Stellingen

behorende bij het proefschrift

Document interpretation

applied to public-utility maps

Johannes Gerardus Maria SCHAEMAKER

Delft, juni 1999
1. De combinatie van voorkennis en mathematische morfologie breekt de traditie van heuristiek in de beeldverwerking. *(Hoofdstukken 2, 3 en 4, dit proefschrift)*

2. Niet elk interpretatieconflict is een conflict in het betreffende applicatiedomein. *(Hoofdstuk 5, dit proefschrift)*

3. Het is beter om meerdere detectoren te combineren dan om er één te optimaliseren. *(Hoofdstuk 5, dit proefschrift)*

4. De Gaussische verdeling is vaak de meest ‘normale’ verdeling. *(alle proefschriften)*

5. Het zal eenvoudiger zijn emotioneel-intelligente computers te vervaardigen dan te motiveren.

6. De ulieme logica is een logica van betekenis.

7. Het kleinste natuurkundige deeltje metamorfoseert tot het opdelingsproces zelf.

8. Uit het begrip natuurkundige wet spreekt een menselijke arrogantie zo groot als het heelal.

9. Het gebruik van citaten uit traktaten van filosofen uit de oudheid in hedendaagse wetenschappelijke werken laat naast eruditie ook het onvermogen van de auteur zien om zijn werk te relativeren.

10. Het argument dat vloeken is aangeleerd is geen reden om niet te vloeken, potjandorie!

11. Sportschoenen zijn tegenwoordig net sportauto’s: snel, mooi en met veel spoilers, maar de meeste gebruikers gaan er nooit hard mee.

12. Zen is onZen.
1. The combination of a priori knowledge and mathematical morphology breaks the tradition of heuristics in image processing.
   (Chapters 2, 3, and 4, this thesis)

2. Not every interpretation conflict is a conflict in the application domain.
   (Chapter 5, this thesis)

3. Combining multiple detectors is better than optimizing one single detector.
   (Chapter 5, this thesis)

4. The Gaussian density is often the most 'normal' density.
   (all theses)

5. It will be easier to create emotionally intelligent computers than to motivate them.

6. The ultimate logic is a logic of meaning.

7. The smallest physical particle metamorphoses up to the division process itself.

8. The notion of law of physics reveals a human arrogance the size of the universe itself.

9. The use of quotes from ancient philosophical treatises in current scientific works shows besides erudition of the author also his incapacity to see his work in the proper perspective.

10. That cursing is a talent that is acquired, as is argued, is not a reason to stop doing it, by jingo!

11. Nowadays, sports shoes are just like sports cars: fast, stylish, and fitted with lots of spoilers, yet most of their owners never take them to top speed.

12. Zen is non-Zen.
Document Interpretation

applied to public-utility maps

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof.ir. K.F. Wakker,
in het openbaar te verdedigen ten overstaan van een commissie,
door het College voor Promoties aangewezen,
op vrijdag 18 juni 1999 te 10.30 uur

door

Johannes Gerardus Maria SCHAVEMAKER

doctorandus in de informatica,
geboren te Nibbixwoud
Dit proefschrift is goedgekeurd door de promotor:
Prof.dr.ir. E. Backer

Samenstelling promotiecommissie:
Rector Magnificus, voorzitter
Prof.dr.ir. E. Backer, Technische Universiteit Delft, promotor
Dr.ir. J.J. Gerbrants, Technische Universiteit Delft, toegevoegd promotor
Prof.dr. J. Kittler, University of Surrey, United Kingdom
Prof.dr.ir. A.W.M. Smeulders, Universiteit van Amsterdam
Prof.dr.ir. M.J.M. Bogaerts, Technische Universiteit Delft
Prof.dr. H. Koppelaar, Technische Universiteit Delft
Prof.dr. I.T. Young, Technische Universiteit Delft

Dr.ir. M.J.T. Reinders heeft als begeleider in belangrijke mate aan de totstandkoming van het proefschrift bijgedragen.

Het promotieonderzoek heeft gedeeltelijk plaatsgevonden in het TopSpin-PNEM project Kennisgestuurd conversie van beheerkaarten van leidingnetten voor de Noord-Brabantse Energie Maatschappij.

This work was carried out in the graduate school ASCI. ASCI dissertation series number 41.

CIP-DATA KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Schavemaker, Johannes Gerardus Maria

Document Interpretation applied to public-utility maps / J.G.M. Schavemaker
Delft : Technische Universiteit Delft,
Faculteit der Informatie Technologie and Systemen
Subfaculteit der Elektrotechniek. - Ill.
Thesis Technische Universiteit Delft. - With ref. - With summary in Dutch.
ISBN 90-9012771-2
Subject headings: document interpretation / image processing.

Copyright © 1999 by J.G.M. Schavemaker

All rights reserved. No part of this thesis may be reproduced or transmitted in any form or by any means, electronic, mechanical, photocopying, any information storage and retrieval system, or otherwise, without written permission from the copyright owner.
Documentinterpretatie

toegepast op beheerkaarten van leidingnetten van nutsbedrijven
# Contents

## 1 Introduction
1.1 Introduction ............................................... 2  
1.2 Thesis scope ............................................... 4  
   1.2.1 Problem definition .................................... 4  
   1.2.2 Current conversion techniques ......................... 5  
   1.2.3 Semi-automatic conversion ............................ 6  
1.3 Thesis outline ............................................. 9  
1.4 Scientific contribution .................................... 11  

## 2 Binarization

2.1 Introduction ............................................... 15  
2.2 Introduction to mathematical morphology .................. 16  
2.3 Structuring functions ..................................... 16  
2.4 Image-sharpening operator class ........................... 17  
   2.4.1 Image-sharpening operator class definition ............ 19  
   2.4.2 Laplacian properties for one-dimensional continuous functions 21  
   2.4.3 Laplacian properties for two-dimensional continuous functions 23  
   2.4.4 Sharpening properties for a one-dimensional edge model .......... 24  
   2.4.5 Sharpening properties for a two-dimensional edge model .......... 24  
   2.4.6 Number of iterative applications ........................ 26  
2.5 Image-sharpening operator: partial differential equation 27  
2.6 Discrete sharpening operator ................................ 30  
2.7 Image sharpening with a continuous approximation .......... 30  
2.8 Experiments ............................................... 31  
   2.8.1 Isotropic sharpening .................................... 31  
   2.8.2 Edge sharpening ........................................ 33  
   2.8.3 Document-image sharpening ............................ 35  
2.9 Conclusions ............................................... 38  

## 3 Automatic thresholding with hysteresis
3.1 Introduction ............................................... 39  
3.2 Optimal global thresholding ................................ 40  
3.3 Thresholding with hysteresis ............................... 41
### CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3.1</td>
<td>Introduction</td>
<td>41</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Definition of thresholding with hysteresis</td>
<td>42</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Thresholding with hysteresis: an example</td>
<td>43</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Error calculation for one propagation step</td>
<td>44</td>
</tr>
<tr>
<td>3.3.5</td>
<td>Error calculation for repeated propagation steps</td>
<td>47</td>
</tr>
<tr>
<td>3.3.6</td>
<td>Optimal thresholding with hysteresis</td>
<td>48</td>
</tr>
<tr>
<td>3.4</td>
<td>Automatic thresholding with hysteresis</td>
<td>49</td>
</tr>
<tr>
<td>3.5</td>
<td>Experiments</td>
<td>51</td>
</tr>
<tr>
<td>3.6</td>
<td>Conclusions</td>
<td>52</td>
</tr>
<tr>
<td>Appendix A</td>
<td></td>
<td>54</td>
</tr>
<tr>
<td>Appendix B</td>
<td></td>
<td>55</td>
</tr>
</tbody>
</table>

### 4 Prevention or correction of binarization errors

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>61</td>
</tr>
<tr>
<td>4.2</td>
<td>Pre-processing method</td>
<td>62</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Scanner model</td>
<td>64</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Definition of morphological top-hat transformation</td>
<td>64</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Detection of small background parts</td>
<td>65</td>
</tr>
<tr>
<td>4.2.4</td>
<td>Prevention of binarization errors</td>
<td>67</td>
</tr>
<tr>
<td>4.3</td>
<td>Post-processing method</td>
<td>69</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Generation of topological decompositions</td>
<td>70</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Judgment of topological decompositions</td>
<td>70</td>
</tr>
<tr>
<td>4.3.3</td>
<td>Algorithm for the post-processing method</td>
<td>73</td>
</tr>
<tr>
<td>4.4</td>
<td>Pre-processing method: experiments</td>
<td>75</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Experiments with artificial images</td>
<td>76</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Experiments with images of scanned utility maps</td>
<td>77</td>
</tr>
<tr>
<td>4.5</td>
<td>Post-processing method: experiments</td>
<td>80</td>
</tr>
<tr>
<td>4.6</td>
<td>Conclusions</td>
<td>81</td>
</tr>
</tbody>
</table>

### II Interpretation

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>83</td>
</tr>
<tr>
<td>5.2</td>
<td>Related work</td>
<td>85</td>
</tr>
<tr>
<td>5.3</td>
<td>Interpretation</td>
<td>86</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Knowledge representation</td>
<td>88</td>
</tr>
<tr>
<td>5.3.2</td>
<td>A knowledge-representation example</td>
<td>92</td>
</tr>
<tr>
<td>5.3.3</td>
<td>Interpretation mechanism</td>
<td>93</td>
</tr>
<tr>
<td>5.3.4</td>
<td>Hashing on geometrical attributes</td>
<td>104</td>
</tr>
<tr>
<td>5.4</td>
<td>Detectors</td>
<td>105</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Single-detector experiments</td>
<td>116</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Multiple-detector experiments</td>
<td>117</td>
</tr>
<tr>
<td>5.5</td>
<td>Judgment of complex objects</td>
<td>117</td>
</tr>
<tr>
<td>5.5.1</td>
<td>Judgment of special (geometrical) relationships</td>
<td>119</td>
</tr>
<tr>
<td>5.5.2</td>
<td>Judgment of complex objects</td>
<td>120</td>
</tr>
<tr>
<td>5.6</td>
<td>Interpretation conflicts</td>
<td>121</td>
</tr>
<tr>
<td>5.6.1</td>
<td>Conflict resolution</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td></td>
<td>123</td>
</tr>
<tr>
<td></td>
<td></td>
<td>126</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This chapter sets the scope of this thesis, and provides the reader with a thesis outline and overview. The scope is extracted from a broad view on visual perception and narrowed towards a viewpoint on image interpretation of utility maps. At the end of this chapter, we identify our scientific contributions to the field of document interpretation.
1.1 Introduction

Vision is without doubt the most important sensory input for human beings. Most of our daily information is acquired through visual input, which is evident as a wide variety of visual information carriers exist, such as newspapers, books, magazines, motion pictures, and television. It will not come as a great surprise that the latest addition to these information carriers, the Internet, has a highly visual front end. Working with a computer system connected to the Internet would be considered Draconian, if it was solely equipped with an auditory output device.

Although the importance of the human visual system is clear, the understanding of the workings of the human visual system is still marginal. This is largely due to the fact that the human visual system is the most complex of all our sensory systems. For example, the number of fibers the auditory nerve contains, about 30,000, is quite small when compared to that of the optic nerve, about one million. Most of what we know nowadays about the functional organization of the visual system has been derived from experiments.

The lack of a fully understood visual system makes the development of computer vision not easy. Although one might argue that the lack of an example to copy gives way to unrestricted research into computer-vision techniques, one may also say that it can lead to an exhaustive search along irrelevant research paths. Despite the fact that our knowledge of the human visual system is limited and experimental, a number of computer-vision approaches exist that mimic (parts or abstractions of) the human visual system. Most noteworthy are the scale-space theory on computer front-end vision and the visual perception layers of Marr.

Scale-space theories are based on the fact that the visual system of humans consists of a large number of (Gaussian) image-integration (or convolution) operators which operate simultaneously. The convolution operators apply zero to higher-order derivative kernels to capture the underlying image structure. Because medical images are multi-scaled by nature and contain dense information, most scale-space applications are found in the medical field.

Marr's visual perception model models the three distinct layers or phases of processing the visual system appears to make use of, namely low-level vision, intermediate-level vision, and high-level vision. Man is unique in the sense that she/he can effectively process information from all three phases. As the name already suggests, these processing layers are interdependent in a particular way: the intermediate-level vision depends on low-level vision, and high-level vision on intermediate-level vision. Low-level vision is defined as a form of processing that is completely data driven, so it does not require or use prior knowledge. As such, it corresponds with the above-mentioned front-end vision and scale-space theories, which do not assume anything about the image data processed either. Low-level vision removes redundant information from a stream of images by translating it into a set of relevant differential features, such as edges, movements, etc. In intermediate-level vision, the processing is a mixture between data-driven processing and model-driven processing where prior knowledge is used. For instance, (Gestalt) grouping laws can be applied to form intermediate-level descriptions of geometrical figures based on the principles of continuity. In the high-level vision layer, knowledge of the physical world is used to process the low and intermediate-level data in a top-down fashion.
Hypotheses of physical world objects are verified, and then accepted or rejected by the data.

In the sixties, at the emergence of Artificial Intelligence (AI) and its logic programming languages (PROLOG) a general optimism prevailed on the feasibility of designing a computer-vision system. The idea that just a number of low-level image-processing tools had to be developed to realize computer vision was widespread. The implementation of intermediate and high-level vision was considered a trivial programming exercise in PROLOG. Failing to fulfill that expectation within the promised time, partly due to the limited possibilities of computer hardware available at that time, the researchers of computer vision changed their approach from top-down to bottom-up design, starting with simple pixel operation theory and highly controllable machine-vision applications. As a result of the ongoing research on developing linear and non-linear filter theory, many thresholding and segmentation algorithms and as many vectorization methods, recent computer-vision research nowadays comes close to realizing the originally stated goals resulting from the AI view, i.e. top-down design.

Many of the current successful computer-vision applications are model- or knowledge-based. These applications show how important it is to design a separate image-inference engine, and to specify the knowledge about an application domain. An image-inference engine can be any system capable of reasoning. In addition, there is a basic set of low-level and intermediate-level algorithms to generate hypotheses. Besides fulfilling apparent software engineering goals, the image-inference engine facilitates a jump to designing computer-vision applications having a more abstract level, which focus on specifying all domain knowledge available instead of on the endless tuning of pixel operators.

The concept of model-based or knowledge-based systems is especially useful in the field of document interpretation. The field of document interpretation focuses on designing systems that recognize document content and structure. Documents usually consist of a limited set of characters, digits and (graphical) symbols and are highly structured. Furthermore, documents are still mainly printed in black and white. As such, the knowledge about the visual appearance of characters and symbols and about the structure of the document can easily be captured such that this can be used in a knowledge-based interpretation system.

In this thesis, we present a knowledge-based system for document-image interpretation as mentioned above and, to show its flexibility we describe three applications of the system: utility-map interpretation, utility-map reconstruction and music-score interpretation. Besides the general applicability of the framework, we extend the common knowledge-based approach by means of the notion of conflict resolution. Conflict resolution is the mechanism that identifies interpretation conflicts and tries to resolve them on the basis of judgment of (partial) document interpretations. We argue that conflicts are context-dependent and that the context in which conflicts may occur and the interpretation level at which they have to be resolved must be explicitly specified. In order to resolve interpretation conflicts correctly, we propose a new judgment function of (partial) document interpretations that is based on learning from examples.
1.2 Thesis scope

The scope of this thesis is the knowledge-based interpretation of utility maps. The research on the segmentation and interpretation of utility maps as described in this thesis was part of our collaboration in a Dutch project for semi-automatic conversion of utility maps. In this section, we set the scope of this thesis by stating the problem of map conversion, and identifying our contributions to the solution of utility-map conversion.

1.2.1 Problem definition

Most of the households and companies of the Netherlands are provided with the services of cable television, electricity, gas, and water by public-utility organizations. The network infrastructure of each service modality is drawn on paper maps. These paper maps hold information on location, contents, structure, and topology of the service modality networks. The maps also hold information on the identification, type, and sort of covering of the network components. The location of the network is measured manually; it is derived from the locations of houses, roads, and other topographical entities. The service network is represented by thick lines on the maps. Houses, roads and other topographical entities are represented by thinner lines to distinguish them from the network. Not only the location of the network expressed on the map originates from houses, roads, and other topographical entities, but its length as well, which are drawn as an arrow with an accompanying distance measure. The information provided on the paper map is considered sufficient to identify and locate the service network in the event of failure.

The paper information carrier make efficient management of the service infrastructure difficult. Although now, most of the new maps are digitally drawn within a Geographical Information System (GIS), still a large number of paper maps exist with information relevant to the service networks that is not available in a digital format. The digital storage of utility maps in a GIS environment has many advantages:

Efficient retrieval of information: manual retrieval of any kind of information on paper maps is time consuming. Depending on the information required, the search involves processing by hand a few to thousands of maps. When all maps are stored digitally in a GIS, the information request takes a fraction of the conventional search time.

Efficient maintenance of information: the network information depicted on the maps is not static, and parts of maps may be subject to change. A change or update of information on a part of a paper map involves the physical erasing and redrawing of map contents. Updating is therefore labor-intensive and reduces the quality of the map. Maintaining map information within a digital GIS environment is easy and does not affect the map quality. The technical draftsman can use the GIS functionality to quickly erase and redraw (groups of) map objects.
1.2 Thesis scope

Network capacity modeling: a digital description of a service network makes it possible to model the network and to test the network's capacity or throughput in peak hours.

Although storing utility maps within a GIS environment has many apparent advantages, it comes at a price. First, there is the hardware and software to be purchased to set up a GIS environment. Secondly, the GIS must be tailored to the different network services, by specifying the relational data format. Thirdly, the existing paper maps must be converted to the GIS format. This map-conversion process is the subject of this thesis.

1.2.2 Current conversion techniques

In the early eighties, public-utility organizations began to convert their existing paper maps to a digital format. The information that is captured and converted deals with the cable network. One major problem in all the conversion techniques is the fact that the utility maps suffer from (local) scale, translation, and rotation distortions. These distortions are due to the fact that the technical draftsmen have some freedom in drawing utility maps. They can use an enlargement of a complex part of a map to enlighten its details. Furthermore, they may translate or rotate parts of a map that show sparsely populated areas to reduce the number of utility maps. A common solution to this problem is to use a geometrically correct topographical base map of the area and to relate the utility map to it on the basis of the topography of the base map. The Dutch cadaster provides such base maps in a digital format, which depicts the position of real estates in topographical coordinates. The digital format of the base maps will be referred to as the GBKN or digital topography. The analog topography is the topography shown on the paper utility maps. In all the techniques mentioned below, the analog topography on the utility map must be matched to the digital topography, either by hand or by machine, given some set of control points. Unfortunately, the matching requires an affine transformation of the analog topography to topographical coordinates.

Redrawing on film Redrawing on film results in a plot of the digital topography on stretch-free film. A technical draftsman redraws the topographical position of the network on the film; he starts from the digital topography and then uses the information on the paper utility maps. Note that the network on the paper utility maps is drawn relative to the (distorted) analog topography. The film is then fixed on a digitization table and the network is traced with a digitizer pad. The redrawing-on-film technique is the oldest of the conversion techniques and is very labor-intensive. It is always applicable and often used with maps showing complex situations or sparsely populated areas.

Digital redrawing The digital redrawing technique requires a plot of the digital topography on a computer display as opposed to a topography plot on film as in the redrawing-on-film technique. Using the paper utility maps, a digitizer, and an alphanumerical terminal, the technical draftsman redraws the topographical position of the network directly on the display and keys in the appropriate map-object
codes. This technique combines the redrawing and digitization in one single step. Digital redrawing is faster than redrawing on film technique but its major drawback is that the technical draftsman has to divide its attention among the paper maps, the digitizer, the computer display, and the alphanumerical terminal.

**Manual map digitization** Manual map digitization allows the technical draftsman to digitize the paper map directly with a digitizer. After the digitization, the digitized map must be translated into topographical coordinates to which end a global affine transformation is applied. For the transformation a number of points must be indicated which correspond on the analog and the digital topography; these are determined by manual selection of the technical draftsman. The next step is to apply automatically the (relative) dimensions of the digitized map to shift the network to its final position. Manual map-digitization is less labor-intensive and easier than digital redrawing but is not applicable to all paper utility maps. This technique is about two times as fast as redrawing on film.

**Map digitization** Map digitization is very similar to the above-mentioned manual map-digitization technique. The difference is that the paper map is scanned and shown as a binary plot on a computer display. The digitization is performed within a GIS environment. The advantages of this technique over the manual map-digitization technique is that the technical draftsman can concentrate on one display and can benefit from the additional drawing tools provided by the GIS environment. Map digitization is the fastest of all conversion methods. A major drawback is the price of a sophisticated workstation and the software needed to run the GIS environment.

### 1.2.3 Semi-automatic conversion

The above-mentioned conversion techniques which public-utility organizations use, require a lot of human participation along the complete conversion process. As a result, not only errors can be made, but the techniques become very expensive as well. For these reasons, an automated conversion technique is desired.

**System requirements** Public-utility organizations provide multiple services. Different services are drawn on different types of maps, each of which uses different sets of symbols. Within a single service, different drawing conventions can be observed. Moreover, drawing conventions are not invariant over time; sometimes they have been adjusted to meet new requirements. Consequently, there are a great number of different types of maps which requires an automatic conversion technique that is flexible. The purpose of the conversion ranges from the extraction of one symbol type to the complete interpretation of a drawing or map. A flexible design of a map or drawing-conversion system could be beneficial to a great number of users. The flexibility requirement stems from the multitude of map types and drawing conventions within public-utility organizations, the purpose and nature of the conversion differing among public-utility organizations, and the wide variety in age and quality of the paper maps to be converted. In this thesis, the flexibility is addressed by proposing a knowledge-based system in which the application-domain knowledge
can be adapted to the specific type of maps to be converted. Our knowledge-based approach makes it simpler to specify the application-domain knowledge on map objects, the geometrical relationships between objects, and to establish a hierarchy of objects. Our approach is extended with the possibility to specify control knowledge and knowledge on interpretation conflicts.

To ensure acceptance of the conversion system within an organization, the system must be reliable. It is generally considered more beneficial to design a conversion system that converts, for example, 80% of the map contents (almost) correctly and leaves the remaining 20% to the technical draftsman, than to design a system that converts all map contents and makes many mistakes as a consequence. Conversion mistakes have to be deleted and redrawn by the technical draftsman, whereas non-converted map parts only have to be redrawn, saving conversion time and costs. In this thesis, the reliability of the conversion is enhanced by extending the knowledge-based system with control knowledge and knowledge about interpretation conflicts, and multiple detectors to increase recognition performance. The reliability requirement must be met to ensure acceptance of the conversion system. Besides the requirements of flexibility and reliability, a system must be user-friendly to ensure its acceptance. The operator-friendliness requirement is chosen to make the interaction time(s) between the operator and the conversion process shorter, and the process shorter. At least the operator is needed for starting the automatic conversion process, checking results and possibly deleting, adjusting and appending conversion results. The conversion process flow must be designed in such a way that the number of interactions is minimal and the total interaction time is as short as possible. Additionally, one or two longer interaction periods are preferred to many small interaction periods with delays in between. For this reason, we have chosen to design the conversion process in such a manner that the process requires no cues during operation and all doubtful or rejected conversion results are kept to be presented at the end of the interpretation as well as at the end of the conversion session in a structured way. Summing up, we can identify the following requirements for a (semi-)automatic conversion system:

- flexible
- reliable
- operator-friendly
- conversion-cost reduction

Reduction of costs is the main requirement; it prevails over all the other requirements. Only the first two system requirements, flexibility and reliability, are dealt with in this thesis. The other two requirements are outside the scope of this thesis and remain, apart from the description above, untouched.

System outline In this section, we outline a system for the (semi-)automatic conversion of utility maps. We propose a modular system design. A modular design facilitates the distribution of different system modules over project partners. As a result, project partners have some flexibility in implementing their modules, as
long as they meet the interface standard to other modules. Furthermore, a modular design facilitates easy replacement of modules with newer versions without the alteration of other modules. Additionally, a modular conversion system gives some flexibility to the system operator by splitting the complete conversion process in a number of subprocesses, which can run separately. For example, the operator can scan a set of utility maps, save the results and perform the next conversion steps one-by-one for the whole set of maps. The proposed conversion system consists of a sequence of steps which correspond to the different system modules:

1. scanning of paper utility maps
2. binarization of image data
3. graphical decomposition and vectorization of binary-image data
4. detection of simple map symbols
5. interpretation of complex map symbols
6. manual correction of interpretation results
7. matching interpretation results to digital topography
8. reconstruction: geometrical alignment of interpretation results to digital topography
9. manual correction of conversion results

Scanning is the first logical step in any map-conversion system. To retain flexibility in later image-analysis routines, the maps are scanned (with an ANA Tech Eagle scanner) at 400 dpi and saved as gray-scale images, each pixel having a gray value ranging from 0 to 255. The use of gray-scale images also benefits to the reliability of image-analysis results as more information is preserved than with binary scanning. To increase the flexibility in detecting and interpreting map contents, the gray-scale image is converted to some other modalities in the following two steps. In the second step, the gray-scale image is pre-processed and binarized with a thresholding algorithm. The graphical decomposition breaks down the binary image into its graphical primitives. The vectorization step consists of reducing each graphical primitive to its skeleton, composed of a number of vectors. The gray-scale image, the binary image, the graphical decomposition, and the vectorization are the input of detectors that can recognize simple graphical map objects. The detection step is separated from the interpretation step to meet the demands of a modular system design, as discussed above. Furthermore, detectors are optimized w.r.t. reliability of detection results by incorporating a combination of different techniques within one single detector. The interpretation process interfaces with the detectors. It takes as input some initially detected simple objects and groups them into complex objects by using detectors to discover geometrically related objects. The knowledge-based approach taken in this step ensures flexibility of representation of the application-domain knowledge. The system requirement of reliability is met by introducing a conflict-resolution engine. The results of the interpretation step are corrected by the operator in the following step. Operator-friendliness is achieved by presenting
1.3 Thesis outline

The outline of this thesis follows the flowchart of the utility-map conversion process as mentioned in Section 1.2.3, but has an additional chapter on music-score interpretation serving as an example for the general applicability and flexibility of the developed image interpretation system.

Chapter 2 starts with a description of a class of morphological image operators for sharpening digitized gray-scale images. The image operators are especially useful for sharpening scanned black-and-white documents. We show that image operators using a concave structuring function have sharpening properties. Furthermore, we introduce the partial differential equation that governs this class of image operators. The parameters, i.e. the number of iterative applications and the structuring function size of the image operator, can be determined on the basis of an estimation of the amount of blur present in the image. The amount of blur is expressed as the standard deviation of the Gaussian point-spread function of the scanner’s lens system. For discrete algorithmic implementations of the image-operator class we show that image operators using a parabolic structuring function have an efficient implementation and isotropic sharpening behavior.

In Chapter 3, we introduce a new binarization method for images: automatic global thresholding with hysteresis. The method gives a segmentation into object and background pixels and is an alternative to automatic global thresholding techniques. Thresholding with hysteresis takes three parameters: two global thresholds and a size parameter of the structuring element of the dilation. We present a novel automatic selection procedure for the parameters of the binarization method. The selection procedure assumes that the scanned image contains a mixture of two normal (Gaussian) distributions of object and background gray values.

Chapter 4 deals with the problem of adjoined objects in the binarization of scanned paper documents. In our opinion, the cause of adjoined objects in the binarization can be twofold:

- The objects already adjoin on the paper document.
- The binarization method is erroneous.

Given the twofold nature of the cause, refinement of the binarization method is not sufficient to abandon the problem of adjoined objects. Therefore, we propose two new solutions to the problem. One solution works on the gray-scale image data (pre-processing method), the other solution operates on the binarization result.
(post-processing method). The main difference between them is that while the pre-processing method cannot solve all instances of adjoined objects, the post-processing method can, but it requires a model of the geometrical relationship of the adjoined objects to do so. The choice of method depends on the cause of the problem.

In Chapter 5, we present a utility-map interpretation system. The interpretation system is fitted into a general framework for (image) interpretation. Within the framework, it is possible to represent a variety of knowledge types. The knowledge types range from specification of the application-domain knowledge by specifying objects and geometrical relationships between objects, to specification of procedural knowledge: interpretation order and specification of interpretation conflicts. For each object in an application domain we can specify whether it is a *simple* object and has a specialized detector or whether it is a *complex* object and is composed of a number of objects having a specific geometrical relationship. We propose a new concept which involves *multiple* detectors. Additionally, we can represent interpretation conflicts in our framework by specifying the *context* in which a conflict may appear and when and how it should be resolved. We show the necessity of explicitly specified context-dependent conflict rules and introduce a conflict-resolution scheme based on learning from examples. Within this scheme, we introduce a new judgment function for the judgment of instances of the specified application-domain objects. Our framework for document interpretation is applied in three domains:

1. utility-map interpretation

2. utility-map reconstruction

3. music-score interpretation

The first application of our framework is in the utility-map domain and is presented in Chapter 5. The second application of our framework for document interpretation is given in Chapter 6. Here, the interpretation system is used for the reconstruction of geometrical map objects. The interpretation system provides the necessary information about relationships between map objects for the reconstruction process. The interpretation takes labeled map objects, represented as vectors and points, from a GIS as input and produces their geometrical relationships. By applying the context-dependent conflict-resolution scheme the interpretation system is able to deal with uncertainty in the location of the objects on the map and, as a consequence, it finds the correct geometrical relationships. A third example of an application of our document (image) interpretation framework is presented in Chapter 7. In this chapter we describe a system for music-score interpretation. By tailoring the general interpretation framework, we show that the design of a system for music-score interpretation can be shifted from a complete system design from scratch to a more abstract level of design. At this abstract level of design, the focus is on writing image detectors for simple graphical objects, on specifying music-score structure and on tuning procedural knowledge to ensure an efficient interpretation order. Finally, Chapter 8 discusses our framework and indicate its limitations and disadvantages.
1.4 Scientific contribution

In this section, we list our scientific contribution, as presented in this thesis, to the problem of document interpretation. For each chapter, we pinpoint our new approaches, concepts, implementations, properties, results, techniques, etc.

