarity to the class *elevated* into the class *low*
- Perform another classification with class-related features

## Performing a classification-based segmentation

The last step will be to create a new level of objects based on the achieved classification results. The goal is to merge all objects which are buildings as well as all those which represent other classes. To define which objects are merged, structure groups have to be defined.

1. Switch to “Structure” in the class hierarchy and take a look at the settings. As you can see, the classes have already been arranged in structure groups. Objects of classes which are in one structure group will be treated similarly in a subsequent classification-based segmentation. This means that if a new level is generated using classification-based segmentation, objects classified as classes which belong to one structure group will be merged. Currently, objects which do not represent buildings are gathered in structure group 1 and objects which represent buildings are gathered in structure group 2.



2. Now add the class *elevated* to structure group 2 and the class *low* to structure group 1.
3. Now select “Segmentation > Classification-based Segmentation”  from the menu and select “Create new level merging objects in selected level.” The new level is now created. This new level contains objects which have been generated based on the classification. Those objects have not been classified yet, therefore, they do not appear in class colors.

To classify the new level, different classes are required, since the objects and therefore also their features have changed considerably.

4. To facilitate the process, simply load a class hierarchy which is created for a two level object hierarchy and contains the suited classes. Load the class hierarchy “denmark\_2level.dkb” and perform a classification of level 2 with class-related features.



Summary of the steps to take in this section:

- Switch to “Structure” in the class hierarchy to view the structure settings
- Add the class *elevated* to the structure group 2
- Add the class *low* to the structure group 1
- Open the “Classification-based Segmentation” dialog and select “Create new level merging objects in selected level”
- Load the class “denmark\_2level.dkb”
- Classify first level1 and then level2



## Summary

In this exercise the following topics were discussed:

- Loading and displaying image data
- Multiwindow functionality
- Defining and using arithmetic features
- Defining and using relational features
- Performing a classification-based segmentation

Congratulations! You have finished the fifth and final guided tour of the eCognition user guide. If you had difficulties while working through any of them, do not hesitate to try again. The guided tours cover substantial features of the eCognition software for handling and classifying images. The different approaches to image analysis problems always come in handy, so repeating them in order to refresh your memory or to master the techniques is recommended.



[www.definiens-imaging.com](http://www.definiens-imaging.com)



Definiens Imaging GmbH  
Trappentreustrasse 1  
80339 München  
Germany

Tel. +49-89-23 11 80-0  
Fax +49-89-23 11 80-90  
eMail: [ecognition@definiens-imaging.com](mailto:ecognition@definiens-imaging.com)