Chapter 2 contributes to the theory of mathematical morphology as introduced by Serra. In this chapter, we show that the image-sharpening operator, as introduced by Kramer et al. in 1973, is an instance of a class of image-sharpening operators. We prove that all image operators using a concave structuring function have sharpening properties and we derive that these operators have a property in which it is the sign of the Laplacian at a pixel of the image to be sharpened that determines whether that pixel is going to be dilated or eroded by the image-sharpening operator. By using the Laplacian property, we introduce the partial differential equation that governs this class of image-sharpening operators. As discrete implementations of the image-sharpening operator, we show that operators using a parabolic structuring function have an efficient implementation and isotropic sharpening behavior. Finally, we derive an automatic selection of the operator's parameters on the basis of an estimation of the amount of blur present in the image.

The scientific contribution of Chapter 3 lies within the field of thresholding algorithms. We formally define thresholding with hysteresis within the theory of mathematical morphology. We give an error function measuring the number of misclassified pixels for thresholding with hysteresis in the case that the image is a mixture of two normal (Gaussian) distributions of object and background pixels. We outline a novel procedure for automatic thresholding with hysteresis that automatically sets the thresholds by estimating the normal distributions and minimizing the error function.

Chapter 4 proposes two new solutions to the problem of adjoined objects in the binarization result. First, we distinguish two causes of adjoined objects and make a distinction between pre- and post-processing methods found in the literature. After that, we show that pre-processing methods cannot solve the problem of adjoined objects if the objects already adjoin on the paper document. We propose a new general pre-processing method, operating on the gray-scale image data and a new post-processing method, operating on the binary-image data. Our pre-processing method is based on an estimation of the blurring of the lens system of the scanner. Our post-processing method requires a model of the geometrical relationship between the adjoined objects and is based on the topology of the adjoined objects.

Chapters 5, 6, and 7 contribute to the area of document interpretation. Chapter 5 presents a new general framework for document interpretation. We extend the common semantic-network approach to image interpretation with the concept of detectors, judgment functions trained with examples, and conflict resolution. We show that combining detectors using classifier-combining techniques to form multiple detectors, reduces the number of false-positive detections. As an uncertainty calculus, we propose new judgment functions for simple and complex objects, and geometrical relationships. These judgment functions are pattern classifiers that are trained with examples. To deal with interpretation conflicts we introduce a novel conflict-resolution scheme. Within this scheme, we can specify in what context conflicts may appear and at what interpretation level they must be resolved. The
conflict-resolution scheme resolves conflicting interpretations by comparing their judgment values and choosing the interpretation that has the highest judgment value. We propose a comparison function that is learned from examples. Concerning the interpretation mechanism, we argue for the use of a priority queue as a fundamental data structure to store and retrieve hypotheses of objects. A priority queue enables a correct interpretation order in a mixed bottom-up and top-down fashion. Furthermore, it enables the specification of procedural knowledge on interpretation by assigning different priorities to objects. Chapters 6 and 7 demonstrate the general applicability of the document-interpretation framework.

Bibliography


Part I

Binarization
Chapter 2

Image sharpening by morphological filtering

This chapter introduces a class of iterative morphological image operators with applications to sharpen digitized gray-scale images. The image operators are especially useful for sharpening scanned black-and-white documents. It is proven that all image operators using a concave structuring function have sharpening properties and we derive that these image operators have a property in which it is the sign of the Laplacian at a pixel of the image to be sharpened that determines whether that pixel is going to be dilated or eroded by the image-sharpening operator. By using the Laplacian property, we introduce the underlying partial differential equation (PDE) that governs this class of iterative image operators. The parameters, i.e. the number of iterative applications and the structuring function size of the image operator, can be determined on the basis of an estimation of the amount of blur present in the image. For discrete algorithmic implementations of the image-operator class it is shown that image operators using a parabolic structuring function have an efficient implementation and isotropic sharpening behavior.

This chapter is accepted for publication in the special issue on Mathematical Morphology and Nonlinear Image Processing of the Pattern Recognition Journal.
2.1 Introduction

Kramer et al. [63] define a non-linear transformation for sharpening digitized gray-scale images. The transformation replaces the gray value at a pixel by either the minimum or the maximum of the gray values in its neighborhood, the choice depending on which one is closer in value to the original gray value. They show that after a finite number of iterative applications, the resulting image stabilizes, that is, every pixel becomes either a local maximum or a local minimum. Osher and Rudin derive the same property for shock filters for image enhancement [80]. They use partial differential equations and their discretizations and show that solutions of the equations satisfy a local maximum and minimum principle.

The advantages of the above non-linear transformation for sharpening digitized gray-scale images are multiple. The transformation is fast, computationally efficient, and easy to implement (local maximum and minimum operator). Furthermore, the transformation reduces the number of gray values present in the original image, and does not suffer from overshooting at edges, as opposed to methods based on the use of standard (linear) filters. These last methods as for instance, unsharp masking [93, 101,102] or the Laplacian operator [41], have the tendency to amplify noise and need clipping or scaling to make the resultant pixels span the interval [0, 255]. Verbeek et al., on the other hand, show in [118] that the non-linear transformation, denoted as dynamic-front operator, has edge-sharpening properties, but their dynamic front operator is not iterative. Practical applications of the sharpening transformation have been reported by Lester et al. in [65] and by den Hartog in [23].

In this chapter, it is shown that the transformation introduced by Kramer et al. actually is an instance of a class of morphological image operators that all have sharpening properties, which class can be characterized by a partial differential equation. Based on this generalization we show that there is another kind of image-sharpening operator that outperforms the original transformation introduced by Kramer et al. in both algorithm order complexity as well as isotropic sharpening behavior.

2.2 Introduction to mathematical morphology

In mathematical morphology [103], the transformation that replaces the gray value at a pixel by the (weighted) maximum of the gray values in its neighborhood is known as the gray-scale dilation image operator:

\[ (f \oplus g)(x) = \bigvee_{u \in \mathbb{R}^2} [f(u) + g(x-u)] \]  \hspace{1cm} (2.1)

in which function \( f(x), f : x \in \mathbb{R}^2 \mapsto f(x) \in \mathcal{R} \) is the original image, and \( g(x), g : x \in \mathbb{R}^2 \mapsto g(x) \in \mathcal{R} \) is the structuring function implicitly defining the weighted neighborhood. Similarly, the transformation that replaces the gray value at a pixel by the (weighted) minimum of the gray values in its neighborhood is known as the gray-scale erosion image operator:

\[ (f \ominus g)(x) = \bigwedge_{u \in \mathbb{R}^2} [f(u) - g(u-x)] \]  \hspace{1cm} (2.2)
Note that the dilation operator is extensive: \((f \oplus g)(x) \geq f(x)\), and the erosion operator is anti-extensive: \((f \ominus g)(x) \leq f(x)\). Figure 2.2(a) gives a one-dimensional example of the dilation and erosion image operator.

### 2.3 Structuring functions

In the remainder of this chapter, we focus on the following two structuring functions. First, the flat structuring functions as used by Kramer et al. in their original definition of the image-sharpening operator:

\[
e^\rho(x) = \begin{cases} 
0, & x^T x \leq \rho^2 \\
-\infty, & x^T x > \rho^2 
\end{cases}
\] (2.3)

Secondly, the quadratic structuring functions (QSF) as introduced by van den Boomgaard [116]:

\[
q^\rho(A)(x) = \rho q(A)(\frac{x}{\rho}) = -\frac{1}{2\rho} x^T A^{-1} x
\] (2.4)

where \(A\) is a \(2 \times 2\) positive definite symmetric matrix. Taking the unity matrix for \(A\) yields the symmetric two-dimensional parabolic structuring function:

\[
q^\rho(x) = -\frac{1}{2\rho} x^T x
\] (2.5)

For both types of a structuring function, the parameter \(\rho\) determines the size (or scale) of the structuring function. Figure 2.1 gives a two-dimensional example of a flat and parabolic structuring function. In practical applications of mathematical morphology, the use of a flat structuring function is widespread [63,64,73]. Its popularity stems from the fact that the gray-scale dilation and erosion operators using

![Figure 2.1: Parabolic and flat structuring function. (a) Parabolic structuring function \(q^\rho(x) = -\frac{1}{2\rho} x^T x\). (b) Flat structuring function \(e^\rho(x)\).](image)
a flat structuring function are easy to implement (local maximum and minimum filter) and result in fast algorithms. Van den Boomgaard et al. [117] prove for the class of quadratic structuring functions that:

- Any quadratic structuring function is dimensionally decomposable with respect to dilation:
  \[
  \forall A : \exists \alpha, \beta : (f \oplus q^\alpha(A))(x) = (f \oplus q^\alpha(R^T (0 \ 0 \ 0) R)) \oplus q^\beta(R^T (0 \ 0 \ 0) R))(x)
  \]
  \[(2.6)\]

- The general class of quadratic structuring functions contains the subclass of structuring functions that are rotationally symmetric: \(q^\rho(x) = -\frac{1}{2\rho}x^Tx\), i.e. those structuring functions for which
  \[
  A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}
  \]
  \[(2.7)\]

These properties allow for very efficient algorithms for implementing the parabolic dilation operator, that have been shown to be independent of the size of the structuring function [117]. Given these scalable structuring functions \(q^\rho(x)\) we redefine the dilation and erosion operator:

\[
F^\oplus(x, \rho) = (f \oplus q^\rho)(x)
\]
\[(2.8)\]

\[
F^\ominus(x, \rho) = (f \ominus q^\rho)(x)
\]
\[(2.9)\]

\[
F(x, 0) = F^\oplus(x, 0) = F^\ominus(x, 0) = f(x)
\]
\[(2.10)\]

The functions \(F^\oplus(x, \rho)\) and \(F^\ominus(x, \rho)\) are the morphological scale-space notations for the gray-scale dilation and gray-scale erosion image operators with structuring function \(q^\rho(x)\) [116]. The parameter \(\rho\) is the scale parameter of the morphological scale space. The original function is \(f(x)\) and \(x\) is the position. The operators \((f \oplus g)(x)\) and \((f \ominus g)(x)\) are the dilation and erosion operators as defined in Equations 2.1 and 2.2. Note that one can derive a scalable structuring function from any concave structuring function \(g(x)\) by means of umbral scaling:

\[
g^\rho(x) = \rho g\left(\frac{x}{\rho}\right)
\]
\[(2.11)\]

For example, the scalable parabolic structuring function \(q^\rho(x)\) can be derived from the structuring function \(q(x) = -\frac{1}{2}x^Tx\):

\[
q^\rho(x) = \rho q\left(\frac{x}{\rho}\right) = -\rho \frac{x^Tx}{2 \rho^2} = -\frac{1}{2\rho}x^Tx
\]
\[(2.12)\]

The scalable flat structuring function \(c^\rho(x)\) can be derived from structuring function

\[
c(x) = \begin{cases} 
    0, & x^Tx \leq 1 \\
    -\infty, & x^Tx > 1
  \end{cases}
\]
\[(2.13)\]

in a similar manner:

\[
c^\rho(x) = \rho c\left(\frac{x}{\rho}\right) = \begin{cases} 
    0, & \frac{x^Tx}{\rho^2} \leq 1 \\
    -\infty, & \frac{x^Tx}{\rho^2} > 1
  \end{cases} = \begin{cases} 
    0, & x^Tx \leq \rho^2 \\
    -\infty, & x^Tx > \rho^2
  \end{cases}
\]
\[(2.14)\]
2.4 Image-sharpening operator class

In this section we give a definition of the image-sharpening operator class in terms of gray-scale dilation image operators and gray-scale erosion image operators; it is an extension of Kramer’s original definition. Furthermore, we prove that iterative applications of the sharpening operator using a concave structuring function have sharpening properties.

2.4.1 Image-sharpening operator class definition

First, we rephrase the original transformation defined by Kramer et al. in the framework of mathematical morphology:

\[
E[f](x, \rho) = \begin{cases} 
F^\Theta(x, \rho), & F^\Theta(x, \rho) - F(x, 0) < F(x, 0) - F^\Theta(x, \rho) \\
F^\oplus(x, \rho), & F^\oplus(x, \rho) - F(x, 0) > F(x, 0) - F^\oplus(x, \rho) \\
F(x, 0), & \text{otherwise}
\end{cases}
\] (2.15)

This image-operator class is parameterized by the structuring function \(g^\rho(x)\). If we take a flat structuring function \(c^\rho(x)\) as the structuring function \(g^\rho(x)\), then the image-sharpening operator equals the original definition of Kramer et al. with one modification: Kramer et al. did not consider the special case where \(F^\oplus(x, \rho) - F(x, 0)\) equals \(F(x, 0) - F^\Theta(x, \rho)\). In that case, the image-sharpening operator as defined by Kramer et al. behaves as the dilation operator \((F^\oplus(x, \rho))\). In case of a single-slope signal \((\forall x : \nabla^2 f = 0)\), the application of the operator as defined by Kramer et al. results in a translation of the original signal, whereas this new definition preserves the original signal. Figure 2.2(b) gives a one-dimensional example of an application of the sharpening operator. Figure 2.3 gives a two-dimensional example of an application of the sharpening operator on a scanned part of a utility map.
\[ f \ominus g \]

original image

\[ \mathcal{E}[f] \]

sharpening result

\[ f \ominus g \]

Figure 2.3: One application of the image-sharpening operator \( \mathcal{E}[f] \). Shown are the original image \( f \), the dilation \( f \ominus g \), the erosion \( f \ominus g \), and the sharpening result \( \mathcal{E}[f] \).

\[ \begin{array}{ccc}
(a) & (b) & (c)
\end{array} \]

Figure 2.4: Choice of the size of a structuring function, i.e. speed versus accuracy. (a) Part of a scanned utility map (400 dpi). (b) Result of the image-sharpening operator with a \( 3 \times 3 \) flat structuring function: three iterative applications were necessary to achieve a sharp image. (c) Result after one iteration of the image-sharpening operator with a \( 7 \times 7 \) flat structuring function: the decimal dot is merged with the digit.
Note that Equation 2.15 defines one application of the sharpening operator. In practical applications, however, it is generally beneficial to use multiple iterative applications of the operator with a small structuring function. The size of the structuring function determines the sharpening speed. If we use a sharpening operator with a large structuring function, we require less iterative applications than with a small structuring function, but we lose small details in the resulting image, see for example Figure 2.4. In general, this choice is a trade-off between speed and accuracy. Repeated application of the sharpening operator for a fixed scale of the structuring function will sharpen the image until a fixed point is reached, i.e. application of the operator will no longer alter the image. In the following we show that this repeated sharpening behavior is controlled by a partial differential equation (PDE). By examining this PDE optimal structuring functions can be derived.

2.4.2 Laplacian properties for one-dimensional continuous functions

In this section it is shown that the sign of the Laplacian of function \( f(\mathbf{x}) \) at position \( \mathbf{x} \) determines whether \( f(\mathbf{x}) \) is going to be dilated or eroded by the sharpening operator. With this important property of the sharpening operator the PDE will be derived. For the sake of explanation, it is shown for one-dimensional continuous functions only, but it can simply be generalized to two-dimensional functions, which is shown in Section 2.4.3.

In the remainder of this chapter a concave function is defined as follows: a function \( g(\mathbf{x}), g : \mathbf{x} \in \mathcal{R} \mapsto g(\mathbf{x}) \in \mathcal{R} \) is concave if for all \( x_0, x_1 \in \mathcal{R} \) the line connecting the points \( (x_0, g(x_0)) \) and \( (x_1, g(x_1)) \) is below the function \( g(x) \). A function is convex when it is not concave. For any one-dimensional symmetric \( g^\rho(\mathbf{x}) = g^\rho(-\mathbf{x}) \) concave structuring function \( g^\rho(\mathbf{x}), g^\rho : \mathbf{x} \in \mathcal{R} \mapsto g^\rho(\mathbf{x}) \in \mathcal{R} \) and \( \rho \in \mathcal{R}^+ \), the following properties of the image-sharpening operator class hold:

\[
\mathcal{E}[f](\mathbf{x}, \rho) > f(\mathbf{x}) \quad \text{if } \nabla^2 f(\mathbf{x}) < 0 \tag{2.16}
\]

\[
\mathcal{E}[f](\mathbf{x}, \rho) < f(\mathbf{x}) \quad \text{if } \nabla^2 f(\mathbf{x}) > 0 \tag{2.17}
\]

\[
\mathcal{E}[f](\mathbf{x}, \rho) = f(\mathbf{x}) \quad \text{if } \nabla^2 f(\mathbf{x}) = 0 \tag{2.18}
\]

where \( \nabla^2 f(\mathbf{x}) \) is the Laplacian of function \( f(\mathbf{x}) \). Before proving Properties 2.16, 2.17, and 2.18, we introduce the slope transform \( \mathcal{S}[f](\mathbf{w}) \). For symmetric concave structuring functions \( g^\rho(\mathbf{x}) \), we have that for both cases \( \mathbf{x} > 0 \) and increasing \( \mathbf{x} \), as well as for \( \mathbf{x} < 0 \) and decreasing \( \mathbf{x} \), that the derivative of \( g^\rho(\mathbf{x}), \nabla g^\rho(\mathbf{x}) \) is a decreasing function. This implies that the intercept of a tangent line of \( g^\rho(\mathbf{x}) \) in \( \mathbf{x} \) with the functional axis (y-axis) is an increasing function for increasing \( \mathbf{x} > 0 \) as well as for decreasing \( \mathbf{x} < 0 \), as depicted in Figure 2.5(a). The intercept function is known as the slope transform \( \mathcal{S}[g^\rho](\mathbf{w}) \) of function \( g^\rho(\mathbf{x}) \), see Figure 2.5(a). The slope transform was introduced by Dorst and Van den Boomgaard in [30,31] and is closely related to the A-transform as introduced by Maragos [71]. The slope transform is a function of slope \( \mathbf{w} \) and is set valued. Because our structuring functions are concave, the slope transform of these functions is single valued. To prove Equation 2.16, we consider a function \( f(\mathbf{x}) \) with \( \nabla^2 f(\mathbf{x}) < 0 \) for a certain value of \( \mathbf{x}, x_1 \). To determine the dilation and erosion value of function \( f(\mathbf{x}) \) at \( x_1 \), we use the hit property of the dilation image operator and the hit property of the erosion image operator. The
hit property of the dilation image operator can be explained with Figure 2.5(b). To determine the dilation value $d_0$ at $x_1$ ($((f \circ g^0)(x_1))$ we place the inverse structuring function $-g^0(x)$ above function $f(x)$ in such a way that the origin of the inverse structuring function $-g^0(x)$ is located at $(x_1, +\infty)$. After that, we shift the origin downwards (i.e. we translate it along the functional axis) until it hits the function $f(x)$ (let's say in $(x_a, f(x_a))$). The function value of the thus translated origin sets the new dilation value of function $f(x)$ at $x_1$. Now assume that we take the interval $[x_0, x_1]$ so small that we may linearly approximate $f(x)$ within this interval. Note that because $\nabla f < 0$, $\nabla f(x_a) > \nabla f(x_b)$. Let us further assume that we have taken the scale of the structuring function sufficiently small ($p \downarrow 0$) that the hit position is within the above interval, i.e. $x_0 < x_a < x_1$. Then the gradient of the structuring function is equal to the gradient of function $f(x)$ at $x = x_a$:

$$\nabla f(x_a) = \nabla -g^0(x_1 - x_a) = \nabla g^0(x_a - x_1)$$

(2.19)

The dilation value $d_0$ at $x_1$ can now simply be derived from the slope transform, i.e. $d_0 = S[g^0](\nabla f(x_a))$. The same applies for the hit property of the erosion image operator. In that case, we use structuring function $g^0(x)$ and place its origin below function $f(x)$, at $x_1$, i.e. $(x_1, -\infty)$ and shift it upwards. Consequently, the erosion value $d_1$ at $x_1$ equals $d_1 = S[g^0](\nabla f(x_b))$. Because $\nabla^2 f(x_1) < 0$, we have

$$\nabla f(x_a) > \nabla f(x_b)$$

(2.20)
and

\[
S[g^p](\nabla f(x_0)) < S[g^p](\nabla f(x_0)) < S[g^p](\nabla f(x_0)) \quad (2.21)
\]

we derive that \(d_0 < d_1\). This sets the image-sharpening operator value at \(x_1\) to the dilution value \(F(\mathbf{x}_1, \rho)\) and proves Property 2.16. Property 2.17 is proven by the duality of the erosion image operator. Property 2.18 is proven by the fact that when \(\nabla^2 f(x) = 0\), the sharpening operator equals \(F(x,0) = f(x)\).

### 2.4.3 Laplacian properties for two-dimensional continuous functions

Generalization of Equations 2.16–2.17 to the two-dimensional case is not trivial because the Laplacian is defined as:

\[
\nabla^2 f(x) = \frac{\partial^2 f}{\partial x_0^2} + \frac{\partial^2 f}{\partial x_1^2} \quad (2.22)
\]

Now there are two cases for which \(\nabla^2 f(x) < 0\):

- **case 1:** \(\frac{\partial^2 f}{\partial x_0^2} < 0\) and \(\frac{\partial^2 f}{\partial x_1^2} < 0\) \quad (2.23)
- **case 2:** \(\frac{\partial^2 f}{\partial x_0^2} < 0\), \(\frac{\partial^2 f}{\partial x_1^2} > 0\) and \(\frac{\partial^2 f}{\partial x_0^2} > \frac{\partial^2 f}{\partial x_1^2}\) (and vice versa for \(x_0\) and \(x_1\)) \quad (2.24)

and two cases for which \(\nabla^2 f(x) > 0\):

- **case 3:** \(\frac{\partial^2 f}{\partial x_0^2} > 0\) and \(\frac{\partial^2 f}{\partial x_1^2} > 0\) \quad (2.25)
- **case 4:** \(\frac{\partial^2 f}{\partial x_0^2} > 0\), \(\frac{\partial^2 f}{\partial x_1^2} < 0\) and \(\frac{\partial^2 f}{\partial x_0^2} > \frac{\partial^2 f}{\partial x_1^2}\) (and vice versa for \(x_0\) and \(x_1\)) \quad (2.26)

For case 1, \(f(x)\) is concave at position \(x\). When we use a rotational-symmetric concave structuring function \(g^p(x)\) for the sharpening operator, the dilution value at position \(x\) is closer in value to the original gray value than the erosion value. Therefore, Property 2.16 still holds for two-dimensional continuous functions; its proof is analogous to the proof for one-dimensional continuous functions.

For case 3, \(f(x)\) is convex. When we again use a rotational-symmetric concave structuring function \(g^p(x)\) for the sharpening operator the value of the erosion at position \(x\) is closer to the original gray value than the dilution value. As a consequence, Property 2.17 also holds for two-dimensional continuous functions; its proof is analogous to the proof for one-dimensional continuous functions.

For cases 2 and 4, \(\partial^2 f/\partial x_0^2\) and \(\partial^2 f/\partial x_1^2\) do not have the same sign, hence \(f(x)\) is neither convex nor concave in \(x\). In these cases it will also depend on the values \(\partial^2 f/\partial x_0^2\) and \(\partial^2 f/\partial x_1^2\) whether the dilution value or the erosion value is chosen as the resulting value after the application of the sharpening operator. The final choice is made on the basis of the highest partial second-order derivative value, giving the lowest slope transform value in \(x_0\) or \(x_1\).
2.4.4 Sharpening properties for a one-dimensional edge model

To demonstrate the sharpening properties of the image-sharpening operator, we construct an analytical edge model. Let the original edge be represented by function $i(x)$, $x \in \mathbb{R} \mapsto i(x) \in \mathbb{R}$. Suppose further that the recording of the original edge can be modeled by a linear system. Hence, the recorded edge function $f(x)$, $f : x \in \mathbb{R} \mapsto f(x) \in \mathbb{R}$ is given by:

$$f(x) = i(x) * h(x) \quad (2.27)$$

where $*$ is the convolution operator and $h(x)$ is the point-spread function (PSF), $h : x \in \mathbb{R} \mapsto h(x) \in \mathbb{R}$. Function $f(x)$ is depicted in Figure 2.6(c). Let us assume a symmetrical lens, i.e. $h(x) = h(-x)$, with finite aperture, i.e. $h(x) = 0$ for $x < -a$ and $x > a$ for some $a \in \mathbb{N}$. Moreover $h(x) \geq 0$ and $h(x)$ is a decreasing function for increasing and decreasing $x$. Point-spread function $h(x)$ is depicted in Figure 2.6(b). Suppose the original edge is an ideal step edge, that is, $i(x)$ is the unit step function:

$$i(x) = \begin{cases} 
1, & x \leq 0 \\
0, & x > 0 
\end{cases} \quad (2.28)$$

Function $i(x)$ is shown in Figure 2.6(a). When convolving with $h(x)$, note that $\nabla f(x) < 0$ for $x \in [-a, a]$ and, as $h(x)$ is decreasing and symmetric, that $\nabla^2 f(x) < 0$ for $x \in [-a, 0)$, $\nabla^2 f(x) > 0$ for $x \in (0, a]$, and $\nabla^2 f(x) = 0$ for $x = 0$. The image-sharpening operator $\mathcal{E}[f]$ only changes function $f(x)$ at points $x$ that have $\nabla^2 f(x) \neq 0$. Consider the interval $[-a, 0)$ in which points $x$ have $\nabla^2 f(x) < 0$, as depicted in Figure 2.6(c). This interval represents a concave part of the function $f(x)$. One application of the image-sharpening operator with a concave structuring function $g^\rho(x)$ results in $F^\Theta(x, \rho)$ at this interval, as shown in Figure 2.6(d). As $F^\Theta(x, \rho) > f(x)$ is true at this interval (Equation 2.16), the interval at which points $x$ have $\nabla^2 F^\Theta(x, \rho) < 0$, i.e. $[-b, 0)$, becomes smaller than the original interval $[-a, 0)$, at which $\nabla^2 f(x) < 0$. Furthermore, because we use a concave structuring function $g^\rho(x)$, $F^\Theta(x, \rho)$ is also concave at the interval $[-a, 0)$ (proven in [30]). As a consequence, repeated applications of the image-sharpening operator on the interval $[-a, 0)$ result in an interval $[-a, -c)$, with $-b < c < 0$, at which all points $x$ have function values equal to the maximum function value in the interval $[-a, 0)$, i.e. $f(-a)$. The same holds for the convex interval $(0, a]$ with points $x$ having $\nabla^2 f(x) > 0$. In this case, repeated applications of the image-sharpening operator result in an interval $(c, a]$, with $0 < c < b$, at which all points $x$ have function values equal to the minimum function value in the interval $(0, a)$, i.e. $f(a)$. After a finite number of applications of the image-sharpening operator, the blurred function $f(x)$ is sharpened to the original picture $i(x)$. The exact number of needed sharpening applications is derived in Section 2.4.6.

2.4.5 Sharpening properties for a two-dimensional edge model

If the structuring function $g^\rho(x)$ is rotationally symmetric and concave, the sharpening properties of the sharpening operator also hold for edges with arbitrary orien-
Figure 2.6: (a) Original picture \( i(x) \). (b) Point-spread function \( h(x) \). (c) Blurred version \( f(x) \) of original picture \( i(x) \). (d) One iteration of the image-sharpening operator.

tations in two-dimensional images. For two-dimensional oriented edges, the sharpening is perpendicular to the direction of the edge, and can be considered to be a collection of one-dimensional sharpening applications, as discussed in Section 2.4.4. Figure 2.7(a) gives a visual example. This property can be derived using the hit property of the dilation and erosion operator and the fact that the structuring function is rotationally symmetric, see Figures 2.7(b) and 2.7(c). The isophotes of Figure 2.7(a) run parallel to the edge. As such, when we sharpen the edge, the structuring function will hit the edge in a point for which the line through the point and the origin of the structuring function is perpendicular to the direction of the edge, see Figure 2.7(b). This implies that sharpening a point \( x \) on the edge only requires function values of points on the line that runs through \( x \) and is perpendicular to the edge. As a result, sharpening two-dimensional oriented edges can be considered to be a collection of one-dimensional sharpening applications, as shown in Figure 2.7(c). Both the flat and parabolic structuring function are rotationally symmetric. However, discrete approximations of small flat structuring functions are not isotropic and can cause anisotropic sharpening behavior, as discussed in Section 2.8.1.
2.4.6 Number of iterative applications

In this section we derive the number of iterative applications of the sharpening operator necessary to sharpen one edge. The number of applications depends on the PSF $h(x)$ and the size of the structuring function. We use the one-dimensional and two-dimensional edge models of Sections 2.4.4 and 2.4.5. Furthermore, we distinguish between the use of a flat structuring function and a parabolic structuring function.

When we use a parabolic structuring function $q^p(x)$ with $p = \Delta \rho$, the determination of the number of necessary applications of the sharpening operator to sharpen a one-dimensional or two-dimensional oriented edge can be visualized with Figure 2.6(c). We have to transport the maximum of $(-a, f(-a))$ and the minimum of $(a, f(a))$ towards $x = 0$. This transport is done by consecutive applications of the sharpening operator. It can be proven that recursive applications of a parabolic dilation of function $f(x)$ with structuring function $q^{\Delta \rho}(x)$ can be performed by one parabolic dilation (same holds for erosion) [116]:

$$f \oplus q^{\Delta \rho} \oplus \cdots \oplus q^{\Delta \rho} = f \oplus q^{j\Delta \rho} \quad (2.29)$$

As a result, we need to determine the parabola $q^{\rho_{tot}} = q^{j\Delta \rho}$ that sets the maximum value at $x = 0$. This equals finding $\rho_{tot}$ for which $q^{\rho_{tot}}(a) = 0$:

$$q^{\rho_{tot}}(a) = 0 \quad (2.30)$$

$$-\frac{a^2}{2\rho_{tot}} = 0 \quad (2.31)$$

This is only true for $\rho_{tot} \to \infty$. In the discrete domain this equals finding $\rho_{tot}$ for
which \( q^{\text{tot}}(a) > -\frac{1}{2} \) the quantization level:

\[
q^{\text{tot}}(a) > -\frac{1}{2} \tag{2.32}
\]

\[
-\frac{a^2}{2\rho_{\text{tot}}} > -\frac{1}{2} \tag{2.33}
\]

\[
\rho_{\text{tot}} > a^2 \tag{2.34}
\]

After the determination of the value of \( \rho_{\text{tot}} \), we can determine the number of necessary applications, denoted \( j \), of the operator to sharpen an edge using a parabolic structuring function:

\[
j = \frac{\rho_{\text{tot}}}{\Delta \rho} \tag{2.35}
\]

If \( \Delta \rho = 1 \), the number of applications equals \( \rho_{\text{tot}} \). As the parabolic structuring function is rotationally symmetric the number of applications is the same for one- and two-dimensional oriented edges.

When we use a flat structuring function \( c^\rho(x) \) with \( \rho = 1 \), the number of necessary applications of the sharpening operator to sharpen a one-dimensional or two-dimensional oriented edge equals \( a \), the aperture of the point-spread function \( h(x) \). Consider the visualization of a one-dimensional edge in Figure 2.6(c). With each application of the image-sharpening operator, the maximum of \( (-a, f(-a)) \) and the minimum of \( (a, f(a)) \) move exactly one pixel \( (\rho = 1) \) towards \( x = 0 \) because:

\[
f \oplus c^1 \oplus \cdots \oplus c^1 = f \oplus c^j \tag{2.36}
\]

For a two-dimensional oriented edge, the maximum and minimum gray value move towards the center of the edge along the perpendicular direction of the edge (see Figure 2.7(b)). As a flat structuring function is rotationally symmetric, the number of necessary applications is the same for a one- and two-dimensional oriented edge.

If we enlarge the size of the structuring function \( c^\rho(x) \) by increasing \( \rho \), the number of applications of the image-sharpening operator decreases to

\[
j = \frac{a}{\rho} \tag{2.37}
\]

### 2.5 Image-sharpening operator: partial differential equation

This section introduces the partial differential equation (PDE) of the image-sharpening operator class. The PDE includes a partial derivative of sharpening operator \( \mathcal{E}[f] \) to \( \rho \), the sign of the Laplacian of function \( f(x) \), as well as the slope transform of the structuring function \( g^\rho(x) \). Given that \( g(x) \) is a concave structuring function
and \( g^\rho(x) = \rho g(\frac{x}{\rho}) \) (umbral scaling), we have:

\[
\frac{\partial F^\Theta}{\partial \rho} = \lim_{\Delta \rho \to 0} \frac{F^\Theta(x, \rho + \Delta \rho) - F^\Theta(x, \rho)}{\Delta \rho} = \lim_{\Delta \rho \to 0} \frac{(f \oplus g^{\rho+\Delta \rho})(x) - F^\Theta(x, \rho)}{\Delta \rho} \tag{2.38}
\]

\[
= \lim_{\Delta \rho \to 0} \frac{\left( (f \oplus g^\rho) \oplus g^{\Delta \rho} \right)(x) - F^\Theta(x, \rho)}{\Delta \rho} \tag{2.39}
\]

\[
= \lim_{\Delta \rho \to 0} \frac{(F^\Theta(x, \rho) \oplus g^{\Delta \rho})(x) - F^\Theta(x, \rho)}{\Delta \rho} \tag{2.40}
\]

\[
\frac{\partial F^\Theta}{\partial \rho} = \lim_{\Delta \rho \to 0} \frac{S[g^{\Delta \rho}](\nabla F^\Theta)}{\Delta \rho} \tag{2.41}
\]

\[
\frac{\partial F^\Theta}{\partial \rho} = \lim_{\Delta \rho \to 0} \frac{\Delta \rho S[g](\nabla F^\Theta)}{\Delta \rho} \tag{2.42}
\]

and by duality of the dilation and erosion image operator:

\[
\frac{\partial F^\Theta}{\partial \rho} = -S[g](\nabla F^\Theta) \tag{2.43}
\]

Equality 1 uses the property that dilation of a function \( f(x) \) by members of a family of umbral-scaled structuring functions \( g^\rho(x) \) forms an additive semi-group in \( \rho \) [30]:

\[
((f \oplus g^{\rho_1}) \oplus g^{\rho_2})(x) = (f \oplus g^{\rho_1+\rho_2})(x) \tag{2.44}
\]

Equality 2 stems from the property of the slope transform based upon umbral scaling of the structuring function \( g^{\Delta \rho}(x) \) [30]:

\[
g^{\Delta \rho}(x) = \Delta \rho g(\frac{x}{\Delta \rho}) \xrightarrow{\Delta \rho} \Delta \rho S[g](w) \tag{2.45}
\]

Equality 3 uses the hit property of the dilation operator, as discussed in Section 2.4.2. For Equality 3 we want to calculate \((F^\Theta(x, \rho) \oplus g^{\Delta \rho})(x) - F^\Theta(x, \rho)\), see also Fig-
2.5 Image-sharpening operator: partial differential equation

We refer to Equation 2.8(a). With the slope transform of function \( g^{\Delta \rho} \), we can set:

\[
(F^{\Theta}(x, \rho) \oplus g^{\Delta \rho})(x) - F^{\Theta}(x, \rho) = S[g^{\Delta \rho}](\nabla F^{\Theta})
\]  

(2.48)

Given Equations 2.44 and 2.45 we can derive the following partial differential equation for iterations of the sharpening operator with a structuring function of width \( \rho \):

\[
\frac{\partial \mathcal{E}[f]}{\partial \rho} = \begin{cases} 
\frac{F^{\Theta}(x, \rho)}{\partial \rho} = S[g](\nabla F^{\Theta}), & F^{\Theta}(x, \rho) - F(x, 0) < F(x, 0) - F^{\Theta}(x, \rho) \\
\frac{\partial F^{\Theta}(x, \rho)}{\partial \rho} = -S[g](\nabla F^{\Theta}), & F^{\Theta}(x, \rho) - F(x, 0) > F(x, 0) - F^{\Theta}(x, \rho) \\
\frac{\partial g^{\Delta \rho}(x, \rho)}{\partial \rho} = 0, & \text{otherwise}
\end{cases}
\]  

(2.49)

If we use Properties 2.16 and 2.17, this results in:

\[
\frac{\partial \mathcal{E}[f]}{\partial \rho} = - \text{sign}(\nabla^2 f) S[g](\nabla f)
\]  

(2.50)

where

\[
\text{sign}(\nabla^2 f)(x) = \begin{cases} 
1, & \nabla^2 f(x) > 0 \\
-1, & \nabla^2 f(x) < 0 \\
0, & \text{otherwise}
\end{cases}
\]  

(2.51)

In the remainder of this section, we derive the PDEs of the sharpening operator with a parabolic and a flat structuring function. For parabolic structuring functions, we have that \( g^p(x) = g^p(x) = -\frac{1}{2}x^2 \), \( g(x) = -\frac{1}{2}x^2 \), and \( S[g](w) = \frac{1}{2}|w|^2 \), see Figure 2.8(b). The PDE for iterations of the sharpening operator with a parabolic structuring function then becomes:

\[
\frac{\partial \mathcal{E}[f]}{\partial \rho} = - \text{sign}(\nabla^2 f) \frac{1}{2} |\nabla f|^2
\]  

(2.52)

Equation 2.52 is, apart from the term \(-\frac{1}{2} \text{sign}(\nabla^2 f)\), equal to the PDE of the morphological scale space [116] which is given by:

\[
\frac{\partial f}{\partial \rho} = |\nabla f|^2
\]  

(2.53)

For flat structuring functions we have that \( g^f(x) = c^f(x) \) and \( S[c](w) = |w| \), see Figure 2.8(c). The partial differential equation for iterations of the sharpening operator with a flat structuring function then becomes:

\[
\frac{\partial \mathcal{E}[f]}{\partial \rho} = - \text{sign}(\nabla^2 f) |\nabla f|
\]  

(2.54)

Equation 2.54 is similar to the PDE for two-dimensional shock filters for image enhancement, as obtained by Osher and Rudin [80]:

\[
u_t = -\sqrt{u_x^2 + u_y^2} F(\mathcal{L}(u))
\]  

(2.55)
in which $L(u)$ is set to

$$L(u) = u_{xx} \cdot u_x^2 + 2 \cdot u_{xy} u_x u_y + u_{yy} \cdot u_y^2$$  \hspace{1cm} (2.56)$$

and where $F$ is a Lipschitz continuous function which satisfies

(i) $F(0) = 0$, \hspace{1cm} (2.57)

(ii) $X(u)F'(u) > 0, \quad u \neq 0$, \hspace{1cm} (2.58)

where $X(u) = 1$ if $u > 0$, $X(u) = -1$ if $u < 0$, and $X(0) = 0$. Osher and Rudin use this $L(u)$ because it contains second directional derivatives, as opposed to the Laplacian ($L(u) = u_{xx} + u_{yy}$) which is curvature insensitive [45].

In Section 2.8, we experiment with the use of both structuring functions for applications of the sharpening operator in a discrete domain. We show that for small values of $\rho$, iterations of the image-sharpening operator using the gray-scale dilation and erosion image operators are a numerical difference scheme to solve the partial differential equation of the image-sharpening operator. Additionally, we show that the stability of the numerical difference scheme depends on the choice of the structuring function, the type of quantisation, and the minimum value $\rho$ that can be set for a structuring function in the discrete domain.

### 2.6 Discrete sharpening operator

The above derivations of the sharpening operator are applicable to the continuous domain. In this section, we discuss its application in the discrete domain. In the previous sections we have shown that the sharpening operator can be either defined by the PDE definition (Equation 2.49), or by the mathematical morphological definition of Equation 2.15. The PDE definition requires first- and second-order derivatives of the discrete image, which can be obtained by convolution with Gaussian derivatives [33]. Because this implies additional blurring of the discrete image as well as two extra convolutions, we focus on the (more efficient) mathematical morphological definition in this section. When using this definition we need to discretize the dilation and erosion operator. In [117], van den Boomgaard et al. present some very fast algorithms for the discrete dilation operator with a parabolic structuring function. The presented algorithms are independent of the size of the structuring function and are of order complexity $O(n)$, with $n$ the number of pixels in the original image. When using flat structuring functions, algorithms for the gray-scale dilation operator have order complexity $O(\rho^2 n)$ and are dependent on the structuring function size $\rho$.

### 2.7 Image sharpening with a continuous approximation

A drawback of the discrete sharpening operator using the mathematical morphological definition is that there is a minimum bound on $\rho$ for the parabolic and flat structuring function to ensure any sharpening effect. The minimum bound exists because the image and structuring function are both discrete. For a flat structuring function, the minimum value of $\rho$ equals the sampling distance (one pixel): $\rho \geq 1$. 
2.8 Experiments

The minimum value of $\rho$ for a parabolic structuring function depends on the number of quantization levels of the image.

In [117], Van den Boomgaard et al. describe an implementation of the gray-scale dilation operator that operates on a continuous approximation of the original discrete image. In this approximation, each pixel of the image is represented with a continuous parabola that has a parameter $\rho$ and a height equal to the gray value of the pixel. The implementation of the dilation operator with a parabolic structuring function, denoted as the union-of-translations implementation, modifies the $\rho$ values of the continuous parabolas of the image. A parabolic dilation with structuring function $g^{\Delta\rho}(x)$ is performed by enlarging each parabola with the amount $\Delta\rho$. The resulting image consists of a set of parabolas with size $\rho + \Delta\rho$ and forms a continuous approximation of the dilation result that can be discretized again. Further research should indicate whether it is possible to use this implementation of the dilation operator to make a discrete sharpening operator in which we can choose $\rho$ arbitrarily small.

2.8 Experiments

This section presents results of applications of the sharpening operator using flat and parabolic structuring functions in the discrete domain. In Section 2.8.1 the isotropy of both structuring functions is tested for two quantization types. Sections 2.8.2 and 2.8.3 give results of experiments on artificially generated step edges and on document images.

2.8.1 Isotropic sharpening

For accurate results of the sharpening operator in the discrete domain, where function values are given as quantized numbers on a uniform sampling grid, it is necessary to choose $\rho$ of the structuring function as small as possible. This section investigates the effects of sampling and quantizing structuring functions. It is shown for two types of quantization that the use of a parabolic structuring function is preferred to the use of a flat structuring function in terms of isotropic sharpening behavior. The first quantization type that is considered is uniform and equals the gray-scale range $[0, 255]$. The second type is non-uniform and uses a floating-point representation.

Uniform quantization with 256 gray values In this section the sharpening operator using a flat structuring function is compared with the sharpening operator using a parabolic structuring function for a uniform quantization with 256 gray values. Note that the size of a flat structuring function in this domain can only be as small as possible for $\rho = 1$ (sampling problem). For that value of $\rho$, the discrete approximation of the disk $S^\rho$ of the flat structuring function equals a diamond (4 connected) or a square (8 connected). The parabolic structuring function can be as small as possible, but in order to have any sharpening effect, $\rho$ has to have a minimum value of $\rho = 1$. For lower values of $\rho$, the sharpening operator is not able to fully sharpen the image; the operator only sharpens image edges to a maximum slope, set by the value of $\rho$. 
Figure 2.9: Applications of the sharpening operator in the case of uniform quantization with 256 gray values of the input image and structuring function. Different structuring functions are shown vertically and the number of iterative applications is shown horizontally. The original input image is a two-dimensional Gaussian function with \( \sigma = 9.0 \) and gray values fitted into the range \([0, 255]\).

Figure 2.9 show the results of the application of the sharpening operator for different numbers of iterations and different structuring functions in case of uniform quantization with 256 gray values. The original image is a digitized two-dimensional Gaussian function with gray values fitted into the range \([0, 255]\). The desired result of the application of the sharpening operator is a cylinder. From the results we may conclude that sharpening with a parabolic structuring function better resembles the desired result than sharpening with a 4-connected or 8-connected flat structuring function. This stems from the fact that sampling a parabolic structuring function gives a more isotropic structuring function than sampling a flat structuring function. However, the sampled and quantized parabolic structuring function contains (repeating) quantization errors, as noted in [94], which may influence the stability of the sharpening operator and the correctness of the sharpening result after several iterative applications of the operator.

Non-uniform quantization with a floating-point representation In case of floating-point function values given on a grid, the \( \rho \) value for the parabolic structuring functions can even be lower than 1 (down to \( \epsilon \)), which is determined by the floating-point precision. The minimal value of \( \rho \) for flat structuring functions remains 1. Figure 2.10 shows the results of the application of the sharpening operator for different numbers of iterations and different structuring functions in case of floating-point function values. The original image is again a digitized two-dimensional Gaussian function and consequently the desired result of the application of the image-sharpening operator is a cylinder. From the results, we may conclude that sharpening with a parabolic structuring function yields the desired result: a cylinder, whereas sharpening with a 4-connected or 8-connected flat structuring function gives cubic-like figures, which
are again due to their anisotropic behavior.

### 2.8.2 Edge sharpening

This section presents results of experiments with the sharpening operator on an artificially generated step edge in a discrete (uniform sampled and uniform quantized with 256 gray values) image, as shown in Figure 2.11(a). The image is distorted with Gaussian blur and additive Gaussian noise. The standard deviation of the blur and noise ranges between 0 and 25:

\[
\sigma_{\text{blur,noise}} = 0, 1, 3, 5, 10, 15, 25
\]  
(2.59)

The structuring functions used in the experiments are the flat structuring function and the parabolic structuring function. The choices for the size \( \rho \) of the flat structuring function have been:

\[
\rho = 0, 1, 2, 4, 8, 16, 32, 64
\]  
(2.60)

where 0 corresponds to a 4-connected flat structuring function and 1 to a 8-connected flat structuring function. The choices for the size \( \rho \) of the parabolic structuring function were:

\[
\rho = 0.125, 0.25, 0.5, 1, 2, 4, 8, 16
\]  
(2.61)

For both structuring functions, the range of iterations is chosen from 1 to 16 iterations. The error measure applied is the mean squared error (MSE) between the
Figure 2.11: (a) Artificially generated step-edge image. (b) Artificially generated image used in the experiment on document-image sharpening.

Figure 2.12: Edge-sharpening experiment. Minimal MSE for different values of $\sigma_{\text{blur}}$ and $\sigma_{\text{noise}}$ in case of (a) flat structuring functions and (b) parabolic structuring functions.
original image in Figure 2.11(a) and the sharpening result. For each value of $\sigma_{\text{blur}}$ and $\sigma_{\text{noise}}$ we record the minimal MSE for the flat and parabolic structuring function over the number of iterations and the range of structuring function size $\rho$. The error graph for the flat structuring functions is shown in Figure 2.12(a); the error graph for the parabolic structuring functions is shown in Figure 2.12(b).

![Graphs](image)

(a)  

(b)  

**Figure 2.13:** Edge-sharpening experiment. (a) The number of times a flat structuring function of size $\rho$ gives a minimal MSE as a function of $\rho$. (b) The number of times a parabolic structuring function of size $\rho$ gives a minimal MSE as a function of $\rho$.

From Figures 2.12(a) and 2.12(b) we may conclude that for $\sigma_{\text{blur}} \leq 5$ and $\sigma_{\text{noise}} \leq 5$, sharpening with a flat and parabolic structuring function can completely (in the MSE sense) reconstruct the original image. For higher values of $\sigma_{\text{blur}}$ and $\sigma_{\text{noise}}$, the error is not zero and gradually increases for increasing amounts of blur and noise. Although it is hard to compare the flat and parabolic structuring with different ranges of $\rho$, the flat structuring functions perform slightly better than the parabolic structuring functions. All minimum errors were found after one iteration. Figures 2.13(a) and 2.13(b) give the frequency of the different sizes $\rho$ of the structuring functions for which the minimum errors were found. Maximums are found at $\rho = 1$ (8 connected) for the flat structuring functions and at $\rho = 1$ for the parabolic structuring functions.

### 2.8.3 Document-image sharpening

In this section, results are shown of experiments with the sharpening operator on an artificially generated document image, as shown in Figure 2.11(b). The image consists of the numbers 0 to 99, each represented with two digits in a 32-points font. The width of the digits is 5 pixels on average. Two digits are at least one pixel apart. For the experiment we again distorted the image with Gaussian blur and additive Gaussian noise. The standard deviation of the blur ranges between 0 and 1.8:

$$\sigma_{\text{blur}} = 0, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8$$  \hspace{1cm} (2.62)
The standard deviation of the noise ranges between 0 and 7:

$$\sigma_{\text{noise}} = 0, 1, 2, 3, 4, 5, 6, 7$$  \hspace{1cm} (2.63)

The structuring functions used are the flat and parabolic structuring with the same range of size $\rho$ and the same number of iterations as in the previous experiment. The error measure applied is the number of disconnected digits after thresholding the sharpened image at threshold value $t = 122$. For each value of $\sigma_{\text{blur}}$ and $\sigma_{\text{noise}}$, we record the number of disconnected digits closest to 200 for the flat and parabolic structuring function over the number of iterations and the range of structuring function size $\rho$. The graphs with the number of disconnected digits are depicted in Figures 2.14(a) and 2.14(b). The corresponding numbers of noise objects are shown in Figures 2.15(a) and 2.15(b).

![Diagram](image)

**Figure 2.14:** Experiment on document-image sharpening. (a) The number of disconnected digits closest to 200 after sharpening with a flat structuring function and thresholding for different values of $\sigma_{\text{blur}}$ and $\sigma_{\text{noise}}$. (b) The number of disconnected digits with a parabolic structuring function.

From Figures 2.14 and 2.15 we may conclude that image sharpening with a flat structuring function restores more of the original digits in the image for higher values of $\sigma_{\text{blur}}$, but it gives many noise objects. Figures 2.16(a) and 2.16(b) give the frequency of the different sizes $\rho$ of the structuring functions for which the closest number of disconnected digits to 200 are found. Maximums are found at $\rho = 1$ (8 connected) for the flat structuring functions and at $\rho = 1$ for the parabolic structuring functions.
Figure 2.15: Experiment on document-image sharpening. The corresponding number of noise objects for (a) a flat structuring function and (b) a parabolic structuring function.

Figure 2.16: Experiment on document image sharpening. (a) The number of times a flat structuring function of size $\rho$ gives a number of disconnected digits closest to 200 as a function of $\rho$. (b) The number of times a parabolic structuring function of size $\rho$ gives a number of disconnected digits closest to 200 as a function of $\rho$. 
2.9 Conclusions

In this chapter we introduced a class of morphological image operators with applications to sharpen digitized gray-scale images. We defined the sharpening operator class in terms of the gray-scale dilation and erosion operator from the theory of mathematical morphology. Additionally, we derived the partial differential equation (PDE) that governs this class of operators. Furthermore, we showed with an analytical edge model that this class of image operators has sharpening properties when we use concave structuring functions. In this chapter, we focused on two instances of this class of operators: one sharpening operator using a flat structuring function, and one sharpening operator using a parabolic structuring function. For two types of quantization, we showed with experiments in the discrete domain that the use of a parabolic structuring function is to be preferred to a flat structuring function in terms of isotropic sharpening behavior. On the other hand, experiments on sharpening document images revealed that flat structuring functions perform better than parabolic structuring functions if we can accept a higher number of sharpening errors.
Chapter 3

Automatic thresholding with hysteresis

This chapter describes an alternative to global thresholding as an image-segmentation technique: thresholding with hysteresis. Thresholding with hysteresis (TwH) differs from global thresholding (GT) in the sense that TwH uses two thresholds instead of one. Furthermore, TwH also incorporates local spatial information as opposed to GT, which solely relies on histogram information. We outline a novel procedure that automatically sets the two thresholds of the hysteresis and show that this procedure is optimal. To prove this, we first formally define TwH within the theory of mathematical morphology and we show that TwH can further reduce the error made by (optimal) GT. By defining an appropriate error function that measures the number of misclassified pixels when the image contains a mixture of two normal (Gaussian) distributions of object and background pixels, it becomes possible to estimate the threshold parameters, namely by estimating the normal distributions through minimization of the error function. Experiments on artificially generated images with normal distributions of object and background pixels show superior results for automatic TwH when compared with the minimum-error thresholding algorithm as proposed by Kittler et al. [61]. The minimum-error thresholding algorithm was found to be the best thresholding algorithm of a set of global histogram-based thresholding algorithms by Glasbey [39].
3.1 Introduction

If a binary segmentation of an image into objects and background is required, this can be accomplished by exploiting the fact that the gray-value distributions of objects and background are different. Examination of the histogram will give a threshold value \( t \) so that each pixel can be labeled as object or as background, depending on whether its grey value is larger or smaller than \( t \). There exists an enormous amount of techniques that automatically determine the threshold level \( t \) on the basis of the histogram [54, 61, 81, 90, 114], for which Glasbey gives a comparison in [39].

The outline of this chapter is the following. Section 3.2 discusses the error made by optimal GT if the gray values of object and background pixels are normally distributed. Section 3.3 gives a definition of TWH within the framework of mathematical morphology. Additionally, in Section 3.3, we derive an error function for the calculation of the error made by TWH if the gray values of objects and background pixels are both normally distributed. Section 3.4 outlines an automatic procedure for the threshold selection for TWH given the error function from Section 3.3. Results of experiments with automatic TWH on artificially generated images are given in Section 3.5.

3.2 Optimal global thresholding

Suppose that an image \( f(x) \), \( f : x \in \mathbb{Z}^2 \mapsto f(x) \in [0, N - 1] \), with \( N \) the number of gray values, contains only two principal brightness regions, segmentation then becomes classification into two classes. The histogram of such an image may be considered an estimate of the brightness probability density function, \( p(z) \), \( p : z \in [0, N - 1] \mapsto p(z) \in \mathbb{R}^+ \). This overall density function is the sum or mixture of two a posteriori densities, i.e. \( p(z) = p(z \mid C_1)P(C_1) + p(z \mid C_2)P(C_2) \). If the shape of both a posteriori probability density functions are known or assumed, determining an optimal threshold (in terms of minimum classification error) for segmenting the image into the two brightness regions is well defined. Suppose that image \( f(x) \) contains two values that are normally distributed. The mixture probability density function is

\[
p(z) = P(C_1)p_1(z) + P(C_2)p_2(z)
\]

(3.1)

Substituting the a posteriori Gaussian distributions gives:

\[
p(z) = \frac{P(C_1)}{\sqrt{2\pi\sigma_1^2}} \exp\left(\frac{-(z - \mu_1)^2}{2\sigma_1^2}\right) + \frac{P(C_2)}{\sqrt{2\pi\sigma_2^2}} \exp\left(\frac{-(z - \mu_2)^2}{2\sigma_2^2}\right) = P(C_1)G(z, \mu_1, \sigma_1) + P(C_2)G(z, \mu_2, \sigma_2)
\]

(3.2)

where \( \mu_1 \) and \( \mu_2 \) are the mean values of the two brightness levels, \( \sigma_1 \) and \( \sigma_2 \) are the standard deviations about the means, and \( P(C_1) \) and \( P(C_2) \) are the a priori probabilities of the two classes (object and background), for which the total probability constraint

\[
P(C_1) + P(C_2) = 1
\]

(3.3)
holds. Thus, in this case, an arbitrary a priori distribution is described by five unknown parameters. When the parameters are known, the optimal threshold is defined as $t | P(C_1)G(z, \mu_1, \sigma_1) = P(C_2)G(z, \mu_2, \sigma_2)$ (Bayes' decision boundary). Suppose that the dark regions correspond to objects, while the bright regions correspond to the background, and objects belong to class 1 and background to class 2. The probability of (erroneously) classifying a background point as an object point is

$$E_1(t) = \sum_{z=0}^{t} p_2(z)$$  \hfill (3.4)

Similarly, the probability of classifying an object point as a background point is

$$E_2(t) = \sum_{z=t}^{N-1} p_1(z)$$  \hfill (3.5)

The overall probability of an error then becomes:

$$E(t) = P(C_2)E_1(t) + P(C_1)E_2(t)$$  \hfill (3.6)

and the optimal threshold $t_{opt}$ can simply be found by choosing $t$ for which $E(t)$ is minimal [41].

### 3.3 Thresholding with hysteresis

In this section we give a definition of TwH within the theory of mathematical morphology. TwH is defined as the propagation of one threshold set into the second threshold set. With an example, TwH is compared with optimal GT. Furthermore, we determine the probability of error of TwH, which depends on the choice of thresholds $t_1$ and $t_2$ and the number of propagation steps. The number of propagation steps corresponds to the size of the structuring element $S$ of the binary geodesic dilation.

#### 3.3.1 Introduction

TwH differs from GT in the sense that TwH uses two thresholds instead of one. Furthermore, TwH also incorporates local spatial information as opposed to GT, which solely relies on histogram information. We show that TwH can further reduce the error made by optimal GT.

TwH was first noted by Canny in [10]. To correctly detect edges in an image, Canny used two thresholds to segment edge pixels from non-edge pixels. One (conservative) threshold indicates which pixels surely belong to an edge and about which no decision can yet be made. The other (less conservative) threshold indicates which pixels possibly belong to an edge and which surely belong to the background. The next step is labeling as edge pixels those pixels that possibly belong to an edge and have a connected path to a pixel that surely belongs to an edge. Practical applications of TwH were also reported by Lať [64], Den Hartog [23] and Schave-maker [95]. A drawback of the above-mentioned practical applications of TwH is
that the described algorithms lack automatic selection of the algorithm's parameters. Furthermore, in all above-mentioned descriptions of TWH, a mathematical description of TWH is not given. In this chapter we formally define TWH within the theory of mathematical morphology and outline a novel procedure that automatically sets the parameters of TWH.

3.3.2 Definition of thresholding with hysteresis

In this section we give a definition of TWH in terms of morphological image operators \([46, 103]\). Given a gray-scale image \(f(x), f : x \in \mathbb{Z}^2 \mapsto f(x) \in \mathbb{Z}\), and a threshold \(t \in \mathbb{Z}\), the threshold set \(\{f(x)\}_t\) is defined as

\[
\{f(x)\}_t = \{x \mid f(x) \leq t\}
\]  

(3.7)

The threshold set \(\{f(x)\}_t\) is a set of pixels \(x \in \mathbb{Z}^2\). The dilation of a binary image (set) \(B\) within a binary mask image \(M\) with a structuring element \(S\) is known as the binary geodesic dilation or the propagation of \(B\) in \(M\) with \(S\) \([46]\):

\[
\rho(B, M, S) = \{x \mid (x \in M \land x \in \bigcup_{y \in B} S_y) \lor (x \not\in M \land x \in b)\}
\]  

(3.8)

Variables \(B\), \(M\), and \(S\) denote sets of pixels \(x \in \mathbb{Z}^2\). The set \(S_y\) equals set \(S\) with its origin translated to position \(y\). In this chapter we only use structuring elements \(S\) that are \(n \times n\) squares (\(n\) odd). A binary geodesic dilation with a \(n \times n\) square corresponds to a propagation with a \(3 \times 3\) square (8 connected) using \((n-1)/2\) propagation steps. Throughout this chapter we use the terms binary image and set interchangeably. Figure 3.1(c) gives a visual example of a binary geodesic dilation.

![Figure 3.1: (a) Original binary image (set) \(B\). (b) Mask binary image (set) \(M\). (c) Binary geodesic dilation \(\rho(B, M, S)\).](image)

Figure 3.1: (a) Original binary image (set) \(B\). (b) Mask binary image (set) \(M\). (c) Binary geodesic dilation \(\rho(B, M, S)\).

dilation. Note that, by definition, \(\rho(B, M, S) \subseteq M\). The consecution of two-fold thresholding and propagation of one threshold set within the other threshold set forms a thresholding with hysteresis:

\[
(\{f(x)\}_{t_1}, \{f(x)\}_{t_2}, S) \quad \text{if} \quad t_1 < t_2
\]  

(3.9)
where \( t_1 \) and \( t_2 \) are two thresholds with \( t_1, t_2 \in \mathcal{Z} \). If we set \( t_1 \geq t_2 \) then TwH equals a GT with \( t_1 \), i.e. \( \{f(x)\}_{t_1} \), because \( \{f(x)\}_{t_1} \supseteq \{f(x)\}_{t_2} \).

### 3.3.3 Thresholding with hysteresis: an example

Suppose that gray-scale image \( f(x) \) contains two gray values which are normally distributed, as given in Section 3.2. The image \( f(x) \) is depicted in Figure 3.2(a). The mixture probability density function \( p(z) \) of image \( f(x) \) is given by Equation 3.2. As an example, we have depicted an image with two brightness regions set by the

![Image](image-url)

Figure 3.2: (a) Image \( f(x) \) consists of two brightness regions with normally distributed gray values. (b) Optimal global thresholding of image \( f(x) \). (c) Histogram of image \( f(x) \).

means \( \mu_1 = 100 \) and \( \mu_2 = 151 \) in Figure 3.2(a). The standard deviations of the brightness regions are \( \sigma_1 = 8 \) and \( \sigma_2 = 15 \). There are as many object pixels as
background pixels, i.e. \( P(C_1) = P(C_2) = 0.5 \). In Figure 3.2(c) the histogram of image \( f(x) \) is shown; it is easy to see that the gray values of the two regions are overlapping. If we apply the optimal global threshold \( t_{opt} \), i.e. \( t = 119 \), to image \( f(x) \), then the image shown in Figure 3.2(b) is obtained. This image contains many (867) misclassified pixels.

In the remainder of this section we show the steps taken when applying TwH. The first step is to threshold the image \( f(x) \) with threshold \( t_1 < t_{opt} \) and equal to \( \bigvee_{t\in\{0,N-1\}} \{ t \mid p_1(t) > 0 \land p_2(t) = 0 \} \). For example, we can take \( t_1 = 102 \). The resulting threshold set \( \{ f(x) \}_{t_1} \) is depicted in Figure 3.3(b) and contains considerably less classified objects. Proceeding, threshold set \( \{ f(x) \}_{t_2} \) is propagated within threshold set \( \{ f(x) \}_{t_2} \) using a \( 3 \times 3 \) square as structuring element \( S \). The result of TwH, shown in Figure 3.3(d), has considerably less misclassified background pixels than when GT was used. However, the number of misclassified object pixels is the same for both methods, because the same upper threshold \( t_{opt} \) was used. If all possible combinations of values for thresholds \( t_1 \) and \( t_2 \) are evaluated, we obtain the error curve as depicted in Figure 3.4. The error is defined as the number of misclassified object and background pixels: \( E = \sum_{x \in C_1}(x \rightarrow \lambda_1 \mid x \in C_2) + \sum_{x \in C_2}(x \rightarrow \lambda_2 \mid x \in C_1) \). The minimum error equals 50 and is found for thresholds \( t_1 = 102 \) and \( t_2 = 128 \).

### 3.3.4 Error calculation for one propagation step

In this section we derive an error function for the calculation of the error made by TwH from threshold \( t_1 \) to threshold \( t_2 \), given the image model of Section 3.2. The measure of error is the number of misclassified object added to the number of misclassified background pixels. The cost of misclassifying an object pixel is taken as high as the cost of misclassifying a background pixel. The error depends on the choice of thresholds \( t_1 \) and \( t_2 \) as well as on the size of structuring element \( S \). In this section, we assume one propagation step, which corresponds to the use of a \( 3 \times 3 \) square as a structuring element \( S \) for the binary geodesic dilation. All derivations that follow are also based on the assumption that intensities are Gaussian.
3.3 Thresholding with hysteresis

Figure 3.4: The number of misclassified pixels when thresholding image \( f(x) \), as depicted in Figure 3.3(a), with hysteresis for different values of the thresholds \( t_1 \) and \( t_2 \). Structuring element \( S \) is a \( 3 \times 3 \) square.

distributed. Then the probability density function for discrete values is given by:

\[
p(z) = P(z - 0.5 \geq X < z + 0.5) = \Phi(z + 0.5) - \Phi(z - 0.5) \tag{3.10}
\]

where function \( \Phi(x) \), \( \Phi : x \in \mathcal{R} \mapsto \Phi(x) \in \mathcal{R} \) is the underlying continuous Gaussian distribution function of the pixel-intensity values. Furthermore, we denote function \( c_1[i], c_1 : i \in [0,8] \mapsto c_1[i] \in \mathcal{N} \) as the number of object pixels that have \( i \) object neighbors and function \( c_2[i], c_2 : i \in [0,8] \mapsto c_2[i] \in \mathcal{N} \) as the number of background pixels that have \( i \) background neighbors. Functions \( c_1[i] \) and \( c_2[i] \) count the number of co-occurrences of classes, in other words, they capture the statistics of the boundary between object and background pixels. Note that for \( 0 < j < 8 \), \( c_1[j] = c_2[8-j] \).

In the first step of TWH (determining the threshold set \( \{ f(x) \}_{t_1} \)), the error made, denoted as \( E_1(t_1) \), equals the following:

\[
E_1(t_1) = MP(C_2) \sum_{i=0}^{t_1} p_2(i) \tag{3.11}
\]

In error function \( E_1(t_1) \), \( E_1 : t_1 \in [0, N-1] \mapsto E_1(t_1) \in \mathcal{R}^+ \), \( M \) is the total number of pixels in the image, and \( P(C_2) \) equals the proportion of background pixels in the image. Error function \( E_1(t_1) \) corresponds to the number of background pixels having pixel gray values lower than threshold \( t_1 \) which are therefore misclassified as object pixels. In the propagation step of TWH (determining \( \rho(\{ f(x) \}_{t_1}, \{ f(x) \}_{t_2}, S) \)), the error made is:

\[
E_2(t_1, t_2) = \left( \sum_{i=t_1+1}^{t_2} p_2(i) \right) \sum_{j=1}^{8} \left[ c_2[j] \left( 1 - \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^j \right) \right] \tag{3.12}
\]
Error function \( E_2 \), \( E_2 : t_1, t_2 \in [0, N - 1] \mapsto E_2(t_1, t_2) \in \mathcal{R}^+ \) corresponds to the number of background pixels that are propagated from the misclassified pixels of error function \( E_1(t_1) \) within the mask \( \{ f(x) \}_{t_2} \). Error function \( E_3(t_1, t_2) \) equals the number of cases that a background pixel has a gray value between threshold \( t_1 + 1 \) and threshold \( t_2 \) and has at least one background neighbor with a gray value lower than threshold \( t_1 \). In the propagation step of TWH, another error that can be made is that object pixels which have a gray value \( z_1 \), with \( t_1 < z_1 \leq t_2, \) do not have a neighboring object or background pixel with a gray value \( z_2 \), with \( 0 < z_2 \leq t_1 \). This error function \( E_3(t_1, t_2) \) equals:

\[
E_3(t_1, t_2) = \left( \sum_{i=t_1+1}^{t_2} p_1(i) \right) \sum_{j=0}^{8} \left[ c_1[j] \left( \sum_{i=t_1+1}^{N-1} p_1(i) \right)^j \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^{8-j} \right] \quad (3.13)
\]

On the boundary between the objects and the background, an additional error can be made by the propagation. The corresponding error function, denoted as \( E_4(t_1, t_2) \), counts the number of background pixels that are misclassified by propagations of object pixels over the boundary. Error function \( E_4(t_1, t_2) \) equals:

\[
E_4(t_1, t_2) = \left( \sum_{i=t_1+1}^{t_2} p_2(i) \right) \sum_{j=1}^{8} \left[ c_2[8-j] \left( 1 - \left( \sum_{i=t_1+1}^{N-1} p_1(i) \right)^j \right) \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^{8-j} \right] \quad (3.14)
\]

Error function \( E_4(t_1, t_2) \) corresponds to the number of background pixels that have one or more neighboring object pixels with a gray value lower than threshold \( t_1 \). This number of background pixels does not include the background pixels which have neighboring background pixels with a gray value lower than threshold \( t_1 \). Those pixels have already been accounted for in error function \( E_2(t_1, t_2) \). We make no assumptions on the form of the boundary between object and background pixels. In determining the threshold set \( \{ f(x) \}_{t_2} \), the following error, denoted as \( E_5(t_2) \), is made:

\[
E_5(t_2) = MP(C_1) \sum_{i=t_2+1}^{N-1} p_1(i) \quad (3.15)
\]

Error function \( E_5(t_2) \), \( E_5 : t_2 \in [0, N - 1] \mapsto E_5(t_1, t_2) \in \mathcal{R}^+ \) corresponds to the number of object pixels that have gray values higher than threshold \( t_2 \). This sets the total error made by TWH with one propagation step to:

\[
E(t_1, t_2) = E_1(t_1) + E_2(t_1, t_2) + E_3(t_1, t_2) + E_4(t_1, t_2) + E_5(t_2) \quad (3.16)
\]

As error function \( E(t_1, t_2) \) is derived for one propagation step, it solely depends on the thresholds \( t_1 \) and \( t_2 \). In Section 3.3.5 we also take into account the number of propagation steps which corresponds to the size of the structuring element \( S \). The derivations of the error functions introduced in this section are described in more detail in Appendix B.
3.3.5 Error calculation for repeated propagation steps

In this section we derive the error made by Twh when two or more propagation steps are applied, again assuming the image model of Section 3.2. For this derivation the structuring element $S$ is fixed to an $n \times n$ square and the number of propagation steps is denoted by $k = (n - 1)/2$. In the following, we extend the individual errors defined in Section 3.3.4 to an error made when $k$ propagation steps are applied, which is denoted by $E^k$. The error functions $E_1(t_1)$ and $E_5(t_2)$ (Equation 3.11 and Equation 3.15) do not depend on $k$ because they are not related to the propagation; thus $E_{1.5}^k = E_{1.5}$. The error functions $E_2(t_1, t_2)$ and $E_4(t_1, t_2)$ (Equation 3.12 and Equation 3.14) increase when more propagation steps are applied, whereas error function $E_3(t_1, t_2)$ (Equation 3.13) decreases with more propagation steps.

The definition of error function $E_2^k(t_1, t_2)$, $E_2 : k \in N, t_1, t_2 \in [0, N - 1] \mapsto E_2^k(t_1, t_2) \in R^+$ is recursive. Error function $E_2^k(t_1, t_2)$ equals function $E_2^{k-1}(t_1, t_2)$ plus the number of background pixels which have a gray value between $t_1 + 1$ and $t_2$ and which have at least one pixel from the set of pixels counted by error function $E_2^{k-1}(t_1, t_2)$ as a neighbor:

$$E_2^1 = E_2$$

$$E_2^k(t_1, t_2) = E_2^{k-1}(t_1, t_2) + \left( \sum_{i=t_1+1}^{t_2} p_2(t) \right) \sum_{j=1}^{8} c_2[j] \left( 1 - (1 - P_{E_2^{k-1}})^j \right)$$

(3.17)

(3.18)

Error function $E_2(t_1, t_2)$ is defined by Equation 3.12. Probability $P_{E_2^{k-1}}$ is defined as:

$$P_{E_2^{k-1}} = \frac{E_2^{k-1}(t_1, t_2)}{MP(C_2)}$$

(3.19)

and represents the probability that a pixel from the set of pixels counted by error function $E_2^{k-1}$ occurs. Error function $E_2^k(t_1, t_2)$ corresponds to the number of pixels that are propagated from the misclassified pixels of error function $E_1(t_1)$ within the mask $\{f(x)\}_{t_2}$ for $k$ propagation steps.

Error function $E_3^k(t_1, t_2)$, $E_3 : k \in N, t_1, t_2 \in [0, N - 1] \mapsto E_3^k(t_1, t_2) \in R^+$ equals the number of object pixels that have a gray value between $t_1 + 1$ and $t_2$ and for which there is no 8-connected path of length $k + 1$ that consists of pixels that have a gray value between $t_1 + 1$ and $t_2$, and that ends with one pixel which has a gray value of $z$, with $0 \leq z \leq t_1$. Such an 8-connected path of length $k + 1$ has probability:

$$P_{\text{path}}(k) = \left( \sum_{i=0}^{t_1} p(i) \right) \left( \sum_{i=t_1+1}^{t_2} p(i) \right)^{k-1} \left( \sum_{i=t_1+1}^{t_2} p_1(i) \right)$$

(3.20)

The number of possible 8-connected paths of length $k + 1$ within structuring element $S$, denoted as $N_{\text{path}}(k)$, can be calculated with Algorithm 2, which is described in Appendix A. After the definition of the number of paths $N_{\text{path}}(k)$ and the probability $P_{\text{path}}(k)$ of a path, we can define error function $E_3^k(t_1, t_2)$:

$$E_3^k(t_1, t_2) = E_3^{k-1}(t_1, t_2) + MP(C_1) \left( 1 - \sum_{i=2}^{k} N_{\text{path}}(k) P_{\text{path}}(k) \right)$$

(3.21)
The definition of error function $E_k^k(t_1, t_2)$, $E_4 : k \in \mathcal{N}, t_1, t_2 \in [0, N - 1] \mapsto E_k^k(t_1, t_2) \in \mathcal{R}^+$ is recursive. Error function $E_k^k(t_1, t_2)$ equals function $E_{k-1}^k(t_1, t_2)$ plus the number of background pixels that have a gray value between $t_1 + 1$ and $t_2$, and that have as a neighbor at least one pixel from the set of pixels counted by error function $E_{k-1}^k(t_1, t_2)$ and not counted by error function $E_{2}^{k-1}(t_1, t_2)$:

$$E_4^1 = E_4$$

$$E_4^k(t_1, t_2) = E_{k-1}^k(t_1, t_2) +$$

$$\left( \sum_{i=t_1+1}^{t_2} p_2(i) \right) \sum_{j=1}^{8} c_2[j] \left( 1 - (1 - P_{E_k^{k-1}})^j \right) \left( 1 - P_{E_{2}^{k-1}} \right)^j$$

(3.23)

Error function $E_4(t_1, t_2)$ is defined by Equation 3.14. Probability $P_{E_{k}^{k-1}}$ is defined as:

$$P_{E_{k}^{k-1}} = \frac{E_{k}^{k-1}(t_1, t_2)}{MP(C_2)}$$

(3.24)

and represents the probability that a pixel from the set of pixels counted by error function $E_{k-1}^k$ occurs. The error function $E_4^k(t_1, t_2)$ counts the number of background pixels that are misclassified by $k$ consecutive propagation steps of object pixels over the boundary. This error function does not count background pixels which have been counted by error function $E_2^k(t_1, t_2)$. We again make no assumptions on the form of the boundary between object and background pixels. This sets the total error $E^k(t_1, t_2)$, $E : k \in \mathcal{N}, t_1, t_2 \in [0, N - 1] \mapsto E^k(t_1, t_2) \in \mathcal{R}^+$, made by TwH with $k$ propagation steps to:

$$E^k(t_1, t_2) = E_1(t_1) + E_2^k(t_1, t_2) + E_3^k(t_1, t_2) + E_4^k(t_1, t_2) + E_5(t_2)$$

(3.25)

The derivations of the error functions are described in more detail in Appendix B.

### 3.3.6 Optimal thresholding with hysteresis

In Section 3.3.5 we derived the error function $E^k(t_1, t_2)$ for the calculation of the error made (measured in the number of misclassified pixels) when we apply a TwH with repeated propagation steps. Optimal TwH is obtained when we apply TwH using the parameter settings for which the error function $E^k(t_1, t_2)$ is minimal. In order to determine the optimal values for thresholds $t_1$ and $t_2$ as well as the number of propagation steps $k$, we need, besides the probability density functions $p_1(z)$ and $p_2(z)$, the class co-occurrence functions $c_1[i]$ and $c_2[i]$. In other words, the parameter settings depend also on the tessellation of the image in object and background pixels. In general, the error function $E^k(t_1, t_2)$ increases when there are more neighboring object and background pixels. More neighboring object and background pixels occur for images that consist of a large number of small objects or for images that consist of objects that have irregular boundaries. Unfortunately, we generally do not know beforehand the tessellation of the image in object and background pixels. In fact, this tessellation is the desired result of TwH.

To overcome the above problem, there are two options. The first option is estimating the class-co-occurrence functions $c_1[i]$ and $c_2[i]$ on the basis of a priori
knowledge on quantity and morphology of the objects in the image. The second
option is excluding the use of a priori knowledge and using an automatic pro-
dure for estimating the class-co-occurrence functions. This procedure is described
in Section 3.4. Before describing the estimation procedure, we address an additional
feature of TWH that reduces thresholding errors even further. Until now, the TWH
considered uses the propagation of threshold set \( \{ f(x) \}_t \) within mask \( \{ f(x) \}_t \).
Another possibility is to use a propagation of threshold set \( \{ f'(x) \}_{t'} \) within mask
\( \{ f'(x) \}_{t'} \), when the original gray-scale image is inverted as well as thresholds \( t_1, t_2 \).
Gray-scale image \( f(x) \) is inverted by applying the expression:

\[
f'(x) = N - f(x)
\]  

(3.26)

Image \( f'(x) \), \( f' : x \in Z^2 \rightarrow f'(x) \in Z \), is the inversion of image \( f(x) \) and \( N \) is
the number of possible gray values. Thresholds \( t_1 \) and \( t_2 \) are inverted in a similar
manner:

\[
t'_1 = N - t_1
\]  

(3.27)

\[
t'_2 = N - t_2
\]  

(3.28)

Thresholds \( t'_1 \) and \( t'_2 \), \( t'_1, t'_2 \in [0, N - 1] \) are the inversions of thresholds \( t_1 \) and \( t_2 \),
respectively. Note that, as a result of the inversion: \( t'_2 < t'_1 \). If applied on the
inverted image \( f(x) \) with the thresholds \( t'_1 \) and \( t'_2 \), the error functions for TWH
remain the same. Throughout the remainder of this chapter, TWH applied on the
original image \( f(x) \) is denoted as forwards TWH, whereas applied on the inverted
image it is denoted as backwards TWH. The choice for forwards or backwards TWH
depends on the shape and gray values of objects and background. In Section 3.4
we outline an automatic parameter estimation procedure for optimal TWH that also
exploits the direction of hysteresis.

### 3.4 Automatic thresholding with hysteresis

In this section we outline the novel automatic estimation procedure of the probability
density functions of the gray values of both classes of pixels, i.e. \( p_1(z) \) and \( p_2(z) \),
as well as the statistics of neighboring pixels, i.e. class co-occurrence functions \( c_1[i] \)
and \( c_2[i] \). The estimations are used to select optimal thresholds \( t_1 \) and \( t_2 \), the
direction of hysteresis, and the number of propagation steps \( k \) by minimizing the error
function \( E^k(t_1, t_2) \). The estimations of the functions \( p_1(z) \) and \( p_2(z) \), and functions
\( c_1[i] \) and \( c_2[i] \), are obtained by an iterative application of TWH. The result of one
iteration, i.e. a segmentation of the image in object and background pixels, is used
in the next iteration of the procedure to refine the estimates. The segmentation
of the image in object and background pixels determines the probability functions
\( p_1(z) \) and \( p_2(z) \), and the class co-occurrence functions \( c_1[i] \) and \( c_2[i] \). These functions
can subsequently be used to minimize the error function \( E^k(t_1, t_2) \), to obtain
a new "suboptimal" TWH parameter setting that can be applied in the next iteration.
To start, we require an initial segmentation of the image \( f(x) \), for which an
optimal GT can be used. The procedure stops when the iterations converge to a
stable segmentation, which is measured by the stability of the estimated probability
density functions and the class co-occurrence functions \( c_1 \) and \( c_2 \). An estimation is considered to be stable if the choice of thresholds \( t_1 \) and \( t_2 \) and the direction of hysteresis remains the same over two iterations.

The above-sketched procedure is denoted as automatic thresholding with hysteresis (ATwH). ATwH draws on similarities with other automatic thresholding techniques, such as the iterative minimum-error thresholding algorithm (MET) as proposed in [61] and other iterative global thresholding algorithms. The main difference is that ATwH applies hysteresis. As a consequence, ATwH can incorporate pixel co-occurrence statistics in the estimations of the probability density functions.

Algorithm 1 Algorithm for automatic thresholding with hysteresis.

Require: \( f(x) \) is a gray-scale image

\[
\begin{align*}
\hat{h} & \leftarrow \text{histogram}(f(x)) \\
t & \leftarrow \text{MET}(\hat{h}) \\
\text{determine } \{f(x)\}_t \\
calculate \mu_1, \mu_2, \sigma_1, \sigma_2, c_1[i], c_2[i] \\
t_1, t_2, \text{dir} & \leftarrow \text{minimize } E^k \\
\text{repeat} \\
t_1^* & \leftarrow t_1 \\
t_2^* & \leftarrow t_2 \\
\text{dir}^* & \leftarrow \text{dir} \\
determine \rho(\{f(x)\}_{t_1}, \{f(x)\}_{t_2}, S(k)) \\
calculate \mu_1, \mu_2, \sigma_1, \sigma_2, c_1[i], c_2[i] \\
t_1, t_2, \text{dir} & \leftarrow \text{minimize } E^k \\
\text{until } t_1^* = t_1 \text{ and } t_2^* = t_2 \text{ and } \text{dir}^* = \text{dir}
\end{align*}
\]

Algorithm 3.4 outlines the automatic procedure. The procedure starts with an optimal GT of the original image, chosen according to the minimum-error threshold algorithm. From this segmentation in object and background pixels, we calculate the means \( \mu_1 \) and \( \mu_2 \) and the standard deviations \( \sigma_1 \) and \( \sigma_2 \) of the Gaussian probability density functions as well as the class-co-occurrence functions \( c_1[i] \) and \( c_2[i] \). With these parameters a selection of thresholds \( t_1 \) and \( t_2 \), number of propagation steps \( k \), and direction of hysteresis is made by minimizing the error function \( E^k \) for TwH. From this point on, the algorithm continues with a cycle of TwH, calculation of the parameters and minimization of the error function until the selection of parameters stays the same for two consecutive iterations.

In the calculations of the class-statistics parameters \( \mu_1, \mu_2, \sigma_1, \) and \( \sigma_2, \) and the calculation of the class-co-occurrence functions \( c_1[i] \) and \( c_2[i] \), we can make some refinements. If we use all pixels which are labeled as object or background pixels by the segmentation of an iteration, the estimate becomes a biased estimate of parameters \( \mu_1, \mu_2, \sigma_1, \) and \( \sigma_2. \) The estimates are biased because some pixels are misclassified and influence the mean and standard deviation of the other class. If we leave out object and background pixels that are likely to be misclassified, we have created a less-biased estimate of the parameters. Based on the error functions given in Section 3.3 that count the number of misclassified pixels, the functions \( C^*_1(t_1, t_2) \)
and $C_2^k(t_1, t_2)$ can be defined:

$$C_2^k(t_1, t_2) = E_2^k(t_1) + E_2^k(t_1, t_2) + E_2^k(t_1, t_2)$$  \quad (3.29)$$

$$C_2^k(t_1, t_2) = E_2^k(t_1, t_2) + E_2^k(t_2)$$  \quad (3.30)$$

Function $C_1^k(t_1, t_2)$, $C_1^k(t_1, t_2)$, $C_1 : k \in N, t_1, t_2 \in [0, N - 1] \mapsto C_1^k(t_1, t_2) \in \mathcal{R}^+$ counts the total number of misclassified object pixels for thresholds $t_1$ and $t_2$, and $k$ propagation steps. Function $C_2^k(t_1, t_2)$, $C_2^k(t_1, t_2)$, $C_2 : k \in N, t_1, t_2 \in [0, N - 1] \mapsto C_2^k(t_1, t_2) \in \mathcal{R}^+$ counts the total number of misclassified background pixels for thresholds $t_1$ and $t_2$, and $k$ propagation steps. Pixels that have a lot of neighboring pixels with different class memberships are likely to have been misclassified in the segmentation of an iteration of the automatic procedure. In our automatic procedure we leave out these pixels in the calculation of the statistics by the amount given by functions $C_1^k(t_1, t_2)$ and $C_2^k(t_1, t_2)$, but only if the number of object and background pixels is sufficient.

### 3.5 Experiments

In this section we show some experiments with ATwH on artificially generated images. The artificial images contain object and background pixels that both have normal gray-scale distributions. We compare ATwH with the MET algorithm, as proposed by Kittler in [61]. The minimum-error thresholding algorithm was found to be the best histogram-based global-thresholding algorithm by Glasbey in [39]. The experiments with artificially generated images are divided into two categories of experiments, based on the shape of the objects used:

**Category I** : One rectangular set of object pixels and one rectangular set of background pixels as depicted in Figures 3.2(a) and 3.3(a).

**Category II** : A uniform-random tessellation of circular objects with a fixed size on a constantly illuminated background. Figure 3.5 gives a visual example.

For both categories of experiments, object pixels and background pixels have a gray value according to a normal distribution with parameters $\mu_{1,2}$ and $\sigma_{1,2}$. Mean $\mu_1$ of the object pixels and mean $\mu_2$ of the background pixels were held fixed at:

$$\mu_1 = 100, \mu_2 = 151$$  \quad (3.31)$$

Standard deviations $\sigma_1$ of the objects and $\sigma_2$ of the background ranged over the values

$$\sigma_1, \sigma_2 = 1, 3, 5, 10, 15, 25$$  \quad (3.32)$$

except that cases were $\sigma_1 + \sigma_2 \leq 10$ were omitted. The proportion $\rho$ of object pixels were set to

$$\rho = 0.005, 0.01, 0.05, 0.1, 0.2, 0.4, 0.6, 0.8, 0.9, 0.95, 0.99, 0.995$$  \quad (3.33)$$

The range of parameters is equal to the range used in the experiments carried out by Glasbey in [39], for the case that there are only object and background pixels
and no remaining pixels. In both categories of experiments we did only consider one propagation step in the ATwH. Table 3.1 gives a comparison between ATwH and MET for category I experiments. From Table 3.1 we may conclude that in all cases ATwH performs better or at least as good as MET. Only for a small proportion of object or background pixels ($\rho = 0.005$, $\rho = 0.99$, and $\rho = 0.995$), we see a considerable degradation of performance of ATwH. For these values of $\rho$, the set of object pixels or the set of background pixels has a width of one or two pixels and therefore the advantage of TWH cannot be used fully.

Table 3.2 gives a comparison between ATwH and MET for category II experiments. From Table 3.2 we may conclude that the object size influences the performance of ATwH, nevertheless, it outperforms the MET method in all cases. Again, we notice a severe degradation of performance for small and large objects. Medium-sized objects give high performance and large standard deviations. The degradation of performance of ATwH for larger-sized objects is mainly due to the performances for object proportions of $\rho = 0.99$ and $\rho = 0.995$ as can be noticed from Figure 3.6. Figure 3.6 gives a comparison between ATwH and MET for different object sizes and object proportions.

### 3.6 Conclusions

In this chapter we introduced an alternative to global thresholding: thresholding with hysteresis. We gave a definition of thresholding with hysteresis within the theory of mathematical morphology and derived an error function which measures the
### 3.6 Conclusions

<table>
<thead>
<tr>
<th>$\rho$</th>
<th>ATwH vs. MET</th>
<th>$\mu$ difference</th>
<th>$\sigma$ difference</th>
<th>Converge ATwH</th>
<th>Converge MET</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.005</td>
<td>79%</td>
<td>-13451</td>
<td>15815</td>
<td>0%</td>
<td>21%</td>
</tr>
<tr>
<td>0.01</td>
<td>100%</td>
<td>-12941</td>
<td>15201</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>0.05</td>
<td>100%</td>
<td>-476.32</td>
<td>848.77</td>
<td>7%</td>
<td>0%</td>
</tr>
<tr>
<td>0.1</td>
<td>96%</td>
<td>-633.93</td>
<td>994.74</td>
<td>4%</td>
<td>4%</td>
</tr>
<tr>
<td>0.2</td>
<td>96%</td>
<td>-997.93</td>
<td>1445.6</td>
<td>0%</td>
<td>4%</td>
</tr>
<tr>
<td>0.4</td>
<td>100%</td>
<td>-1303.9</td>
<td>1953</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>0.6</td>
<td>100%</td>
<td>-1312.7</td>
<td>1980.1</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>0.8</td>
<td>96%</td>
<td>-989.21</td>
<td>1428.7</td>
<td>0%</td>
<td>4%</td>
</tr>
<tr>
<td>0.9</td>
<td>100%</td>
<td>-866.43</td>
<td>1358.1</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>0.95</td>
<td>93%</td>
<td>-512.43</td>
<td>893.68</td>
<td>7%</td>
<td>7%</td>
</tr>
<tr>
<td>0.99</td>
<td>79%</td>
<td>-6302</td>
<td>12491</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>0.995</td>
<td>54%</td>
<td>204.57</td>
<td>264.96</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>total</td>
<td>91%</td>
<td>-3298.6</td>
<td>8648.1</td>
<td>2%</td>
<td>3%</td>
</tr>
</tbody>
</table>

Table 3.1: Comparison results between ATwH and MET for category I experiments. The $\rho$ column denotes the proportion of object pixels. The second column gives the percentage of the number of times for which ATwH has less misclassified pixels than MET for the range of parameters $\sigma_1$ and $\sigma_2$. The third column gives the average difference $\mu$ (measured in number of pixels) between ATwH and MET. The fourth column gives the standard deviation $\sigma$ of the difference. The fifth and sixth column denote the percentage of the number of times for which the ATwH and MET algorithms did not converge within 32 iterations for the range of parameters $\sigma_1$ and $\sigma_2$.

<table>
<thead>
<tr>
<th>Object size</th>
<th>ATwH vs. MET</th>
<th>$\mu$ difference</th>
<th>$\sigma$ difference</th>
<th>Converge ATwH</th>
<th>Converge MET</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>86%</td>
<td>-1685.6</td>
<td>10428</td>
<td>2%</td>
<td>7%</td>
</tr>
<tr>
<td>8</td>
<td>92%</td>
<td>-3594.0</td>
<td>10855</td>
<td>2%</td>
<td>3%</td>
</tr>
<tr>
<td>16</td>
<td>96%</td>
<td>-4201.2</td>
<td>10234</td>
<td>2%</td>
<td>3%</td>
</tr>
<tr>
<td>32</td>
<td>97%</td>
<td>-4243.5</td>
<td>10237</td>
<td>2%</td>
<td>3%</td>
</tr>
<tr>
<td>64</td>
<td>95%</td>
<td>-4660.0</td>
<td>10667</td>
<td>2%</td>
<td>3%</td>
</tr>
<tr>
<td>128</td>
<td>95%</td>
<td>-5454.5</td>
<td>11421</td>
<td>1%</td>
<td>2%</td>
</tr>
<tr>
<td>256</td>
<td>92%</td>
<td>-5188.6</td>
<td>11188</td>
<td>1%</td>
<td>1%</td>
</tr>
<tr>
<td>512</td>
<td>90%</td>
<td>-4700.9</td>
<td>10722</td>
<td>1%</td>
<td>1%</td>
</tr>
<tr>
<td>1024</td>
<td>90%</td>
<td>-4101.1</td>
<td>10040</td>
<td>2%</td>
<td>3%</td>
</tr>
<tr>
<td>2048</td>
<td>90%</td>
<td>-4057.4</td>
<td>9934</td>
<td>2%</td>
<td>3%</td>
</tr>
<tr>
<td>4096</td>
<td>87%</td>
<td>-3809.1</td>
<td>9642.4</td>
<td>2%</td>
<td>5%</td>
</tr>
</tbody>
</table>

Table 3.2: Comparison results between ATwH and MET for category II experiments. The first column denotes the size of objects in the artificial image.
number of misclassified pixels. For the error function, we assumed that gray values of the object pixels and background pixels are normally distributed. Furthermore, we outlined a novel automatic iterative estimation procedure for thresholding with hysteresis that estimates the distributions of the gray values of object and background pixels and minimizes the error function to automatically select the threshold values, the number of propagation steps, and the direction of hysteresis. Experiments on artificially generated images show superior results for automatic thresholding with hysteresis when compared with the minimum-error thresholding algorithm.

Appendix A

In this appendix we present an algorithm that is needed for the derivation of the error function $B_{\sigma}^k(t_1, t_2)$ of Section 3.3.5. Algorithm 2 calculates the number of possible 8-connected paths with length $k + 1$ within structuring elements $S$. We denote $S^1$ as the $3 \times 3$ square, $S^2$ as the $5 \times 5$ square, etc. We denote elements of set $S^k$ as $(x, y) \in S^k$ with $x, y \leq k$ and $x, y \geq -k$. We denote a path of length $k + 1$ as an array of (8-connected) elements of $S^k$: $[(x_0, y_0), \ldots, (x_k, y_k)]$. The algorithm uses a priority queue [107] as the main data structure. The two major operations
on a priority queue \( Q \) are to \textit{enqueue} an element \( e \) with priority \( p \), denoted as \( Q.\text{enqueue}(e, p) \), and to \textit{serve} an element with the highest priority from the queue, denoted as \( (e, p) = Q.\text{serve}() \). In our algorithm, we use paths as queue elements and path lengths as priorities. Algorithm 2 starts with enqueuing all possible 8-

\textbf{Algorithm 2} Algorithm for the calculation of the number of possible 8-connected paths of length \( k + 1 \).

\textbf{Require:} structuring element \( S[x, y] \), empty priority queue \( Q \)

\{\textit{enqueue paths of} \( S^1 \)\}
- \( Q.\text{enqueue}([(0, 0), (0, -1)], 1) \)
- \( Q.\text{enqueue}([(0, 0), (1, -1)], 1) \)
- \( Q.\text{enqueue}([(0, 0), (1, 0)], 1) \)
- \( Q.\text{enqueue}([(0, 0), (1, 1)], 1) \)
- \( Q.\text{enqueue}([(0, 0), (0, 1)], 1) \)
- \( Q.\text{enqueue}([(0, 0), (-1, 1)], 1) \)
- \( Q.\text{enqueue}([(0, 0), (-1, 0)], 1) \)
- \( Q.\text{enqueue}([(0, 0), (-1, -1)], 1) \)

\textbf{repeat}

\{\textit{serve a path}\}
- \( ([\ldots, (x_i, y_i)], p) = Q.\text{serve()} \)

\{\textit{extend path and enqueue}\}

\textbf{for all} 8-connected neighbors \( n \) of \( (x_i, y_i) \) \textbf{do}

\textbf{if} \( n \not\in S^1 \) \textbf{and} \( n \) \textbf{is not an} 8-connected neighbor of \( (x_{i-1}, y_{i-1}) \) \textbf{then}
- \( Q.\text{enqueue}([\ldots, (x_i, y_i), n], p + 1) \)
\textbf{end if}
\textbf{end for}

\textbf{until} no queue elements left with length \( l < k \)

connected paths within structuring element \( S^1 \). After that, the algorithm continues to extend paths by serving them from the priority queue (shortest path first) and adding an 8-connected neighbor to the end of the path. The 8-connected neighbors may not be elements of \( S^1 \) or 8-connected neighbors of the last path pixel but one. After the algorithm has finished, the queue is left with all possible paths of length \( k + 1 \).

\textbf{Appendix B}

In this appendix the derivations of the error functions introduced in Section 3.3.4 are described in more detail. Figure 3.7 gives a visual example of functions \( p_1(z) \) and \( p_2(z) \), thresholds \( t_1 \) and \( t_2 \), and errors that correspond with certain function integration areas of \( p_1(z) \) and \( p_2(z) \) that are set by the two thresholds.

\textbf{One propagation step}

The derivations of errors functions \( E_1(t_1) \) and \( E_5(t_2) \) (Equations 3.11 and 3.15) are trivial. Error \( E_1 \) corresponds to background pixels having a gray value lower than
threshold $t_1$ (A). The probability that a background pixel has a gray value
in $A$ equals $\sum_{i=0}^{t_1} p_2(i)$. The total number of background pixels in $A$ is calculated by multiplying this probability with the total number of background pixels ($MP(C_2)$), yielding error function $E_1$:

$$E_1(t_1) = MP(C_2) \sum_{i=0}^{t_1} p_2(i)$$  \hspace{1cm} (3.34)

Error $E_5$ is derived in a similar way. Errors $E_2$, $E_3$, and $E_4$ are made in the propagation step. Error $E_2$ equals the number of background pixels in $B$ that have at least one background neighbor in $A$. The probability that a background pixel has a gray value in $B$ equals $\sum_{i=t_1+1}^{t_2} p_2(i)$. The probability that a background pixel has at least one background neighbor in $A$ equals the probability that the pixel has eight background neighbors which are not all eight in $B$ or $C$, plus the probability that the pixel has seven background neighbors which are not all in $B$ or $C$, and one object neighbor plus the probability that the pixel has six background neighbors which are not all six in $B$ or $C$, and two object neighbors, etc. The probability that
$j$ background pixels are not in $B$ or $C$ equals:

$$1 - \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^j$$

(3.35)

Multiplying this with the number of background pixels in the image that have $j$ background neighbors, i.e. $c_2[j]$, yields:

$$c_2[j] \left( 1 - \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^j \right)$$

(3.36)

Summation over the eight possibilities and multiplying with the probability that the background pixel is in $B$ gives error $E_2$:

$$E_2(t_1, t_2) = \left( \sum_{i=t_1+1}^{t_2} p_2(i) \right)^8 \left[ c_2[j] \left( 1 - \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^j \right) \right]$$

(3.37)

Error $E_3$ is the number of object pixels in $B$ that have no neighboring object or background pixel in $A$. The probability that an object pixel has a gray value in $B$ equals $\sum_{i=t_1+1}^{t_2} p_1(i)$. The probability that a pixel has no neighboring pixels in $A$ equals the probability that it has eight neighboring object pixels in $B$ or $C$ plus the probability that it has seven neighboring object pixels in $B$ or $C$ and one background pixel in $B$ or $C$, etc. The probability that $j$ neighboring object pixels and $8-j$ background pixels are in $B$ or $C$ equals:

$$\left( \sum_{i=t_1+1}^{N-1} p_1(i) \right)^j \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^{8-j}$$

(3.38)

Multiplying this with the number of object pixels in the image that have $j$ object neighbors, i.e. $c_1[j]$, yields:

$$c_1[j] \left( \sum_{i=t_1+1}^{N-1} p_1(i) \right)^j \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^{8-j}$$

(3.39)

Summation over the nine possibilities and multiplying with the probability that the object pixel is in $B$ gives error $E_3$:

$$E_3(t_1, t_2) = \left( \sum_{i=t_1+1}^{t_2} p_1(i) \right)^8 \left[ c_1[j] \left( \sum_{i=t_1+1}^{N-1} p_1(i) \right)^j \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^{8-j} \right]$$

(3.40)

Error function $E_4$ is the number of background pixels in $B$ that have one or more neighboring object pixels in $A$ and no neighboring background pixels in $A$. The probability that a background pixel has a gray value in $B$ equals $\sum_{i=t_1+1}^{t_2} p_2(i)$. The probability that a background pixel has at least one object neighbor in $A$ and no neighboring background pixels in $A$ equals the probability that the pixel has eight
object neighbors which are not all eight in B or C, plus the probability that the pixel has seven object neighbors which are not all in B or C, and one background neighbor that is in B or C, plus the probability that the pixel has six object neighbors which are not all six in B or C, and two background neighbors in B or C, etc. The probability that \( j \) object pixels are not in B or C equals:

\[
1 - \left( \sum_{i=t_1+1}^{N-1} p_1(i) \right)^j
\]  

(3.41)

The probability that \( 8 - j \) background neighbors are in B or C equals:

\[
\left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^{8-j}
\]  

(3.42)

Multiplying these two probabilities gives the probability that \( j \) object pixels are not in B or C and \( 8 - j \) background pixels are in B or C:

\[
\left( 1 - \left( \sum_{i=t_1+1}^{N-1} p_1(i) \right)^j \right) \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^{8-j}
\]  

(3.43)

Multiplying this with the number of background pixels in the image that have \( j \) object neighbors, i.e. \( 8 - j \) background neighbors \( (c_2[8-j]) \), yields:

\[
c_2[8-j] \left( 1 - \left( \sum_{i=t_1+1}^{N-1} p_1(i) \right)^j \right) \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^{8-j}
\]  

(3.44)

Summation over the eight possibilities and multiplying with the probability that the background pixel is in B gives error \( E_4 \):

\[
E_4(t_1, t_2) = \left( \sum_{i=t_1+1}^{t_2} p_2(i) \right)
\]

\[
\sum_{j=1}^{8} c_2[8-j] \left( 1 - \left( \sum_{i=t_1+1}^{N-1} p_1(i) \right)^j \right) \left( \sum_{i=t_1+1}^{N-1} p_2(i) \right)^{8-j}
\]  

(3.45)

Repeated propagation steps

In this section the derivations of the error functions introduced in Section 3.3.5 are described in more detail. Errors \( E_2^1 \) and \( E_5^k \) are trivial and need no further description. Error function \( E_2 \) equals error function \( E_2^{k-1} \) plus the number of background pixels in B which at least one pixel of set \( E_2^{k-1} \) as a neighbor. The probability that a background pixel has a gray value in B equals \( \sum_{i=t_1+1}^{t_2} p_2(i) \). The probability that a pixel is in the set of \( E_2^{k-1} \) equals the number of \( E_2^{k-1} \) divided by the total number of background pixels:

\[
P_{E_2^{k-1}} = \frac{E_2^{k-1}}{MP(C_2)}
\]  

(3.46)
The probability that a pixel has at least one pixel of $E_2^{k-1}$ as a neighbor equals the probability that the pixel has eight background neighbors which are not all eight not in $E_2^{k-1}$ plus the probability that the pixel has seven background neighbors which are not all not in $E_2^{k-1}$, and one object neighbor plus the probability that the pixel has six background neighbors which are not all six not in $E_2^{k-1}$, and two object neighbors, etc. The probability that $j$ background pixels are not all not in $E_2^{k-1}$ equals:

$$1 - (1 - P_{E_2^{k-1}})^j$$  \hspace{1cm} (3.47)

Multiplying this with the number of background pixels in the image that have $j$ background neighbors, i.e. $c_2[j]$, yields:

$$c_2[j] \left(1 - (1 - P_{E_2^{k-1}})^j\right)$$  \hspace{1cm} (3.48)

Summation over the eight possibilities and multiplying with the probability that the background pixel is in $B$ gives error $E_2^k$:

$$E_2^1 = E_2$$

$$E_2^k(t_1, t_2) = E_2^{k-1}(t_1, t_2) + \left(\sum_{i=t_1+1}^{t_2} p_2(i)\right) \sum_{j=1}^{8} \left[c_2[j] \left(1 - (1 - P_{E_2^{k-1}})^j\right)\right]$$  \hspace{1cm} (3.50)

Error function $E_4$ equals error function $E_2^{k-1}$ plus the number of background pixels in $B$ which at least have one pixel of set $E_4^{k-1}$ as a neighbor which is not in set $E_2^{k-1}$. The probability that a background pixel has a gray value in $B$ equals $\sum_{i=t_1+1}^{t_2} p_2(i)$. The probability that a pixel is in the set of $E_4^{k-1}$ equals the number of $E_4^{k-1}$ divided by the total number of background pixels:

$$P_{E_4^{k-1}} = \frac{E_4^{k-1}(t_1, t_2)}{MP(C_2)}$$  \hspace{1cm} (3.51)

The probability that a pixel has at least one pixel of set $E_4^{k-1}$ as a neighbor which is not in set $E_2^{k-1}$ equals the probability that the pixel has eight background neighbors which are not in $E_2^{k-1}$ and not all eight not in $E_4^{k-1}$, plus the probability that the pixel has seven background neighbors which are not in $E_2^{k-1}$ and not all seven not in $E_4^{k-1}$, plus the probability that the pixel has six background neighbors which are not in $E_2^{k-1}$ and not all six not in $E_4^{k-1}$, etc. The probability that $j$ background pixels are not in $E_2^{k-1}$ and not all $j$ not in $E_4^{k-1}$ equals:

$$\left(1 - (1 - P_{E_4^{k-1}})^j\right) \left(1 - P_{E_2^{k-1}}\right)^j$$  \hspace{1cm} (3.52)

Multiplying this with the number of background pixels in the image that have $j$ background neighbors, i.e. $c_2[j]$, yields:

$$c_2[j] \left(1 - (1 - P_{E_4^{k-1}})^j\right) \left(1 - P_{E_2^{k-1}}\right)^j$$  \hspace{1cm} (3.53)
Summation over the eight possibilities and multiplying with the probability that the background pixel is in $B$ gives error $E_4^k$:

$$E_4^1 = E_4$$

$$E_4^k(t_1, t_2) = E_4^{k-1}(t_1, t_2) +$$

$$\left( \sum_{i=t_1+1}^{t_2} p_2(i) \right) \sum_{j=1}^{8} \left[ c_2[j] \left( 1 - (1 - P_{E_4^{k-1}})^j \right) \left( 1 - P_{E_2^{k-1}} \right)^j \right]$$

(3.54)  

(3.55)
Chapter 4

Prevention or correction of binarization errors

This chapter introduces two solutions to the problem of adjoined objects in the binarization of scanned paper documents. One solution is a general pre-processing method operating on the gray-scale image data to prevent binarization errors and is based on estimating the blur of the scanner's lens system. The other solution is a post-processing method operating on the binary image data to correct binarization errors by exploiting the geometrical relationship (topology) between the adjoined objects. Experiments on scanned paper public-utility maps show a high performance for the post-processing method.
4.1 Introduction

Adjoined and smeared objects in the binarization of a scanned paper document is a long-standing problem in the scientific and commercial world of binarization, vectorization, segmentation, recognition, and interpretation of engineering drawings, public-utility maps, music scores, business graphics, business forms, and diagrams, in short document processing which has frequently been reported in the literature [23, 28, 55, 66, 69, 70, 75, 92, 111–113]. Figure 4.1(b) gives an example of adjoined objects; it shows the binarization of a part of the scanned utility map where two digits adjoin. In Figure 4.1(d) one can notice the smearing of a digit 8 in the

![Figure 4.1: Example of adjoined and smeared objects in the binarization. (a) Part of a gray-scale image of a scanned utility map. (b) Binarization. (c) Part of a gray-scale image of a scanned utility map. (d) Image.](image)

binarization of a part of a scanned utility map. Several methods have been proposed in literature to solve the problem of adjoined and smeared objects in the binarization. The methods can be categorized into pre-processing methods and post-processing methods. Pre-processing methods operate on the original gray-scale image data of the scanner and try to prevent the adjoining and smearing of objects during the binarization. Post-processing methods operate on the binary image data and try to correct the adjoined and smeared objects. Both the pre- and the post-processing method can be differentiated further by the type of information they use to prevent or correct adjoining artifacts:

**Syntactic (no model):** Continuity of graphical primitives, for example, arcs, circles, lines, etc.

**Semantic (model):** Recognition of meaningful objects and (geometrical) relationships between these objects. This implies an application-domain model. For example, recognition of characters and digits (OCR), recognition of graphical objects meaningful in a specific application domain, use of contextual information (dictionaries), etc.

Table 4.1 gives a categorization of the proposed methods found in the literature into pre-processing and post-processing methods, and into syntactic and semantic methods. The categorization of methods into syntactic and semantic methods is
Table 4.1: Categorization of methods into pre- and post-processing methods, and into syntactic and semantic methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Syntactic</th>
<th>Semantic</th>
<th>Pre or post</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liang et al. [66]</td>
<td>yes</td>
<td>yes</td>
<td>post</td>
</tr>
<tr>
<td>Rocha et al. [92]</td>
<td>yes</td>
<td>no</td>
<td>pre</td>
</tr>
<tr>
<td>Trier et al. [111]</td>
<td>yes</td>
<td>no</td>
<td>pre</td>
</tr>
<tr>
<td>Den Hartog [23]</td>
<td>yes</td>
<td>yes</td>
<td>pre</td>
</tr>
<tr>
<td>Mullot et al. [75]</td>
<td>yes</td>
<td>no</td>
<td>post</td>
</tr>
<tr>
<td>Trier et al. [113]</td>
<td>yes</td>
<td>no</td>
<td>pre and post</td>
</tr>
</tbody>
</table>

important because semantic methods use a model of the objects and their (geometrical) relationships and are therefore not generally applicable to other application domains. In our opinion, none of the proposed methods in the literature touches upon the essence of the cause of adjoining artifacts. This cause is twofold:

1. The objects already physically adjoin on the original paper document in the sense that the ink of the adjoining objects has merged completely in one particular spot.

2. The adjoined objects do not physically adjoin on the original paper document, but the blank space that separates the objects is narrow. As a consequence, adjoining artifacts may occur due to blurring in the scanning process or due to imperfection of the binarization method. This cause of adjoining objects is visualized in Figure 4.2.

![Figure 4.2: The small part of background that separates the digits 9 and 8 is lost in the binarization (global thresholding).](image)

The adjoining artifacts of cause 1 are in fact not even to be called artifacts because the gray-scale input data is already imperfect. Nevertheless, in the remainder we shall consider these as artifacts. The differentiation of the cause is important because it reveals that for cause 1 adjusting the settings of the scanner or refining the binarization method is not sufficient to solve the problem of adjoined objects. Hence, pre-processing methods cannot repair the adjoining artifacts of cause 1. On the other hand, post-processing methods may be able to repair adjoining artifacts caused by both causes 1 and 2, as they work on the binary image data and not on the gray-scale image data.
### Utility map

<table>
<thead>
<tr>
<th>Utility map</th>
<th># adjoined</th>
<th># smeared</th>
<th>% cause 1</th>
<th>% cause 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1272w</td>
<td>38</td>
<td>2</td>
<td>37%</td>
<td>63%</td>
</tr>
<tr>
<td>R1273q</td>
<td>89</td>
<td>1</td>
<td>49%</td>
<td>51%</td>
</tr>
<tr>
<td>R1318c</td>
<td>36</td>
<td>0</td>
<td>39%</td>
<td>61%</td>
</tr>
<tr>
<td>R1318e</td>
<td>192</td>
<td>0</td>
<td>46%</td>
<td>54%</td>
</tr>
</tbody>
</table>

Table 4.2: Number of artifacts and their cause for four different utility maps. The second and third column depict the number of adjoined and smeared objects found in the binarisations. The fourth and fifth column depict the percentage of the number of artifacts that are due to cause 1 and 2.

In our utility-map application domain, the biggest problem of adjoined objects is the problem of two adjoined digits in a (dimension) number (see Figure 4.1(b)). Table 4.2 gives the number of adjoined objects for a set of utility maps. Our utility maps have size A0 and are scanned at 400 dpi. After the utility maps have been scanned, they are binarized by a global thresholding with hysteresis, as introduced in Chapter 3. From Table 4.2 we may conclude that the number of smeared objects is negligible and that the majority of adjoined objects are due to cause 2. For the artifacts due to cause 2, a pre-processing method is proposed that takes into account the amount of blurring of the scanner and further detects small parts of background that separate the adjoined objects. The advantage of this syntactic pre-processing method is that it is generally applicable and does not require any model of the objects or the geometrical relationships between the objects. The pre-processing method is described in Section 4.2. For the remaining artifacts due to causes 1 (and 2), a post-processing method is proposed that requires a model of the geometrical relationship between the adjoining objects but can correct adjoined objects due to both causes 1 and 2. The semantic post-processing method is described in Section 4.3. Finally, in Sections 4.4, 4.5, and 4.6 some experiments are described and conclusions are drawn.

### 4.2 Pre-processing method

This section describes our proposed pre-processing method. First, a model of a scanner is introduced in Section 4.2.1. After that, the morphological top-hat transformation is defined in Section 4.2.2. In Section 4.2.3, we describe the detection of small parts of background in the gray-scale image data by the top-hat transformation, given the scanner model. Finally, in Section 4.2.4, it is shown how the detected parts of background are used to prevent the occurrence of adjoined objects in the binarization result. Figure 4.3 gives a schematic view of our proposed pre-processing method.

#### 4.2.1 Scanner model

The scanning process is considered to be a blurring and discretizing, i.e. sampling and quantization, of the original paper document. The resulting image, \( f(x) \) can
thus be represented by:

\[ f(x) = i(x) * h(x) \]  

(4.1)

where function \( i(x) \), \( i : x \in \mathcal{R}^2 \mapsto i(x) \in \mathcal{R} \) is the original image, \( h(x) \), \( h : x \in \mathcal{R}^2 \mapsto h(x) \in \mathcal{R} \) the point-spread function (PSF) of the scanning system function \( f(x) \), \( f : x \in \mathcal{Z}^2 \mapsto f(x) \in \mathcal{Z} \) is the resulting image, \('*' represents the convolution operator. In the remainder of this chapter, it is assumed that the function \( h(x) \) is known (or at least estimated [40]) and is a Gaussian function that is isotropic in origin \((\mu = 0)\), i.e.:

\[ h(x) = G(x;\mu,\sigma) = G(x;0,\sigma) \]  

(4.2)

The parameter \( \sigma \) is the standard deviation of the Gaussian function.

### 4.2.2 Definition of morphological top-hat transformation

The morphological top-hat transformation [73, 103] is a well-known and commonly used morphological technique for extracting locally bright (positive contrast) objects from a gray-scale image, based on size and relatively brightness [64, 119]. The morphological top-hat transformation is defined as:

\[ \{ f(x) - (f \circ g)(x) \}_t \]  

(4.3)

The image operator \((f \circ g)(x)\) is a morphological gray-scale opening operator and \( \{ f(x) \}_t \) is a threshold set of gray-scale image \( f(x) \) that is defined as:

\[ \{ f(x) \}_t = \{ x \mid f(x) \leq t \} \]  

(4.4)

Parameter \( t, t \in [0, N - 1] \) is the threshold level. A threshold set is thus a set of pixels \( x \in \mathcal{Z}^2 \) that equals a binary image, in which each pixel has a binary value that indicates whether that pixel is an element of set \( S \) or not. We define the opening operator by first applying a morphological gray-scale dilation image operator and then a morphological gray-scale erosion image operator:

\[ (f \circ g)(x) = ((f \ominus g) \oplus g)(x) \]  

(4.5)

The dilation operator is defined as:

\[ (f \oplus g)(x) = \bigvee_{u \in \mathcal{Z}^2} [f(u) + g(x - u)] \]  

(4.6)
Similarly, the erosion operator is defined as:

\[(f \ominus g)(x) = \bigwedge_{u \in \mathbb{Z}^2} \{f(u) - g(u - x)\}\]  \hspace{1cm} (4.7)

In Equations 4.3 to 4.7, function \(f(x)\), \(f : x \in \mathbb{Z}^2 \mapsto f(x) \in \mathbb{Z}\) is the original image and \(g(x)\), \(g : x \in \mathbb{Z}^2 \mapsto g(x) \in \mathbb{Z}\) is the structuring function. For a top-hat transformation, a flat structuring function is commonly used as structuring function \(g(x)\), which is defined as a cylinder specified by a radius \(\rho \in \mathbb{N}\):

\[g^\rho(x) = \begin{cases} 0, & x \in S^\rho \\ -\infty, & x \not\in S^\rho \end{cases}\]  \hspace{1cm} (4.8)

In Equation 4.8, \(S^\rho\) is a set of pixels \(x \in \mathbb{Z}^2\) that equals a discrete approximation of a disk of radius \(\rho\). As a result, the top-hat transformation of Equation 4.3 is finally parameterized by radius \(\rho\) and threshold \(t\) (to select a set). Figure 4.4 gives an example of the application of the top-hat transformation. The top-hat transformation defined by Equation 4.3 is known as the white top-hat transformation. The black top-hat transformation, capable of extracting locally dark (negative contrast) objects, is defined with a morphological gray-scale closing operator \((f \circ g)(x)\):

\[\{f(x) - (f \circ g)(x)\}_t\]  \hspace{1cm} (4.9)

The closing operator is defined similar to the opening operator, but the order of operations is reversed:

\[(f \circ g)(x) = ((f \ominus g) \Theta g)(x)\]  \hspace{1cm} (4.10)

Because the complexity of the opening operator equals \(O(\rho n)\), where \(n\) is the number of pixels, and the complexity of the threshold operator \(\{\cdot\}_t\) and the subtraction operation are both \(O(n)\), the complexity of the top-hat transformation equals \(O(\rho n)\). Consequently, as long as \(\rho \ll n\), the top-hat transformation has a linear complexity to the number of pixels in the image.

![Figure 4.4: Consecutive steps in a white top-hat transformation.](image-url)

(a) Part of a gray-scale image of a scanned utility map. (b) Morphological gray-scale opening with structuring function \(g^4(x)\). (c) Subtraction of opening result with original image. (d) Threshold set of subtraction result with threshold level \(t = 14\).
4.2.3 Detection of small background parts

In our application, the objects of interest are black text and drawn objects on a white paper background. Adjoining artifacts (of cause 2) occur when a small part of the white background is surrounded by (large) black objects. Due to the scanner characteristics the surrounding black objects may merge over the white background part. The white top-hat transformation can be used to detect these small white parts in order to prevent that the adjoining artifacts occur. In the one-dimensional case the small white part can be modeled by a continuous function \( i(x) \) that has a small range (of width \( d \)) of high (white) gray values, whereas outside this range (surrounding objects) the gray values are low (black). If we take the scanner model of Section 4.2.1, the scanned version of this signal is by (Equations 4.1 and 4.2):

\[
f(x) = \int i(x)h(x - y)dy = \int i(x)G(x - y, x, \sigma)dy \tag{4.11}
\]

The width of the scanned function \( f(x) \) is now defined as the width of the range of samples that have a non-zero gray value, whereas the height is defined as the maximum gray value. Both are illustrated in Figure 4.5. As an example, consider

![Figure 4.5: Width and height of a small blurred part of background.](image)

an original signal, \( i(x) \), having a certain width and a height of 255 (the maximum gray value). The width is set in such a way that on the average scanning \( i(x) \) results in the PSP \( h(x) \) of the scanner. Further, assume that the scanning characteristic, \( h(x) \), equals \( G(x; \mu = 0, \sigma = 1) \). Then we can show the scanned signal \( f(x) \) in Figure 4.6(a).

The width and height of the original signal \( i(x) \) can be estimated by applying the top-hat transformation to \( f(x) \). Different settings of the top-hat transformation (i.e. parameters \( \rho \) and \( t \)) lead to different estimates, as shown in Figure 4.6(b). Note that in order to find the correct width, a different threshold has to be chosen for a different choice of radius \( \rho \). By choosing radius \( \rho \) equal to \( w/2 + 1 \), with parameter \( w \) equaling the width of function \( f(x) \), we obtain in the opening residue all samples of function \( f(x) \) that have a non-zero gray value (\( \rho = 3 \) in Figure 4.6(b)). By consequently setting the threshold value \( t \) in between the maximum and second maximum (\( t = 82 \), Figure 4.6(b)) then we detect functions \( f(x) \) which have a width of at least \( w \) and a height of at least \( t \).

If the amount of blur present in the image (\( \sigma_{\text{blur}} \)) is known, and the average background gray value (\( \mu_B \)), the values of radius \( \rho \) and threshold \( t \) can automati-
<table>
<thead>
<tr>
<th>$\sigma_{\text{blur}}$</th>
<th>0.6</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>1.8</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>width of function $f(x)$</td>
<td>5</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>9</td>
<td>9</td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td>height of function $f(x)$</td>
<td>106</td>
<td>93</td>
<td>82</td>
<td>72</td>
<td>64</td>
<td>58</td>
<td>52</td>
<td>48</td>
</tr>
<tr>
<td>radius $\rho$</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>threshold $t$</td>
<td>106</td>
<td>93</td>
<td>82</td>
<td>72</td>
<td>64</td>
<td>58</td>
<td>52</td>
<td>48</td>
</tr>
</tbody>
</table>

Table 4.3: Width and height of function $f(x)$ plus selections for the parameters radius $\rho$ and threshold $t$ of the top-hat transformation for different values of $\sigma_{\text{blur}}$. Function $f(x)$ is the blurred (by the amount of $\sigma_{\text{blur}}$) and discrete version of function $i(x)$ that is shown in Figure 4.6(a).

![Diagram](image)

Figure 4.6: Detection of function $f(x)$ samples with different values for radius $\rho$ and threshold $t$. (a) Original signal $i(x)$ and scanned functions $f(x)$. (b) Opening $(f \circ g)(x)$ and opening residue $f(x) - (f \circ g)(x)$.

cally be set based upon the above procedure and the convolution integral given in Equation 4.11. Table 4.3 lists the settings for radius $\rho$ and threshold $t$ for different values of $\sigma_{\text{blur}}$ and $\mu_B = 255$. The corresponding values of the width and height of function $f(x)$ are also listed. Further, if necessary, the threshold value $t$ can be set to a lower value in order to detect white parts of functions $i(x)$ having a smaller width.

In most document-processing applications, the two objects surrounding the small part of background have the same width as the background part. For example, in the case of two adjoining characters, the width of the characters is almost the same as the distance between the characters. Due to the scanner characteristics, the gray values of small objects become higher, see Figure 4.7. In order to detect the background in between these objects, we have to take the higher gray values of the objects into account and must lower the threshold settings accordingly.
Table 4.4: Threshold selections for the top-hat transformation for different values of $\sigma_{\text{blur}}$ and for different values of the width of the two surrounding objects (in pixels). The value $-1$ means that no threshold can be selected to estimate the width of $f(x)$ because all values became zero due to blurring.

<table>
<thead>
<tr>
<th>$d_0$</th>
<th>0.6</th>
<th>0.8</th>
<th>1.0</th>
<th>$\sigma_{\text{blur}}$</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>1.8</th>
<th>2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>86</td>
<td>16</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>127</td>
<td>63</td>
<td>27</td>
<td>7</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>3</td>
<td>127</td>
<td>69</td>
<td>39</td>
<td>21</td>
<td>10</td>
<td>3</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>4</td>
<td>127</td>
<td>69</td>
<td>40</td>
<td>25</td>
<td>15</td>
<td>9</td>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>127</td>
<td>69</td>
<td>40</td>
<td>25</td>
<td>16</td>
<td>11</td>
<td>7</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>127</td>
<td>69</td>
<td>40</td>
<td>25</td>
<td>16</td>
<td>11</td>
<td>8</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>127</td>
<td>69</td>
<td>40</td>
<td>25</td>
<td>16</td>
<td>11</td>
<td>8</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>127</td>
<td>69</td>
<td>40</td>
<td>25</td>
<td>16</td>
<td>11</td>
<td>8</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

Figure 4.7: Small objects that surround a background part imply a lower threshold selection.

Consider, for example, a function $i(x)$ that has a background part with the same width as above and maximum gray value which is surrounded by two equally small objects having a zero gray value. All other values of function $i(x)$ are equal to the maximum gray value 255. Function $i(x)$ is shown in Figure 4.7. Table 4.4 gives the threshold settings for the top-hat transformation for different values of $\sigma_{\text{blur}}$ and for different values of the width ($d_0$) of the two objects. The settings of the radius parameter $\rho$ remain the same, as listed in Table 4.3.

4.2.4 Prevention of binarization errors

The results of the detection of small background parts with the top-hat transformation are used to prevent binarization errors. The threshold set of pixels that results from the application of the top-hat transformation is regarded as a set of pixels that definitely belong to the background. Thus the resulting background pixels af-
ter binarization are merged (using bitwise OR) with the set of background pixels that are detected by the top-hat transformation. An example of using the top-hat transformation to prevent binarization errors is given in Figure 4.8.

![Figure 4.8: Preventing binarization errors with detection results of the top-hat transformation. (a) Part of a gray-scale image of a scanned utility map. (b) Detection results of the top-hat transformation. (c) Binarization result. (d) Logical bitwise OR of the top-hat result and the binarization.](image)

4.3 Post-processing method

This section describes our proposed post-processing method. The method works solely on the binary image data and corrects adjoining artifacts. The method generates possible decompositions into two disconnected of adjoined objects. Each decomposition is evaluated by judging the geometrical relationship between the two disconnected objects. In order to judge such a relationship, it is necessary to have a model of the objects as well as a model of their geometrical relationship. Both these models highly depend on the application domain. Here, for the application domain of binarizing utility maps, a model of two adjoined digits that are written next to each other is chosen (most frequent occurrence of adjoining artifact). An example is shown in Figure 4.1(a). The judgment functions for the geometrical relationships are learned from examples using pattern-recognition techniques.

The outline of this section is as follows: Section 4.3.1 describes how object decompositions are generated, Section 4.3.2 discusses the judgments of the decompositions, and Section 4.3.3 presents the general algorithm to accomplish the post-processing.

4.3.1 Generation of topological decompositions

This section describes how to disconnect adjoining objects by removing branch points from their skeleton - it always decomposes into two objects. Furthermore, it shows how to reconstruct the disconnected objects from the resulting skeletons. The proposed decomposition method is a restricted version of the method for decomposing graphics into graphical primitives described by Den Hartog in [25]. It is restricted in the sense that not all branch points are removed, but only a restricted number.
4.3 Post-processing method

Let the function $d(x)$, $d : x \in \mathbb{Z}^2 \mapsto d(x) \in \mathcal{N}$ represent the pseudo-Euclidean distance from an object pixel $x$ to the background, and let $C$, a set of pixels $x \in \mathbb{Z}^2$ denote a connected component. The pseudo-Euclidean distance skeleton $S$ of component $C$ is defined as the smallest set of connected pixels that satisfies

$$\bigcup_{x \in S} D(x, d(x)) = C$$  \hspace{1cm} (4.12)

where $S$ is a set of pixels $x \in \mathbb{Z}^2$, and $D(c, r)$ is a set of pixels $x \in \mathbb{Z}^2$, together forming a (discrete) disk with center $c$ and radius $r$. $D(c, r)$ is defined as

$$D(c, r) = \{x \mid (c_0 - x_0)^2 + (c_1 - x_1)^2 \leq r^2\}$$  \hspace{1cm} (4.13)

Thus, a skeleton of a component compromises a minimal set of connected pixels in such a way that the union of the disks centered on them (with radii equal to the distance to the background) reconstructs the component [25, 74]. An example of a pseudo-Euclidean distance skeleton is shown in Figures 4.9(a) and 4.9(b). The set of branch points $B$ of a skeleton $S$ are defined as these pixels that have more than two neighbors:

$$B = \{x \mid x \in S \land \# \{y \mid y \in S \land y \in N(x)\} \geq 3\}$$  \hspace{1cm} (4.14)

$B$ is a set of pixels $x \in \mathbb{Z}^2$, whereas $\#\{\cdot\}$ determines the length of a set, and $N(x)$ is a set of pixels $x \in \mathbb{Z}^2$ that are 8-connected neighbors of pixel $x \in \mathbb{Z}^2$. $N(x)$ is defined as:

$$N(x) = \{(x_0 - 1, x_1 - 1), (x_0 - 1, x_1), (x_0 - 1, x_1 + 1), (x_0, x_1 - 1), (x_0, x_1 + 1), (x_0 + 1, x_1 - 1), (x_0 + 1, x_1), (x_0 + 1, x_1 + 1)\}$$  \hspace{1cm} (4.15)

For example, the skeleton of the digit 3 shown in Figure 4.9(b) has one branch point, and the skeleton of the two adjoined digits shown in Figure 4.9(d) has three branch points. We state, without proof\footnote{In the experiments we did not encounter any counter example.}, that the number of branch points of the pseudo-Euclidean distance skeleton of two adjoined digits is at least one higher than the
sum of the number of branch points of the skeletons of the two disconnected digits. Two adjoined digits can be decomposed into two disconnected objects by removing the appropriate branch points of the skeleton of the two digits. A branch point \( x \) is removed from a pseudo-Euclidean skeleton \( S \) by removing one of its neighbors \( y \in N(x) \cap y \in S \) from \( S \). Note that each branch point \( x \) has at least three neighbors \( y \in S \). For instance, the two adjoined digits, as shown in Figure 4.9(c), can be repaired by removing one of the two branch points introduced by the adjoinment of the two digits. To decompose an object, only those branch-point removals are considered that result in two disconnected pieces \( S_1 \) and \( S_2 \) of the skeleton \( S \). The two disconnected objects can be reconstructed by applying the definition of the pseudo-Euclidean distance skeleton (Equation 4.12):

\[
C_1 = \bigcup_{x \in S_1} D(x, d(x))
\]

\[
C_2 = \bigcup_{x \in S_2} D(x, d(x))
\]

Figure 4.10 shows different decompositions of the pseudo-Euclidean distance skeleton of Figure 4.9(d) by possible removals of one branch point. Additionally, the reconstructed digits are shown. The selection of the right decomposition is the topic of the next section. The adjoined digits of Figure 4.9(c) can in principle be repaired by removing one branch point from the skeleton. Unfortunately, not all occurrences of adjoined digits can be repaired by removing only one branch point. For instance, to decompose the adjoined digits shown in Figure 4.11(b), it is necessary to remove two branch points simultaneously. In this case, an additional problem arises: one of the components misses a part of the skeleton that is shared by both components. One solution for this problem is to copy the part of the skeleton between the two removed branch points and to paste it in the component missing it. The experiments will show that this is a suboptimal solution. For the majority of adjoined digits, copying this shared part of the skeleton is the correct procedure; the remaining cases are labeled as errors in the experiments. Besides adjoined digits that require the simultaneous removal of one or two branch points, there are also adjoined digits that require the simultaneous removal of three or more branch points. Table 4.5 shows the percentage of the required number of branch points that must be removed simultaneously for adjoined digits that occur in the binarization of a number of utility maps. Their frequency of occurrence is relatively low and therefore we do not discuss their decomposition in detail; it can be based upon repeatedly removing one or two branch points simultaneously.

![Figure 4.10](image-url)
4.3 Post-processing method

Figure 4.11: (a) Two adjoining digits zero. (b) Pseudo-Euclidean distance skeleton and branch points. (c) (d) Decompositions (in black) and reconstructions (in gray).

<table>
<thead>
<tr>
<th>Utility map</th>
<th>1</th>
<th>2</th>
<th>3 or more</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1272w</td>
<td>80%</td>
<td>17%</td>
<td>3%</td>
</tr>
<tr>
<td>R1273q</td>
<td>54%</td>
<td>34%</td>
<td>12%</td>
</tr>
<tr>
<td>R1318c</td>
<td>58%</td>
<td>39%</td>
<td>3%</td>
</tr>
<tr>
<td>R1318e</td>
<td>61%</td>
<td>31%</td>
<td>8%</td>
</tr>
</tbody>
</table>

Table 4.5: The percentage of the required number of branch points that must be removed simultaneously for adjoined digits that occur in the binarization of a number of utility maps.

4.3.2 Judgment of topological decompositions

Having discussed the generation of topological decompositions of adjoined digits in the previous section, we now deal with the judgment of the different decompositions. Each decomposition is evaluated by judging the geometrical relationship between the two components after the decomposition. In order to judge the geometrical relationship between the two components it is necessary to have a model of their relationship. The geometrical model adopted in this section is based upon two neighboring digits that have a similar size. From both neighboring objects a number of features is derived that are used to learn by examples the judgment function of the relationship by pattern-recognition techniques. Concluding, for each decomposition generated from the adjoined objects a feature vector is calculated with which the geometrical relationship between the two components is judged.

To judge the geometrical relationship five features are considered that are based on the Minimum-Area Encasing Rectangle (MAER) [34] of the two components. The minimum-area encasing rectangle of an object is the smallest rectangle that can be fitted around the object. Two examples of a MAER are given in Figure 4.12. We take the width of a MAER as the smallest side of the rectangle, and the height of a MAER as the longest side of the rectangle. The five features of the geometrical relationship measure differences in position, orientation, width, and height of the MAERs of the two components. The feature vector \([dx, dy, \alpha, dw, dh]\) is defined by the following definition of the features (an example is shown in Figure 4.12):
Figure 4.12: The representation of a geometrical relationship between two objects uses the minimum-area encasing rectangles of the objects.

\( dx \): translation along the x-axis (in pixels) between the points of gravity of the MAER of object 1 and the MAER of object 2.

\( dy \): translation along the y-axis (in pixels) between the points of gravity of the MAER of object 1 and the MAER of object 2.

\( \alpha \): difference in orientation angle (in radians) of the MAER of object 1 and the MAER of object 2. The difference in orientation angle is taken as the angle between the longest side of the MAER of object 2 and the shortest side of the MAER of object 1.

\( dw \): difference in width (in pixels) of the MAER of object 1 and the MAER of object 2.

\( dh \): difference in height (in pixels) of MAER of object 1 and the MAER of object 2.

After having defined the feature vector to represent the geometrical relationship between the two neighboring digits, we can learn the judgment functions to judge different decompositions using pattern-recognition techniques. These classifiers take a feature vector \([dx, dy, \alpha, dw, dh]\) as input and give the judgment value (JV) that expresses a measure of belief (or confidence) in the possibility of the decomposition as output. A JV of zero represents a wrong decomposition, and a JV of one represents a correct decomposition of two adjoined digits into two neighboring digits. Because an adjoined digit can be decomposed in multiple decompositions, each of them is judged by the classifier and is consequently labeled with a JV. After all decompositions have been judged the decomposition that has the highest JV is chosen.
as decomposition of the adjoined object. The procedure of correcting adjoined digits is described in more detail in Section 4.3.3. The classifiers for the judgment of a possible decomposition are trained from correctly labeled (JV zero or one) examples of decompositions. Section 4.5 describes in more detail which classifiers were used and shows results of experiments with the post-processing method on binarization errors in the binarization of a number of utility maps.

4.3.3 Algorithm for the post-processing method

The algorithm to perform the post-processing method incorporates the generation and judgment of decompositions of adjoined objects, as discussed in the previous sections. Algorithm 3 describes the general procedure of correcting adjoined objects in the binarization of a document. For simplicity, Algorithm 3 only considers decompositions made by the removal of one branch point. The algorithm starts by

```
Algorithm 3 General algorithm for the correction of adjoined objects.

Require: I is a binary image that contains an adjoinment of two objects and judge_decomposition() is a trained judgment function.

jvmax ← 0
C1max ← Ø
C2max ← Ø
{determine pseudo-Euclidean skeleton}
D ← distance(I)
S ← pseudo-Euclidean.skeleton(I, D)
{determine branch points of skeleton}
B ← branch_points(S)
{process branch points}
for i ← 1 to size(B) do
    {generate decompositions by removing branch point i}
    C1,2 ← generate_decompositions(S, i)
    {judge decompositions}
    for j ← 1 to size(C1) do
        jv ← judge_decomposition(C1(j), C2(j))
        if jv > jvmax then
            {save decomposition}
            jv ← jvmax
            C1max ← C1(j)
            C2max ← C2(j)
        end if
    end for
end for
```

determining the pseudo-Euclidean distance skeleton of the binary image I. Afterwards, the branch points of the skeleton are determined. For each branch point i, the decompositions of the adjoined object are generated by removing branch point i and judged afterwards. If the JV of a decomposition is higher than the JV of any other decomposition that was judged earlier, then the decomposition is kept. The
algorithm ends after all branch points have been processed. If the algorithm has finished, the variables \( \{C_1, C_2\}^{\max} \) store the decomposition with the highest JV.

### 4.4 Pre-processing method: experiments

In this section, we carry out experiments on test images with the pre-processing method, which uses the white top-hat transformation. We have performed experiments with the top-hat transformation on artificially generated images and on images obtained by scanning paper utility maps. The experiments are described in Sections 4.4.1 and 4.4.2, respectively.

#### 4.4.1 Experiments with artificial images

The artificially generated image (shown in Figure 4.13) consists of the numbers 0 to 99. Each number is represented with two digits in a 32-points font. The width of the digits is five pixels on average. The two digits are at least one pixel apart. We degrade the image by blurring the image with a Gaussian blur of \( \sigma_{\text{blur}} \) and by adding additive Gaussian noise of \( \sigma_{\text{noise}} \). The thus artificially generated image is binarized by applying a global threshold of \( t = 128 \). The parameters of the top-hat transformation are set according to the values in Table 4.4. Table 4.6 shows the percentage of the number of unconnected digits that appear in the binarization result without applying the pre-processing method. A percentage of 100% thus means that no objects are adjoined. The percentage is depicted for different degradations of the image. Table 4.7 shows the same percentage when the pre-processing method is applied. The number of false positive preventions is given in Table 4.8.

From Tables 4.6, 4.7, and 4.8 we may conclude that the pre-processing method can successfully prevent objects that adjoin in the binarization result. For \( \sigma_{\text{blur}} \leq 1.0 \), the pre-processing method can prevent all occurrences of digits that adjoin. The noise does not influence the results of the experiments. For higher values of \( \sigma_{\text{blur}} \),
4.4 Pre-processing method: experiments

<table>
<thead>
<tr>
<th>$\sigma_{\text{noise}}$</th>
<th>0.6</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>1.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>100%</td>
<td>86%</td>
<td>76%</td>
<td>72%</td>
<td>72%</td>
<td>68%</td>
<td>66%</td>
</tr>
<tr>
<td>2.0</td>
<td>100%</td>
<td>88%</td>
<td>76%</td>
<td>74%</td>
<td>72%</td>
<td>68%</td>
<td>67%</td>
</tr>
<tr>
<td>3.0</td>
<td>100%</td>
<td>86%</td>
<td>76%</td>
<td>74%</td>
<td>71%</td>
<td>68%</td>
<td>67%</td>
</tr>
<tr>
<td>4.0</td>
<td>100%</td>
<td>85%</td>
<td>76%</td>
<td>74%</td>
<td>71%</td>
<td>68%</td>
<td>67%</td>
</tr>
<tr>
<td>5.0</td>
<td>100%</td>
<td>85%</td>
<td>76%</td>
<td>74%</td>
<td>70%</td>
<td>68%</td>
<td>67%</td>
</tr>
<tr>
<td>6.0</td>
<td>100%</td>
<td>85%</td>
<td>76%</td>
<td>74%</td>
<td>70%</td>
<td>68%</td>
<td>66%</td>
</tr>
<tr>
<td>7.0</td>
<td>100%</td>
<td>84%</td>
<td>76%</td>
<td>74%</td>
<td>70%</td>
<td>68%</td>
<td>66%</td>
</tr>
<tr>
<td>8.0</td>
<td>100%</td>
<td>84%</td>
<td>76%</td>
<td>74%</td>
<td>70%</td>
<td>68%</td>
<td>66%</td>
</tr>
</tbody>
</table>

Table 4.6: Percentage of number of unconnected digits in the binarization of the image in Figure 4.13 when no pre-processing method is applied. The percentage is depicted for different values of $\sigma_{\text{blur}}$ and $\sigma_{\text{noise}}$.

<table>
<thead>
<tr>
<th>$\sigma_{\text{noise}}$</th>
<th>0.6</th>
<th>0.8</th>
<th>1.0</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>1.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>96%</td>
<td>96%</td>
<td>96%</td>
<td>100%</td>
</tr>
<tr>
<td>2.0</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>96%</td>
<td>96%</td>
<td>98%</td>
<td>100%</td>
</tr>
<tr>
<td>3.0</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>96%</td>
<td>96%</td>
<td>98%</td>
<td>100%</td>
</tr>
<tr>
<td>4.0</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>96%</td>
<td>96%</td>
<td>98%</td>
<td>100%</td>
</tr>
<tr>
<td>5.0</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>96%</td>
<td>96%</td>
<td>98%</td>
<td>100%</td>
</tr>
<tr>
<td>6.0</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>96%</td>
<td>96%</td>
<td>98%</td>
<td>100%</td>
</tr>
<tr>
<td>7.0</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>96%</td>
<td>96%</td>
<td>98%</td>
<td>100%</td>
</tr>
<tr>
<td>8.0</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>96%</td>
<td>96%</td>
<td>98%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Table 4.7: Percentage of number of unconnected digits in the binarization of the image in Figure 4.13 when the pre-processing method is applied. The percentage is depicted for different values of $\sigma_{\text{blur}}$ and $\sigma_{\text{noise}}$.

the performance degrades, and the number of false positive preventions of adjoined digits increases.

4.4.2 Experiments with images of scanned utility maps

This section shows the results of the pre-processing method applied to scanned utility maps. The scanner used is the ANA Tech Eagle 3640 scanner [4]. The ANA Tech scanner scans documents which can have a size of A0 by sampling at 400dpi and quantizing the samples to 256 gray values. We estimated the PSF of the scanner by scanning an IEEE Std 167A-1987 facsimile test chart [5]. This facsimile test chart has a number of small white dots on a black background. The scanned and blurred versions of these white dots are approximations of the PSF of the lens system of the scanner. Averaging over a number of blurred dots and fitting the average dot into the Gaussian PSF of our scanner model gives a Gaussian PSF with $\sigma = 1.0$. The obtained PSF is found to be isotropic: $\sigma_{x_0} = \sigma_{x_1} = 1.0$. Because
Table 4.8: Number of false positive preventions of digits that adjoin in the binarization of the image in Figure 4.13 when the pre-processing method is applied.

<table>
<thead>
<tr>
<th>$\sigma_{\text{noise}}$</th>
<th>0.6</th>
<th>0.8</th>
<th>1.0</th>
<th>$\sigma_{\text{blur}}$</th>
<th>1.2</th>
<th>1.4</th>
<th>1.6</th>
<th>1.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>3.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>4.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>5.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>36</td>
<td></td>
</tr>
<tr>
<td>6.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>37</td>
<td></td>
</tr>
<tr>
<td>7.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>8.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>33</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.9: Results of experiments with the pre-processing method on scanned utility maps. The second column lists the percentage of the number of adjoined objects that are correctly prevented by the application of the pre-processing method for a number of different utility maps. The third column lists the number of false positive preventions. The fourth column lists the sample size for each map.

<table>
<thead>
<tr>
<th>Utility map</th>
<th>% correct</th>
<th># fp</th>
<th># samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1272w</td>
<td>61%</td>
<td>1</td>
<td>23</td>
</tr>
<tr>
<td>R1273q</td>
<td>82%</td>
<td>4</td>
<td>45</td>
</tr>
<tr>
<td>R1318c</td>
<td>86%</td>
<td>3</td>
<td>22</td>
</tr>
<tr>
<td>R1318e</td>
<td>76%</td>
<td>4</td>
<td>104</td>
</tr>
</tbody>
</table>

The 'paper' (actually linen) of the utility maps is not truly white and because the ink used to draw the text and graphics on the utility map is not exactly black, it is necessary to calculate the average object gray value and average background gray value from the gray-scale image that was obtained from scanning the map. This is done by selecting the object pixels using a global threshold technique and then averaging their gray value. The average background gray value is calculated in a similar fashion. For the experiments we have used a number of scanned utility maps and their corresponding binarizations. The binarization is a sequence of a deblurring step (Chapter 2), and a thresholding with hysteresis (Chapter 3). To evaluate the pre-processing, small parts were taken from the binary images of the utility maps that contain adjoined objects (of course 2). On the corresponding gray-scale-image sections the pre-processing method has been applied. The pre-processing method is successful if it prevents the objects to adjoin in the binarization. We have also counted the number of false positive preventions, i.e. objects that were divided by the pre-processing method (because small background parts were found there).

Table 4.9 shows the results of the experiments with the pre-processing method on four different utility maps. The parameters of the top-hat transformation of the pre-processing method are set according to Table 4.4. Figure 4.14 gives the receiver operating characteristic (ROC) curves if we omit the threshold selection of the top-
hat transformation. Because the threshold settings of Table 4.4 are based on an estimation of the blur present in the image, the optimal value for threshold \( t \) of the top-hat transformation may vary from the value in Table 4.4. The ROC curves give us more insight in the performance of the pre-processing method for variations of the threshold setting.

![ROC curves](image)

**Figure 4.14**: Results of experiments with the pre-processing method on scanned utility maps: receiver operating characteristic curves for utility maps R1272w, R1273q, R1318c, and R1318e. The vertical axis denotes the fraction of the number of adjoined objects that are correctly prevented by the application of the pre-processing method. The horizontal axis denotes the number of false positive preventions. (a) ROC R1272w. (b) ROC R1273q. (c) ROC R1318c. (d) ROC R1318e.

From the results in Table 4.9 we may conclude that the pre-processing method can correctly prevent about 75\% of the binarization errors due to objects that adjoin in the binarization and involves a low number of false positive detections. Figure 4.14 shows that a less conservative threshold selection of the top-hat transformation can increase performance of the pre-processing method considerably, if we can accept more false positive preventions.
<table>
<thead>
<tr>
<th>Utility map</th>
<th>Linear</th>
<th>Quadratic</th>
<th>Parzen</th>
<th>ANN</th>
<th>k-NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1272w</td>
<td>100%</td>
<td>93%</td>
<td>93%</td>
<td>100%</td>
<td>87%</td>
</tr>
<tr>
<td></td>
<td>15%</td>
<td>3%</td>
<td>8%</td>
<td>3%</td>
<td>8%</td>
</tr>
<tr>
<td>R1318c</td>
<td>94%</td>
<td>94%</td>
<td>94%</td>
<td>100%</td>
<td>94%</td>
</tr>
<tr>
<td></td>
<td>9%</td>
<td>10%</td>
<td>6%</td>
<td>4%</td>
<td>6%</td>
</tr>
</tbody>
</table>

Table 4.10: Percentage of correctly repaired adjoined digits for different utility maps by application of Algorithm 3 (first row). Additionally, classification error for all decompositions (second row).

4.5 Post-processing method: experiments

This section shows the results of the application of our post-processing method to correct adjoining artifacts in scanned utility maps. Furthermore, results are shown for different pattern-recognition techniques used to learn the judgment functions for the decomposition of the adjoined objects. The utility maps used in the experiments are scanned and binarized in the same manner as described in Section 4.4.2; the preprocessing method is not applied. Again, all adjoined digits are collected, and for each of them we manually labeled all possible decompositions as correct \((J_V = 1)\) or not correct \((J_V = 0)\). Each possible decomposition is generated according to the procedure described in Section 4.3.1 and a subset of the labeled decompositions is used to train the classifiers that judge possible decompositions, as discussed in Section 4.3.2. For the evaluations five different pattern-recognition techniques [32] have learned the judgment functions. The following classifiers have been used in the experiments:

**Linear classifier**: A linear classifier that is based on normal densities and the Bayes' rule.

**Quadratic classifier**: A quadratic classifier that is based on normal densities and the Bayes' rule.

**Parzen classifier**: A classifier that is based on Parzen densities and the Bayes' rule.

**ANN**: A classifier that is based on a feed-forward artificial neural network by back-propagation. The network is optimized w.r.t. the number of hidden units in one hidden layer.

**k-NN**: \(k\)-nearest neighbor classifier optimized w.r.t. the parameter \(k\).

Table 4.10 shows the classification error on the test set of all possible decompositions and the performance of Algorithm 3 for each classifier. The results of the linear and quadratic classifiers are somewhat surprising: the linear classifier combines the highest decomposition error (15%) with the highest correction rate (100%), whereas the quadratic classifier has the lowest decomposition error (3%) but reaches a correction rate of only 93%. This can be explained by the fact that the correction algorithm chooses the decomposition with the highest judgment value. As such, the linear classifier may make mistakes by classifying wrong decompositions as right but the
right decomposition may remain the maximum. In case of the quadratic classifier, the right decompositions may be classified wrongly, resulting in more correction errors.

4.6 Conclusions

In this chapter we have shown that there are two causes for adjoining objects in the binarization of scanned paper documents. The first cause is that objects already adjoin on the paper document. The second cause is that objects adjoin in the binarization because (i) the binarization method is erroneous or (ii) the blurring of the scanner is too severe. For both latter causes we introduced solutions.

The first solution we propose is a generally applicable syntactic method that does not require a model of the adjoined objects or their (geometrical) relationship. The method is a pre-processing method that operates on the gray-scale image data, obtained from scanning the document, and is based on an estimation of the blurring function of the lens system of the scanner. The pre-processing method is therefore applicable to different scanners, as long the PSF of the scanner is known or can be estimated. The parameters of the pre-processing method are derived from the standard deviation of the Gaussian PSF. The parameters of the pre-processing method are the radius $\rho$ and threshold $t$ of the morphological top-hat transformation. The top-hat transformation detects the missing parts of the background that caused objects to adjoin during binarization. We have shown that the results of the pre-processing method are satisfying - both for experiments on artificially generated image data as well as for experiments on scanned paper utility maps.

The solution to the second cause we propose is a semantic method that requires a model of the adjoined objects and their geometrical relationship. Because the method uses a model, it is not generally applicable. The method is a post-processing method operating on the binary image data, which is obtained by the binarization of the scanned document. The method is based on the topology of the adjoined objects that is represented with the pseudo-Euclidean distance skeleton and its branch points. The adjoined objects are corrected by breaking up the skeleton into two pieces at the position of a branch point, and by reconstructing the two objects from the resulting two pieces of skeleton. Each possible decomposition of the skeleton is judged in terms of the geometrical relationship between the two objects that are learned from examples. The decomposition with the highest judgment value is chosen as the correct decomposition of the adjoined objects into two separate objects. Experiments on scanned utility maps show a high performance for the post-processing method.
Part II

Interpretation
Chapter 5

Utility-map interpretation

This chapter presents a system for utility-map interpretation. The system is fitted into a general framework for (image) interpretation. Within the framework it is possible to represent a variety of knowledge types. The types range from specification of application-domain knowledge by specifying objects and geometrical relationships between objects, to specification of interpretation-control knowledge: interpretation order and specification of interpretation conflicts. For each object in an application domain, we can specify whether it is a simple object which has a specialized detector or whether it is a complex object which is composed of a number of objects having a specific geometrical relationship. We show that combining detectors using classifier-combining techniques to form multiple detectors reduces the number of false positive detections of simple objects. Additionally, we can represent interpretation conflicts in our framework by specifying the context in which a conflict may appear and when and how it should be resolved. We show the necessity of explicitly specified context-dependent conflict rules and introduce a conflict-resolution scheme based on learning from examples. The conflict-resolution scheme resolves conflicting interpretations by comparing their judgment values and choosing the interpretation that has the highest judgment value. Within this scheme, we introduce new judgment functions for the judgment of interpretations of the application-domain objects and geometrical relationships. These judgment functions are classifiers that are learned from examples and form an uncertainty calculus. Concerning the interpretation mechanism, we argue for the use of a priority queue as a fundamental data structure to store and retrieve hypotheses of objects. A priority queue enables a correct interpretation order in a mixed bottom-up and top-down fashion. Furthermore, it enables the specification of control knowledge on interpretation by assigning different priorities to objects. We demonstrate the applicability of our framework with experiments on utility-map interpretation.

This chapter is partly based on the following publication:
5.1 Introduction

In the last decades we have witnessed an enormous increase in the use of the computer database as a storage device for information. For all kinds of information, from simple household finances to complex road infrastructures of cities, the electronic digital information carrier is more and more commonly used than its analog paper counterpart. The advantages of electronic storage are many: it makes storage, retrieval, reproduction, exchange (for instance over the Internet), and editing of information much simpler. Although nowadays most information is stored in a digital format, some information remains only available as paper documents. If we take, for example, the public-utility organizations, we notice that most of them still have a large number of paper maps that contain information that has not yet been converted to a digital format. Den Hartog [23] estimates that for the Netherlands alone, manual conversion of the maps at the cadaster, the telecommunication services, and the public-utility organizations could take about 10.000 man years of labor. The techniques that they use to convert paper maps into digital maps differ in the amount of automation of the conversion. The conversion techniques can be classed in the following categories:

1. scanning of paper maps
2. (computer-aided) redrawing of paper maps
3. (semi-) automatic conversion of paper maps

The scanning technique is straightforward: the paper maps are digitized with a scanner and stored as a binary or gray-scale image on tape or disk. The scanning technique is not sufficient to provide digital maps that are suitable for a Geographical Information System (GIS), because it does not capture the semantic contents of the map: it only makes a digital copy. However, it can be used as an electronic backup or data-compression device, and further processing of the scanned maps can be postponed until better conversion techniques (or more funds) are available. The scanning result can also be used in another technique, i.e. computer-aided redrawing. Within this technique, the binary image of the scanning result is used as a background image on the computer display of the GIS. Using a mouse or a drawing tablet, the operator redraws the entire paper map on the computer display by positioning the corresponding GIS map objects over the binary image. A common problem with this computer-aided conversion technique is that the paper maps have (local) scale, translation, and rotation variations. This implies that objects on the paper map do not have the same size, position, and orientation as corresponding objects drawn within the GIS environment. This problem makes computer-aided redrawing not a trivial task. Furthermore, computer-aided redrawing requires a lot of human participation, which can cause errors and makes the technique expensive. For these reasons, there is currently a large interest in the third technique: automatic conversion of paper maps. This interest legitimates the scientific research on automatic conversion techniques. As fully-automated conversion is not feasible yet, we propose a semi-automatic conversion process, in which the operator corrects the results of the automatic conversion process. We stress the importance of the role of the operator in the conversion process. We provide the operator with a complete
5.1 Introduction

toolbox within the GIS environment to correct erroneous results. Furthermore, all doubtful or rejected results are kept to be presented at the end of the conversion process in a structured way. As such, the number of interaction times between the operator and the conversion process is minimal and the total interaction time is as short as possible. The proposed conversion system consists of a sequence of steps:

1. scanning of paper utility maps
2. binarization of image data
3. graphical decomposition and vectorization of binary image
4. detection of simple map objects
5. interpretation of complex map objects
6. matching interpretation results to digital topography (GBKN)
7. automatic reconstruction: geometrical alignment of interpretation results to digital topography (GBKN)

The utility maps contain spatial and semantic information of electricity networks or other modalities. They typically have scale 1 : 500 and are A0-sized. The utility maps are scanned with an ANA Tech Eagle scanner [4] at 400 dpi and saved as a gray-scale image, in which each pixel has a gray value in the range from 0 to 255. In the second step, the image is deblurred with a sharpening operator [99, 100], as introduced in Chapter 2, and binarized with a thresholding algorithm, as introduced in Chapter 3. The graphical decomposition breaks down the binary image into its graphical primitives [25]. The vectorization step consists of reducing each graphical primitive to its skeleton, composed of a number of vectors. The gray-scale image, the binary image, the graphical decomposition, and the vectorization are the input of detectors that can recognize simple map objects [24, 50]. The interpretation process interfaces with the detectors. It takes as input some initially detected simple objects and groups them into complex objects by using detectors to discover geometrically related objects [97, 98]. The results of the interpretation step are used in the matching procedure, together with a so-called large-scale base map (GBKN). The base map is a GIS database of the Dutch cadaster that stores geographical information on roads, houses, and other topographical objects. In the matching procedure, the houses found in the interpretation are matched to the houses of the base map [21]. The matching information is used to reconstruct the utility map such that its topographical position matches the actual position of the map's contents [20]. For the system layout of the complete conversion process, we refer to [26, 105]. A simplified overview of the proposed system is shown in Figure 5.1. This chapter focuses on the first interpretation part that works by way of detectors on the gray-scale image of a scanned utility map. The second interpretation part, which works on labeled vector data, is used in the automatic reconstruction of utility maps, and is described in more detail in Chapter 6 and [20].

The outline of this chapter is as follows: an overview of related work on document interpretation is given in Section 5.2. Section 5.3 introduces our interpretation system and focuses on its knowledge representation, the mechanism of interpretation, and hashing on geometrical attributes. Section 5.4 describes our concept of
single and multiple detectors for the detection of simple map objects. For both types of detectors, results of experiments are given. In Section 5.5, we describe our uncertainty calculus based on learning from examples for the judgment of interpreted map objects. The judgment procedure is illustrated with experimental results. In Section 5.6 this judgment procedure is used for the resolution of interpretation conflicts. A new conflict-resolution strategy is introduced and the necessity of explicitly specified conflicts is shown. Section 5.6.3 gives the results of experiments with the interpretation system on a set of utility maps.

5.2 Related work

Related work on utility-map interpretation is reported by Joseph et al. [51], Myers et al. [76], Den Hartog [23], Ogier et al. [79], Deseilligny et al. [27], and Frischknecht et al. [35]. Related work on (engineering-) drawing interpretation is reported by Kasturi et al. [55], Dori [29], Thomas et al. [110], Ablamayko et al. [1], Hutton et al. [47], and Joseph et al. [52]. What makes these interpretation systems hard to compare is that most of the them are tuned to their specific (commercial) applications. Although some authors claim that their systems are generally applicable [23], most of the authors make no such claims or do not validate them with experimental results [76]. The general applicability of an interpretation system ensures that the system has an open, modular architecture and that application-domain knowledge can easily be adjusted. A dedicated interpretation system, however, may have a rapidly-developed prototype, but it suffers from its unstructured design at the end of its development. Additionally, a dedicated system can have a large amount of its application-domain knowledge hidden inside the source code of the system. As a result, the adjustment of application-domain knowledge is cumbersome and slow to test because it implies a recompilation of the system’s source code.

When we compare interpretation systems, another problem is that the level of interpretation differs with the application. The level of interpretation ranges from the detection of graphical objects [1] to an almost complete recognition of all map
<table>
<thead>
<tr>
<th>System</th>
<th>Knowledge-based</th>
<th>Knowledge representation</th>
<th>Control knowledge</th>
<th>Information source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kasturi '90 [55]</td>
<td>no</td>
<td>source code</td>
<td>none</td>
<td>vectors</td>
</tr>
<tr>
<td>Joseph '92 [52]</td>
<td>yes</td>
<td>frames</td>
<td>control rules</td>
<td>gray-scale image</td>
</tr>
<tr>
<td>Dori '95 [29]</td>
<td>yes</td>
<td>object-process diagram (OPD)</td>
<td>OPD</td>
<td>graphical primitives</td>
</tr>
<tr>
<td>Myers '95 [76]</td>
<td>yes</td>
<td>unspecified</td>
<td>none</td>
<td>gray-scale image</td>
</tr>
<tr>
<td>Thomas '95 [110]</td>
<td>yes</td>
<td>semantic network</td>
<td>none</td>
<td>vectors and text</td>
</tr>
<tr>
<td>Den Hartog '95 [23]</td>
<td>yes</td>
<td>semantic network</td>
<td>spatial relationships</td>
<td>graphical primitives</td>
</tr>
<tr>
<td>Ablameyko '97 [1]</td>
<td>yes</td>
<td>grammar</td>
<td>unspecified</td>
<td>vectors</td>
</tr>
<tr>
<td>Hutton '97 [47]</td>
<td>no</td>
<td>none</td>
<td>none</td>
<td>vectors</td>
</tr>
<tr>
<td>Ogier '97 [79]</td>
<td>yes</td>
<td>unspecified</td>
<td>none</td>
<td>gray-scale image</td>
</tr>
<tr>
<td>Deseilligny '97 [27]</td>
<td>yes</td>
<td>graph</td>
<td>process order</td>
<td>binary</td>
</tr>
<tr>
<td>Frischknecht '97 [35]</td>
<td>yes</td>
<td>templates</td>
<td>none</td>
<td>image and vectors binary image</td>
</tr>
</tbody>
</table>

Table 5.1: Comparison between different document-interpretation systems (continued in Table 5.2).

objects and the map structure [23]. This problem is largely due to the lack of a (formal) definition of image interpretation. We regard the detection of simple graphical objects in maps or drawings as a (pattern-) recognition process. Interpretation of a map or a drawing is regarded as the process that finds an interpretation of its application-domain model in the input data. The application-domain model contains knowledge on simple objects and structure. Structure is viewed as the geometrical relationships between objects and the aggregation of simple objects into complex objects. In our view on interpretation, an important role is played by representation of application-domain knowledge, the interpretation mechanism, the conflict resolution strategy, and the applied uncertainty calculus and these make the difference between a recognition and an interpretation process. With the above-mentioned problems of comparison in mind, we will compare the interpretation systems. Tables 5.1 and 5.2 give a comparison between the different interpretation systems. The comparison is made based on the following questions:

- Is the interpretation system a knowledge-based approach?
- What kind of knowledge representation is used?
- Can knowledge on interpretation control be specified?
- What is the information source?
- How does the interpretation mechanism function?
- Does the interpretation system have conflict resolution or inconsistency handling?
- What is the applied uncertainty calculus, if any?
<table>
<thead>
<tr>
<th>System</th>
<th>Interpretation mechanism</th>
<th>Conflict resolution</th>
<th>Uncertainty calculus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kasturi '90 [55]</td>
<td>bottom-up</td>
<td>no</td>
<td>none</td>
</tr>
<tr>
<td>Joseph '92 [52]</td>
<td>hybrid</td>
<td>no</td>
<td>none</td>
</tr>
<tr>
<td>Dori '95 [29]</td>
<td>unspecified</td>
<td>no</td>
<td>none</td>
</tr>
<tr>
<td>Myers '95 [76]</td>
<td>hybrid</td>
<td>yes</td>
<td>unspecified</td>
</tr>
<tr>
<td>Thomas '95 [110]</td>
<td>bottom-up</td>
<td>no</td>
<td>none</td>
</tr>
<tr>
<td>Den Hartog '95 [23]</td>
<td>hybrid</td>
<td>yes</td>
<td>none</td>
</tr>
<tr>
<td>Ablameyko '97 [1]</td>
<td>bottom-up</td>
<td>no</td>
<td>none</td>
</tr>
<tr>
<td>Hutton '97 [47]</td>
<td>bottom-up</td>
<td>no</td>
<td>none</td>
</tr>
<tr>
<td>Ogier '97 [79]</td>
<td>hybrid</td>
<td>yes</td>
<td>none</td>
</tr>
<tr>
<td>Deseilligny '97 [27]</td>
<td>top-down</td>
<td>no</td>
<td>none</td>
</tr>
<tr>
<td>Frischknecht '97 [35]</td>
<td>bottom-up</td>
<td>no</td>
<td>none</td>
</tr>
</tbody>
</table>

Table 5.2: Comparison between different document-interpretation systems (continued from Table 5.1).

We call an interpretation system knowledge-based if the knowledge is made explicit in a specification that is specially designed to represent knowledge. For a knowledge-based system, we list the categorization of the knowledge representation into (production) rules, (string) grammars, (syntactic) trees, graphs, frames, or semantic networks [7, 41]. If the knowledge representation cannot be categorized, it is listed with its own name. Control knowledge defines knowledge on interpretation order and interpretation control. It is sometimes beneficial in terms of interpretation efficiency and interpretation accuracy to be able to specify which application-domain objects must be detected before other (geometrically related) objects. Furthermore, control knowledge can define heuristics (or meta-rules) on actions to be taken by the interpretation process in some process states: it defines control on interpretation.

The information source describes which type of information is the primary source of input to the interpretation system. A large number of interpretation systems operate solely on vectorization data, after the graphics is separated from the text, which can be recognized by OCR [1, 47, 55, 110]. Other information sources include: grayscale image data [52, 76, 79], binary-image data, and graphical primitives obtained from the binary-image data [23].

Interpretation mechanisms can be bottom-up (data-driven), top-down (goal-driven), or a hybrid combination of both strategies. Rule-based interpretation systems are by definition strictly bottom-up or strictly top-down [7]. It has been widely accepted that an interpretation mechanism that has both data-driven and goal-driven capabilities is the best choice for the majority of interpretation applications [7, 23, 48]. A combined bottom-up and top-down approach to image interpretation is favored because neither bottom-up nor top-down processing yields a solution to image interpretation: a top-down approach is limited by the restraints of the application-domain model and a bottom-up approach may result in a combinatorial explosion of possible interpretations. A combined strategy has the advantages of both approaches: bottom-up processing to bootstrap interpretation, and top-down processing to control interpretation and direct bottom-up processing to parts of the
image where application-domain objects are hypothesized.

Finally, we analyze how the interpretation systems deal with interpretation conflicts or interpretation inconsistencies, and whether they use an uncertainty calculus in the resolution of interpretation conflicts. An uncertainty calculus defines the functions for the (bottom-up) propagation of uncertainty from the early image processing to the highest interpretation level over the aggregation of objects and geometrical relationships. An uncertainty calculus can, for example, be based on probabilistic approaches (Certainty Factor model [104], Bayesian belief networks [82], Dempster-Shafer theory [22]), possibilistic approaches (fuzzy logic [120], fuzzy set theory [121], possibility theory [122]), or be learned from examples.

From Tables 5.1 and 5.2 we conclude that the most interesting approaches are by Den Hartog [23], Myers et al. [76], Joseph et al. [52], and Ogier et al. [79]. These approaches are knowledge based, operate on gray-scale images or graphical primitives, have a hybrid interpretation mechanism, and apply conflict resolution. We discuss them in the following paragraphs.

**Den Hartog's system** The approach by Den Hartog is a knowledge-based system for the interpretation of Dutch public-utility maps. The system has been extensively tested and its performance is well evaluated. The drawbacks of the system are that control knowledge is specified as spatial relationships, hierarchical knowledge is not supported, and the conflict-resolution scheme is limited. The knowledge on the utility-map domain is represented within a semantic network in a procedural way: a chain of map objects and geometrical relationships between map objects is used to depict the order of interpretation of the objects. An adjustment of the interpretation order implies a redesign of the utility-map knowledge. Furthermore, the system processes the map objects in the chain one by one, the geometrical relationship between two consecutive map objects sets the search area for the next map object on the basis of the previously detected object. A problem occurs if a map object cannot be found in the image data: the remaining map objects in the chain remain unprocessed. Because application-domain knowledge is represented in a procedural way, hierarchical knowledge, i.e. how simple objects aggregate into more complex or more abstract objects, cannot be specified. The conflict-resolution scheme of the system is limited: it can only detect and delete (contextual) inconsistencies.

**Myers's system** The system by Myers et al. converts maps from scanned images to an object-oriented representation. The system applies conflict resolution to resolve ambiguities of each processing stage by driving additional search actions for related features or associated information. A drawback of the system is that interpretation conflicts are resolved at the interpretation level of all data. This could result in an enormous amount of possible interpretations if the input data is dense or noisy. Furthermore, although confidence measures are applied, the exact nature of these confidence measures is not revealed.

**Joseph's system** Joseph et al. present a methodology for the interpretation of images of engineering drawings. Their approach is based on the combination of frames (denoted as schemata) describing prototypical drawing constructs with a library of
low-level image-analysis routines and a set of explicit control rules. The control rules are applied by a parser. The system integrates bottom-up and top-down processing strategies within a single, flexible framework modeled on the human-perception cycle. The drawbacks of the system are that the system has deficiencies in the handling of multiple interpretations of entities and conflict resolution is impossible for some situations because of the system’s architecture.

Ogier’s system The interpretation system of Ogier et al. is developed for the interpretation of French cadastral maps. The system is based on visual-perception studies and a hierarchical description of a model map. Its conflict-resolution strategy tries to achieve semantic internal and external coherency of the interpretation of the map. Internal coherency is achieved if a interpreted map object is complete (in the aggregation sense), whereas external coherency is achieved if the map object fits into its neighborhood, as defined by the model map. Conflicts arise if a map object is not externally coherent. The conflict-resolution scheme of the system calls low-level image operators to refine the detection of the incoherent map object. This scheme only works if low-level image operators can provide a solution to the conflicts; for more complex map objects this may not be the case.

Concluding, we may state that there are a number of knowledge-based interpretation systems that could meet (partly) our objectives but lack, in our opinion, some (or a combination of) important features. These features include the following:

- hierarchical-knowledge specification
- control-knowledge specification
- multiple information sources
- hybrid interpretation mechanism
- a proper handling of interpretation conflicts
- application of an uncertainty calculus

We regard these features as important but not essential to the design of an interpretation system for utility maps. In other words, one can design an interpretation system for utility maps that has none of the listed features but that is able to find an interpretation of a map. We view the above-mentioned features as extensions to existing approaches to image interpretation that contribute to the flexibility, reliability, and performance of an interpretation system.

5.3 Interpretation

This section introduces our framework for an interpretation system. We start with a description of its knowledge representation in Section 5.3.1. An example of the use of the knowledge representation is given in Section 5.3.2. Section 5.3.3 describes the mechanism of interpretation and the application of a special data structure for it, an example of an interpretation is given. Section 5.3.4 introduces hashing on geometrical attributes, a technique which can speed up interpretation by application of special data structures.
5.3.1 Knowledge representation

An image-interpretation process finds an interpretation of its application-domain model in the image. A representation for an application-domain model and application-domain knowledge can be a (syntactic) tree, a graph, a (string) grammar, (production) rules, frames, or a semantic network [7, 41]. The choice of representation depends on the suitability of the representation for the particular application. The common factors among these representations are that they have simple (atomic) objects and structure as well as an additional possibility to specify control knowledge. Simple objects are, for instance, tree nodes, graph nodes, grammar terminals, frames, semantic-network nodes, etc. A structure is built with tree edges, graph edges, grammar sentences, links, and relationships. We view interpretation as the process that forms instantiations of complex application-domain objects from simple (and recursively complex) objects that have geometrical relationships. Interpretation bears great similarity to logical inference [41]. Indeed, most of the representations can be converted to First-Order Predicate Logic (FOPL), in which atomic objects become predicates and structure becomes rules. The reason that logical resolution is not often used for image-interpretation systems lies within the lack of possibilities to control logical resolution with knowledge, which is available for most applications. Current implementations of FOPL, which are based on logical resolution, are only controllable by means of the choice of type of resolution or lexical ordering of the rules (PROLOG [62]). Furthermore, they usually lack an interface to the C programming language [58], which is necessary to connect to the low-level image-processing routines. As a result, most image-interpretation systems have some kind of tree, graph, grammar, or semantic network as a model or knowledge representation. Because these knowledge representations are not clearly defined, one has the opportunity to modify the representation to one's needs. Furthermore, as mentioned above, the knowledge representations have some common factors and that make the distinction between the different knowledge representations vague and a choice between different representations possibly arbitrary. Additionally, representations exist that combine characteristics of different representations. A semantic network, for example, may represent knowledge on an application domain as frames in a tree or graph-like structure. Because of its combined character we have chosen a semantic network as the knowledge representation for our interpretation system.

The knowledge representation used in our interpretation system adopts a semantic-network implementation as described by Niemann et al. in [78]. Many practical applications of this implementation have been reported in the literature [43, 44, 67, 96, 97]. The choice of a semantic network as a knowledge representation for our utility-map domain is justified by the fact that map objects and (geometrical) relationships between map objects can be adequately represented with concepts and relationships within the semantic network. Furthermore, the part and the specialization links of the semantic network provide a facility to represent hierarchical utility-map domain knowledge. Part links describe how complex objects are formed out of a number of other objects, e.g. a dimension consists of an arrow, house side, and a conduit. Specialization links describe that two or more objects are in fact the same object: a single- and a double-headed arrow are both arrows, see Figure 5.2. A semantic network has a so-called long-term memory (LTM) and a short-term memory (STM). Items stored in LTM are known beforehand and
Figure 5.2: Semantic network as a knowledge representation for a dimension that measures the distance between a house side and the conduit. (a) Semantic network depicted visually with graphical figures for concepts, relationships, and links. (b) Part of a utility map which shows an example of a dimension that measures the distance between a house side and the conduit with a double-headed arrow. (c) Two examples of a dimension with a single-headed arrow.
define the application-domain knowledge; usually they are explicitly specified in a knowledge file (see Figure 5.3). Items stored in STM are built up during interpreta-

/* CONCEPT DESCRIPTIONS */
DEFINE CONCEPT
  name "dimension";
  priority 9;
  attribute:vector (0, 0, 99999, 99999);
  attribute_function:vector<dimension>;  
  judgment_function:judge_dimension;
ENDDEF;

DEFINE CONCEPT
  name "house side";
  priority 10;
  attribute:vector (0, 0, 99999, 99999);
  attribute_function:vector<house_side_width>;
  detector:house_side_detector;
ENDDEF;

DEFINE CONCEPT
  name "arrow";
  priority 12;
  attribute:vector (0, 0, 99999, 99999);
  attribute_function:vector<arrow>;
  judgment_function:judge_arrow;
ENDDEF;

DEFINE CONCEPT
  name "conduit";
  priority 11;
  attribute:vector (0, 0, 99999, 99999);
  attribute_length:10, 256.00;
  detector:conduit_detector;
ENDDEF;

DEFINE CONCEPT
  name "double-headed arrow";
  priority 14;
  attribute:vector (0, 0, 99999, 99999);
  detector:double-headed_arrow_detector;
ENDDEF;

DEFINE CONCEPT
  name "single-headed arrow";
  priority 13;
  attribute:vector (0, 0, 99999, 99999);
  detector:single-headed_arrow_detector;
ENDDEF;

/* PART LINK DESCRIPTIONS */
DEFINE PART_LINK
  parent "dimension";
  child "house side";
ENDDEF;

DEFINE PART_LINK
  parent "dimension";
  child "arrow";
ENDDEF;

DEFINE PART_LINK
  parent "dimension";
  child "conduit";
ENDDEF;

/* SPECIALIZATION LINK DESCRIPTIONS */
DEFINE SPECIALIZATION_LINK
  parent "arrow";
  child "double-headed arrow";
ENDDEF;

DEFINE SPECIALIZATION_LINK
  parent "arrow";
  child "single-headed arrow";
ENDDEF;

/* RELATIONSHIP DESCRIPTIONS */
DEFINE RELATIONSHIP
  role "in line";
  parent "dimension";
  argument0 "conduit";
  argument1 "arrow";
  argument2 "house side";
  judgment_function:judge_in_line(10, 256.00);
  pruning_function:vector_endpoint(100, 100, 100);
ENDDEF;

DEFINE RELATIONSHIP
  role "perpendicular";
  parent "dimension";
  argument0 "arrow";
  argument1 "conduit";
  judgment_function:judge_perpendicular(10, 100, 100, 200);
  pruning_function:vector_endpoint(100, 100, 100);
ENDDEF;

Figure 5.3: Knowledge file of knowledge represented in the semantic network of Figure 5.2.

The LTM items are concepts, whereas the STM items are modified concepts and instantiations. The difference between concepts and modified concepts is that modified concepts have stricter restrictions on attributes, set upon by (geometric) relationships with instantiations during interpretation, whereas instantiations actually have attribute values.

Concepts in LTM are specified by a declarative part and a procedural part. The declarative part describes attributes of concepts (e.g. length, width, area), links to other concepts (e.g. part link, specialization link), and geometrical relationships between concepts (e.g. perpendicular, parallel, in-line, T-junction). Concepts, links, and relationships are described in the following sections. The declarative knowledge part of a semantic network facilitates the composition of a map-object hierarchy by means of links. The part link descriptions describe the aggregation of objects into more complex objects. The specialization link descriptions provide the mechanism of abstraction (like between class structures in object-oriented programming languages). The map objects at the leaves of a map-object hierarchy have a specialized detector to generate instantiations. The map objects that have parts or specializations will be denoted as complex map objects and have no specialized detector, but are generated by combining lower-level map-object instantiations. The procedural part describes which functions (e.g. written in the C(++) programming language) are to be called by the interpretation process to calculate attributes, judge
instantiations and relationships between instantiations, and to detect simple concepts. The procedural part defines an interface to the libraries of attribute and judgment functions as well as the library containing the detectors, see Figure 5.4.

Concepts

This section gives descriptions of application-domain concepts within a semantic network and its knowledge file. Concepts are specified in a knowledge file, as depicted in Figure 5.3, by a name string, a priority, attributes, attribute-calculation functions as well as a detector or judgment function. Simple concepts have a detector, whereas complex concepts have a judgment function and attribute-calculation functions. The name string of a concept is the unique key attribute of a concept and it is used to retrieve concepts from the LTM. The priority attribute is for interpretation-control purposes only, and its use is discussed in Section 5.3.3. Each simple and complex concept can have several attributes, for which an initial range can be specified. The attribute values of a simple concept are calculated from its corresponding graphical primitive or are given by the detector. The attribute values of a complex concept are calculated from the attribute values of its parts or specializations. The way attributes of a complex concept are calculated is specified with an attribute-calculation function. Attributes can be selected from the following list of standard attributes:

- length of the object
- minimum, maximum, and average width of the object
- area of the object
- point attribute
5.3 Interpretation

- vector attribute
- vector-list attribute
- Minimum-Area Encasing Rectangle (MAER) attributes:
  - MAER attribute
  - height of MAER
  - width of MAER
  - density of MAER

Notice that some attributes are only meaningful for simple objects. The length attribute of a (simple) object, calculated from its graphical primitive, equals the number of pixels of the skeleton of the graphical primitive. An example of a skeleton of a graphical primitive is given in Figure 5.5(a). The width attributes are calculated from the width of the graphical primitive at the pixel positions of the skeleton, see Figure 5.5(b). The area attribute equals the number of pixels of the graphical primitive. The point attribute of a (simple) object equals the center of gravity of its graphical primitive. The vector-list attribute takes its value from the vectorization of the graphical primitive. The value of the vector attribute is taken as the vector from the starting point to the end point of the vector list. The MAER attributes are calculated from the minimum-area encasing rectangle [34] of the graphical primitive. The MAER an object is the smallest rectangle that can be fitted around the object. An example of a MAER of a graphical primitive is given in Figure 5.5. The density of a MAER equals the ratio between the number of pixels of the graphical primitive and the area of the MAER. The point, vector(-list), and MAER attributes of an object represent the geometrical position of the object. Each simple and complex object must have at least one of those attributes to position it. Because the list of attributes is limited, future applications of our framework for interpretation could require

\[\text{Figure 5.5: Attributes are calculated using the skeleton and Minimum-Area Encasing Rectangle (MAER) of a graphical primitive.}\]
the writing of additional functions to calculate other attributes from the graphical primitives. A simple object can have a detector and attributes with restrictions. For such simple objects, the detection results of the detector are pruned by the given attribute restrictions: if a detected simple object has attribute values that lie outside the attribute restrictions then it is removed from the detection results.

For each complex object, a judgment function can be specified or a default judgment function can be selected. A judgment function of a complex object is a combination function which takes the judgment of its parts or specializations and the judgment of its geometrical relationships as function arguments. The judgment function takes its arguments as real numbers in the range [0, 1] and maps its function (or judgment) value to a real number in the range [0, 1]. A judgment value of zero represents the worst judgment or no confidence, whereas the value one represents the best judgment or total confidence. The default judgment function is a combination function that calculates the average judgment value of its parts, specializations, and geometrical relationships. Otherwise, if the default judgment function is not chosen, a special judgment function can be implemented and put into the judgment library, see Figure 5.4. Consequently, the complex concept must be augmented with a specification of the judgment function, see Figure 5.3. The design of special judgment functions is discussed in Section 5.5. A complex object can have a default or special judgment function and attributes with restrictions. For such complex objects, the interpretation results are pruned by the given attribute restrictions in the same way as with simple objects.

Part and specialization links

In our semantic network approach, a distinction is made between two kinds of links: \textit{part} and \textit{specialization} links. A part or specialization link is defined in the knowledge file by specifying the parent and child concept of the link. The difference between a part and a specialization link is that a complex object which has parts can only be instantiated if all parts are instantiated, whereas a complex object which has specializations can be instantiated if only one of its specializations is instantiated. The part and specialization links are the semantic-network counterparts of the logical conjunction (AND) and disjunction (OR). For instance, the complex \textit{dimension} object, shown in Figure 5.2(a), can only be instantiated if its parts \textit{house side}, \textit{arrow}, and \textit{conduit} are instantiated. The complex \textit{arrow} object, also shown in Figure 5.2, can be instantiated if one of its specializations (\textit{single-} or \textit{double-headed arrow}) is instantiated. The specialization links make an abstraction from \textit{single-} and \textit{double-headed} possible. They give way to a more compact specification of a \textit{dimension} because it avoids replication of specification. Another example is given in Figure 5.6. The complex \textit{dimension-leader} object, given in Figure 5.6(a), can be instantiated if its \textit{dimension-leader house side}, \textit{dimension-leader arrow} or \textit{dimension-leader conduit} is instantiated. The specialization of a \textit{dimension leader} makes the specification of \textit{dimensions} using \textit{dimension leaders} simpler. Figures 5.6(b), 5.6(b), and 5.6(b) give some examples of a \textit{dimension} that measures the distance from a \textit{dimension leader} to a \textit{conduit} with different specializations of the \textit{dimension leader}. 

Figure 5.6: Difference between a part and a specialization link. Specialization link: a dimension leader equals a dimension leader from a house side OR from an arrow OR from a conduit. Part link: a dimension leader from a house side is composed of a house side AND a dashed line. (a) Semantic network representation of a dimension leader. Parts of a map that shows a dimension leader originating from (b) a house side, (c) an arrow, and (d) a conduit.
During interpretation, the part and specialization links are used to copy restrictions on attributes from parent to child concepts (or vice versa for bottom-up processing). If the parent and child concept do not have the same attributes then the attribute restrictions are not copied except for attributes that represent the position of a concept: vector, MAER, and point attributes. For example, if a parent concept has a vector attribute and a part link to a child concept that has a point attribute then the restrictions on the vector attribute of the parent are copied to the point attribute of the child.

Relationships

In our semantic network approach, it is possible to specify binary and tertiary (geometrical) relationships between objects. Relationships are given within the context of a complex object; a relationship is in fact an attribute of a complex object. This stems from the fact that objects may be part of more than one complex object. As a result, the relationships between two or three objects may vary over the different complex objects the objects are part of. In order to differentiate between those different relationships, a relationship is defined within the context of a complex object. For each relationship, a judgment and pruning function has to be specified. A judgment function judges an instantiation of a geometrical relationship and gives a real number in the range [0, 1] that expresses an amount of confidence or belief in the existence of the geometrical relationship. A pruning function of a geometrical relationship restricts the attribute values of an argument of the relationship, i.e. an object, by means of the values or restrictions of the attributes of the other argument(s) of the relationship. Pruning functions are used to specify search areas for related objects during interpretation.

Judgment functions A judgment function can either be chosen from a set of standard judgment functions, or a special judgment function can be implemented and put into the judgment library. The set of standard judgment functions include judgment functions for the following relationships:

in line : The two arguments of the relationship are in line. The two parameters of the relationship are the angle, denoted as $\alpha$, between the arguments and the distance, denoted as $d$, between the arguments.

perpendicular : The two arguments of the relationship are perpendicular. The two parameters of the relationship are the angle, denoted as $\alpha$, between the arguments and the distance, denoted as $d$, between the arguments.

parallel : The two arguments of the relationship are parallel. The two parameters of the relationship are the difference in orientation angle, denoted as $\alpha$, and the distance, denoted as $d$, between the arguments.

T-junction : The two arguments of the relationship form a T-junction. The three parameters of the relationship are the angle $\alpha$ between the arguments, the distance $d$ between the arguments and the minimum distance, denoted as $d_{\text{min}}$, between the point of intersection of the arguments and the end point of one of the arguments.
distance: The two arguments of the relationship are within the specified distance of each other. The parameter of the relationship is the distance $d$.

All of the standard geometrical relationships, except the distance relationship, require that the objects that have such a relationship have a vector or MAER attribute.

![Diagram of standard geometrical relationships](image)

Figure 5.7: (a) Examples of standard geometrical relationships. (b) Judgment functions of geometrical relationships calculate a judgment value from deviations of the relationship's parameters from the expected parameter values according to a $L1$ norm that approximates $P(\text{relationship} \mid \Delta\text{parameter})$. The depicted judgment function calculates a judgment value from the deviation $\Delta\alpha$ from the expected value $\mu_\alpha$. If the deviation is within $\sigma_\alpha$ distance of $\mu_\alpha$, the judgment is greater than zero. (c) Parameters of standard geometrical relationships.

The standard geometrical relationships are visualized in Figure 5.7(a). The judgment value of a geometrical relationship is computed from the deviations of the actual parameter values of a relationship from the expected parameter values:

\begin{align}
\Delta\alpha &= \alpha - \mu_\alpha \\
\Delta d &= d - \mu_d \\
\Delta d_{\text{min}} &= d_{\text{min}} - \mu_{d_{\text{min}}}
\end{align}

(5.1)  (5.2)  (5.3)
We have for all standard geometrical relationships: $\mu_d = 0$, and $\mu_{d_{\text{min}}} = 0$ for the relationship T-junction. Consequently, we have: $\Delta d = d$ and $\Delta d_{\text{min}} = d_{\text{min}}$. The value $\mu_\alpha$ varies over the different relationships. The parameters $\Delta \alpha$, $d$ and $d_{\text{min}}$ are shown in Figure 5.7(c). For example, the judgment value of an in-line relationship is calculated from approximations of the conditional probability density functions of the in-line relationship given the deviations of the parameters of the relationship:

$$P(\text{in line} \mid \Delta \alpha, \Delta d) = P(\text{in line} \mid \Delta \alpha)P(\text{in line} \mid \Delta d) = G(\Delta \alpha; \mu_\alpha, \sigma_\alpha)G(\Delta d; \mu_d, \sigma_d)$$

We assume that the parameters $\Delta \alpha$ and $d$ are conditionally independent under the hypothesis of the in-line relationship. Function $G(z; \mu, \sigma)$ can be a Gaussian function or L1 norm parametrized by $\mu$ and $\sigma$. Figure 5.7(b) shows a L1 norm as a judgment function, given parameter $\Delta \alpha$.

**Special judgment functions** If a relationship from an application domain cannot be represented with one of the standard relationships, or if the standard judgment function is not adequate, then a special judgment function can be implemented and put into the judgment library. The implementation of a special, i.e. non-standard, judgment function for geometrical relationships is discussed in Section 5.5.1.

**Pruning functions** A pruning function for a geometrical relationship can be selected from a set of standard pruning functions. The set of standard pruning functions is divided in a set of pruning functions for binary relations and a set for tertiary relationships. The pruning functions for binary relationships restrict the attribute values of the second argument of the relationship by means of the values or restrictions of the attributes of the first argument of the relationship. The pruning functions for tertiary relationships restrict the attribute values of the third argument of the relationship by means of the values or restrictions of the attributes of the first and second argument of the relationship. The standard pruning functions for binary relationships are the following:

- **point**: a restriction on the MAER, point, and vector attributes of the second argument are set by the attribute value of the point attribute of the first argument of the relationship. This pruning function has one parameter, denoted as $r$, that is the radius of the circle which forms the restriction, see Figure 5.8(b).

- **vector end points**: two restrictions on the MAER, point, and vector attributes of the second argument are set by the attribute value of the vector attribute of the first argument of the relationship. This pruning function has four parameters, denoted as $r$, $b$, $l$, and $f$, that compose two rectangles at the end points of the vector, see Figure 5.8(a).

- **MAER ends**: two restrictions on the MAER, point, and vector attributes of the second argument are set by the attribute value of the MAER attribute of the first argument of the relationship. This pruning function has four parameters, denoted as $r$, $b$, $l$, and $f$, that compose two rectangles at the ends of the MAER in the same way as for the vector end points pruning function.
Figure 5.8: Standard pruning functions for binary and tertiary relationships (search-area specification). (a) Standard pruning function vector end points \((r, b, l, f)\) for binary relationships. (b) Standard pruning function point \((r)\) for binary relationships. (c) Standard pruning function vector end points \((r, b, l, f)\) for tertiary relationships. (d) Application of pruning functions vector end points and vector end point for the binary and tertiary relationships, as defined in the knowledge file of Figure 5.3.
The standard pruning functions for tertiary relationships are the following:

**vector end point**: a restriction on the MAER, point, and vector attributes of the third argument are set by the attribute values of the vector attribute of the second argument and the MAER, point, or vector attribute of the first argument of the relationship. This pruning function has four parameters, denoted as \( r, b, l, \) and \( f \), that compose a rectangle at the end point of the vector that has the greatest distance to the first argument, see Figure 5.8(c).

**MAER end**: a restriction on the MAER, point, and vector attributes of the third argument are set by the attribute values of the MAER attribute of the second argument and the MAER, point, or vector attribute of the first argument of the relationship. This pruning function has four parameters, denoted as \( r, b, l, \) and \( f \), that compose a rectangle at the end of the MAER that has the greatest distance to the first argument.

Two examples of the applications of standard pruning functions for binary and tertiary relationships are given in Figure 5.8(d). The geometrical relationships are taken from the example given in Figures 5.2 and 5.3. The pruning function of the binary relationship between an arrow and a conduit sets the restrictions (search areas) for the conduit object by means of the vector of the arrow, and the pruning function of the tertiary relationship between a conduit, arrow, and house side sets a restriction for the house side object by means of the vectors of the conduit and arrow.

### 5.3.2 A knowledge-representation example

Figures 5.2 and 5.3 give an example of a semantic network and its corresponding knowledge file. The knowledge file defines a (complex) concept named dimension with priority zero, a vector attribute, an attribute-calculation function vector dimension and a judgment function judge dimension. The priority is set to zero because the concept dimension is the root node of the object hierarchy. The vector attribute has an initial restriction of the attribute to the whole image represented with the box \([0, 0, 99999, 99999]\), and is calculated using the specified attribute function. The specified judgment function is a special judgment function for concepts. After the definition of the concept dimension, the concepts house side, arrow, and conduit are defined in the knowledge file. The arrow concept is a complex object specified in a similar way as the concept dimension. The house side and conduit are simple objects and have a detector and no part or specialization links. Both the house side and conduit have an additional attribute, which are calculated from the graphical primitive and have a restriction to the intervals \([18.3, 20.0]\) and \([10.0, 256.0]\) respectively. The following two simple concepts, i.e. double-headed arrow and single-headed arrow, only have a detector. Then, the three part links house side, arrow, and conduit of the dimension concept are defined, followed by the two specializations of the concept arrow. Finally, the in-line geometrical relationships between the house side and arrow concepts, and the perpendicular geometrical relationship between the conduit and arrow concepts are defined. The in-line relationship is a standard tertiary relationship. The standard judgment function judge
in line(20,0.3) specifies that the judgment is positive if the distance between the arrow and house side is less than 20 pixels and the orientation difference is less than 0.3 radian. The pruning function of this in-line relationship is specified by means of the standard pruning function vector end points(10,10,10,20) and sets the search areas for the house side given the vectors of the instantiations of the conduit and arrow concepts. The perpendicular relationship between an arrow and conduit is defined in a similar fashion, but has a special judgment function.

5.3.3 Interpretation mechanism

This section describes the mechanism of interpretation, and the use of the priority queue [107] as a data structure to store hypotheses and to control interpretation. The mechanism uses the well-known hypothesis-generation, hypothesis-selection, and hypothesis-verification cycle [48] as shown in Figure 5.9. The interpretation cycle processes hypotheses on the existence of an LTM concept somewhere in the image data. The data structure of a hypothesis records a pointer to the LTM concept, a priority, a status field, a list of alternatives (modified concepts) as well as a list of instantiations, see Figure 5.10. The list of alternatives contains one or more (partial) copies of the LTM concept that are not or only partly instantiated. The copies represent alternative interpretations of the LTM concept and differ from the
original LTM concept in the sense that they have stricter restrictions on attributes imposed by pruning functions of relationships or links with instantiated concepts. As such, they are denoted as modified concepts. The list of instantiations contains a number of instantiations of the LTM concept.

The interpretation starts with finding all (possible) instantiations of one or more simple objects in LTM. The instantiations are found by calling the corresponding detectors. This initial bootstrap results in a set of instantiations of simple objects, which are stored in STM. The choice which simple objects are to be detected first is important because the bootstrap results must cover most of the map contents to ensure a complete map interpretation by the interpretation cycle. This demand can be met if we choose a simple object that has a detector with a high detection rate or if we choose more than one simple object has to be detected first. All resulting instantiations are also stored as (verified and instantiated) hypotheses having an empty alternative list and one instantiation. From that point on, interpretation continues with a cycle of hypothesis selection, verification and generation. In the selection step, the hypothesis with the highest priority is chosen out of a set of hypotheses. The selected hypothesis is denoted as the working hypothesis. Next, in the verification step, the working hypothesis is verified and if verification is successful, instantiations of the concept of the working hypothesis are made and stored in the list of instantiations of the hypothesis. Verification is not successful when a hypothesis on a complex object cannot be verified because one or more parts, or one specialization of the alternatives of the complex object are not instantiated yet. The generation step depends on the results of the verification. If verification is successful, then new hypotheses on the existence of higher-level objects are generated on the basis of the instantiations made in the verification step. The generation of hypotheses follows the part of or specialization of links of the LTM concept of the working hypothesis. If verification is not successful, new hypotheses on the existence of lower-level objects are generated; the generation follows the part and specialization links of the LTM concept of the working hypothesis.
5.3 Interpretation

Hypothesis selection  Hypothesis selection is performed by means of a priority queue. The use of a priority queue offers the possibility to control interpretation by assigning priorities to the different concepts in LTM. The priority of each queue item, a hypothesis, corresponds to the priority of the hypothesized concept in LTM. The priority of each object is based on the hierarchy level (vertical order) of that object in the object hierarchy, and on an additional number which specifies the priority of that object on that hierarchy level (horizontal order). Because vertical order must prevail over horizontal order to ensure proper instantiation of complex objects, all the priorities of all objects on a hierarchy level must be greater than the priorities of all objects on the hierarchy level above it. The priority of each object is either specified in the knowledge file or it can be determined automatically when building the object hierarchy. The automatic determination of priorities provides a correct vertical order of priorities in the hierarchy, but the horizontal order is chosen arbitrarily. Hypotheses are enqueued in the generation step, and served from the priority queue in the selection step to select a working hypothesis. The use of a priority queue ensures that lower-level objects are verified and instantiated before higher-level objects, making the instantiation of objects strictly bottom-up. In our application we have chosen the Splay-Queue implementation [9]. A Splay Queue has $O(\log d)$ complexity for both the serve and enqueue operation, with $d$ the number of distinct priorities, and is inherently stable and has a directly available implementation.

Hypothesis verification  The working hypothesis, which is selected in the selection step, is verified in the verification step. The verification step tries to instantiate the alternatives in the list of alternatives of the working hypothesis. The verification of a hypothesis concerning the existence of a simple object is successful when one or more simple objects are detected in the search areas of the alternatives of the hypothesis. The alternatives are processed by calling the detector or searching the STM for each alternative. If a simple object is detected in the bootstrap procedure it no longer has to be detected but can be retrieved from STM. The search area for an alternative is inherited from its parent concept by a part or specialization link, or set by the pruning function of a relationship which the simple object has with another object. Successful verification results in instantiations of simple objects, which are put into the list of instantiations of the working hypothesis. If all alternatives of the hypothesis are processed, the instantiations are moved to the STM.

Verification of a hypothesis on the existence of a complex object is successful when one or more alternatives can be instantiated. An alternative can be instantiated if all parts (or one specialization) of the alternative are instantiated, geometrical relationships fulfilled, and the overall judgment value of the complex object is non-zero. The calculation of a judgment value of a complex object is discussed in Section 5.5. If all parts (or one specialization) of an alternative have not been instantiated yet, verification is not successful and the consecutive generation step will generate hypotheses on the missing parts or a hypothesis on the missing specialization. The alternative is kept in the list of alternatives. If an alternative has all parts or one specialization instantiated and it has a zero judgment value then it is removed from the list of alternatives. If all alternatives are removed and no instantiations are made, the working hypothesis is deleted and a new working hypothesis is selected after skipping the generation step. A successful verification of
the hypothesis on a complex object results in one or more instantiations of the complex object. The resulting instantiations are stored in the list of instantiations of the working hypothesis. If all alternatives are processed it is checked whether or not the resulting instantiations contain conflicting objects, according to the specification of conflicts in the conflict-rule base. The conflict-rule base also specifies whether conflict resolution must be applied at this hierarchy level, which is set by the level of the working hypothesis. If the instantiations contain conflicting objects and the conflict-rule base specifies that those conflicts must be resolved at this hierarchy level, then conflict resolution is applied. Otherwise conflict resolution is skipped or postponed until the right hierarchy level is reached. Conflict resolution can result in the removal of one or more instantiations; it is discussed in more detail in Section 5.6.1. After conflict resolution the remaining instantiations are moved to the STM.

**Hypothesis generation** Because we use object hierarchies, hypotheses on the existence of other objects can be generated in two ways: bottom-up or top-down. The way in which hypotheses are to be generated depends on the result of the verification of the working hypothesis in the preceding verification step: bottom-up if it was successful, top-down if it was not.

**top-down hypothesis generation**: The verification step was not successful because the alternatives of the working hypothesis miss some instantiated parts or one instantiated specialization. For each alternative of the working hypothesis, the existence of missing parts or a missing specialization, which are lower-level objects, are hypothesized. The generation of hypotheses is along the part or specialization links of the corresponding LTM concept. Each alternative is extended with the hypothesized parts or specialization by adding corresponding modified concepts as parts or specialization. An example of top-down generation of an alternative is given in Figures 5.14 and 5.15. The attribute restrictions of the alternative are copied to the generated parts or specialization. If an alternative already has one or more instantiated parts that have a relationship with the generated modified concepts, the pruning functions of the relationship are applied to restrict the attribute restrictions of the generated parts further. The hypothesized parts or specialization are enqueued on the priority queue. Additionally, the working hypothesis is enqueued again so it can be verified later when its lower-level objects have been instantiated.

**bottom-up hypothesis generation**: The existence of higher-level objects is hypothesized after the instantiations of the concept of the working hypothesis in the successful verification step. The generation of hypotheses is along the part of or specialization of links of the corresponding LTM concept of the working hypothesis. Had the working hypothesis been generated in a top-down fashion, the hypothesis to be generated, the parent hypothesis, would already exist. In this case, alternatives for the parent hypothesis are generated instead of a new hypothesis. Each instantiation of the working hypothesis results in one alternative for the parent concept, in which the modified concept of the working hypothesis is replaced with an instantiation. The generated alternatives only differ in the instantiations. Because the working hypothesis was generated in
Figure 5.11: Interpretation-mechanism example: (a) gray-scale image of a part of a utility map that is the input to the interpretation process. (b) Two detected double-headed arrows.

In a top-down fashion, it is possible that the instantiations have a relationship with other (hypothesized) concepts. In that case, to each alternative the pruning functions of the relationship are applied to restrict the attribute values of the other concepts. If the working hypothesis was generated in a bottom-up fashion then the parent hypothesis does not exist before now. In this case, a new hypothesis on the existence of a higher-level object is generated that follows the part of or specialization of link of the corresponding LTM concept of the working hypothesis. For each instantiation of the working hypothesis a new hypothesis on the parent concept is generated and enqueued on the priority queue. Each generated parent hypothesis has one alternative that has an instantiation as part or specialization.

Interpretation-mechanism example

We take as a (simplified) example the interpretation of the concept dimension (Figures 5.2 and 5.3) within a part of a utility map, see Figure 5.11. The interpretation starts by a bootstrap procedure that finds all instantiations of double- and single-headed arrows by means of their detectors. All resulting instantiations are enqueued as (verified and instantiated) hypotheses that have a non-zero judgment. These hypotheses have an empty alternative list and one instantiation, and a status that reflects the verified and instantiated nature of the hypotheses. Figure 5.12 depicts the case when we have detected two double-headed arrows in the input image: two hypotheses are enqueued and in the next step, hypothesis selection, one of the double-headed arrows is selected as the working hypothesis.
Figure 5.12: Start of interpretation: a bootstrap procedure finds all instantiations of double- and single-headed arrows, which are enqueued as hypotheses: the detected double-headed arrows from Figure 5.11(b) are enqueued and one of the arrows is selected as working hypothesis in the hypothesis-selection step.

After the bootstrap procedure, the interpretation mechanism selects one of the double-headed arrows as working hypothesis by serving it from the priority queue. Because the working hypothesis is already verified and instantiated by the detector, the verification step is skipped and the consecutive generation step is executed. As the working hypothesis is successfully verified by the detector, the generation step is executed in a bottom-up way: the instantiation of a double-headed arrow hypothesizes the existence of the higher-level object arrow. Because the working hypothesis was not generated in a top-down fashion a new hypothesis on the existence of an arrow concept is generated and enqueued, as depicted in Figure 5.13. The generated parent hypothesis has one alternative that has an instantiation of

Figure 5.13: Bottom-up hypothesis generation over the specialization of link of the instantiations of a double-headed arrow: new hypotheses on the existence of arrow concepts are enqueued. One of the two arrow concepts becomes the new working hypothesis.

- a double-headed arrow as a specialization. In the next interpretation cycle, the other instantiation of a double-headed arrow is processed, resulting in one more hypothesis on the existence of an arrow concept.

Next, one of the hypotheses on the existence of an arrow is selected as the working hypothesis. Verification of this hypothesis is performed by checking whether its alternative, a modified-concept arrow, has one instantiated specialization. Because it has one specialization (double-headed arrow), it can be instantiated. After the
5.3 Interpretation

Instantiation, its attributes are calculated and the instantiation is judged. We assume that the vector attribute value of the instantiation falls within the (initial) restriction and the overall judgment of the instantiation is non-zero. As a result, the hypothesis is successfully verified and the instantiation is moved to the list of instantiations of the working hypothesis. The conflict-resolution step is skipped because there is only one instantiation, which is moved to STM. As the working hypothesis is successfully verified, the generation step is executed in a bottom-up way: the instantiation of an arrow hypothesizes the existence of the higher-level dimension object. Because the working hypothesis was generated in a bottom-up way, a new hypothesis on the existence of a dimension concept is generated and enqueued, as depicted in Figure 5.14. The generated hypothesis has one alternative

![Diagram](image)

**Figure 5.14:** Successful verification of the hypothesis on the existence of an arrow results in the bottom-up hypothesis generation over the part of link of the instantiations of an arrow: new hypotheses on the existence of dimension concepts are enqueued. The dimension concept becomes the new working hypothesis.

That has an instantiation of an arrow as a part. In the next interpretation cycle, the other hypothesis on the existence of an arrow is processed, resulting in one more instantiation of the arrow concept and one more hypothesis on the existence of a dimension concept.

In the next interpretation cycle, the hypothesis dimension is selected as the working hypothesis. Because its alternative does not have all of its parts (house side and conduit) instantiated, it cannot be instantiated and verified successfully. Hence, in the generation step, the alternative is extended by adding modified concepts house side and conduit at the position of its non-instantiated parts. Cor-
responding hypotheses on the existence of *house side* and *conduit* objects are enqueued, as shown in Figure 5.15. The generated hypotheses have a modified concept *house side* or *conduit* as alternative. Because the concept *arrow* has a binary geometrical relationship (perpendicular) with the concept *conduit* the pruning function of the geometrical relationship is invoked: the search area of the modified concept *conduit* is calculated from the vector-attribute value of the *arrow* instantiation, see Figure 5.15. The pruning function of the tertiary in-line relationship between a *conduit*, an *arrow* and a *house side* cannot be applied because it requires an instantiated *conduit*. The hypothesis *dimension* is also enqueued so it can be verified later after its parts have been instantiated. In the next interpretation cycle, the other hypothesis on the existence of a *dimension* is processed, resulting in more hypotheses on the existence of a *house side*, *conduit*, and a *dimension*.

One of the hypotheses *conduit* is selected in the following cycle as it has the highest priority. A *conduit* concept is a simple object and has a corresponding detector. Therefore, hypothesis verification is performed by calling its detector for each alternative of the working hypothesis to operate on its pruned search area. Application of the detector for the single alternative of the working hypothesis results in two detections and therefore two instantiations of the *conduit* concept: verification is successful. The detected *conduits* are shown in Figure 5.16. Because the verification step was successful the following generation step is executed bottom-up. However, as the parent hypothesis, i.e. *dimension*, already exists, no new hypotheses are generated, but alternatives are made for the parent hypothesis. Each instantiation of the *conduit* concept makes an alternative for the modified concept *dimension* which has an instantiated *conduit* as a part. For each generated alternative, the pruning function of the tertiary in-line relationship between a *conduit*, an *arrow*, and a *house side* is invoked to specify a search area for the *house side* concept, see Figure 5.16. The same procedure is applied in the next cycle for the detection of two other *conduits* for the second *dimension*.

Verification of the *house side* concept is performed in the same way as the *conduit*; by calling the detector for each alternative of the working hypothesis to operate on its pruned search area. Application of the detector for the working hypothesis results in one detection and, therefore, one instantiation of the *house side* concept: verification is successful. The detected *house side* is shown in Figure 5.17. Because the verification step was successful, the following generation step is executed bottom-up. However, as the parent hypothesis, i.e. *dimension*, already exists, so no new hypotheses are generated, but both its alternatives are extended with the instantiated *house side*. The same procedure is applied in the next cycle for the detection of two other *house sides* for the second *dimension*.

After the verification of the *house side*, all parts of the *dimension* are instantiated and in the next selection step we select the hypothesis *dimension* for the second time. The verification of the two alternatives for the concept *dimension* leads to two instantiations of the concept *dimension*. If the conflict-rule file (see Section 5.6.1) specifies that this is a conflicting situation (because two instantiations of a *dimension* share the same *arrow* or *house side*), conflict resolution is applied to the two conflicting instantiations. Conflict resolution is discussed in more detail in Section 5.6.1, but basically it chooses one of the two conflicting instantiations as the right instantiation and the other instantiation is deleted.
Figure 5.15: Top-down hypothesis generation: The existence of the conduit and house side concepts are hypothesized using the part links of the LTM concept dimension. (a) Applications of the standard pruning function vector end points \( r, b, l, f \) to restrict the attributes of the modified concepts conduit on the basis of the vector attributes of the arrow instantiations. (b) The single alternative of one of the dimension hypotheses with its generated house side and conduit parts. (c) The generated parts and the hypothesis dimension are enqueued. One of the two conduits is the next working hypothesis.
Figure 5.16: Bottom-up alternative generation: The detection of two conduits results in two alternatives for a hypothesis dimension. (a) Detected conduits and applications of the standard pruning function vector end point(r, b, l, f) to restrict the attributes of the modified concepts house side on the basis of the vector attributes of the conduit and arrow instantiations. (b) One of the two generated alternatives of the hypothesis dimension. (c) One of the two house sides is the next working hypothesis.
Figure 5.17: Verification of the house side concept and the dimension concept results in two dimension instantiations. (a) Detected house sides. (b) A dimension instantiation. (c) Queue is empty: interpretation has finished.
5.3.4 Hashing on geometrical attributes

During interpretation, the STM is continuously accessed for storage and retrieval of instantiations. Typical STM operations include search actions for certain instantiations in a given search area and the storage of instantiations on a certain position. When the data structure of STM is implemented as a simple linked list (SLL) [107], the algorithmic complexity of a typical search operation in the STM is $O(n)$, with $n$ the number of elements in STM. In order to speed up search operations, we apply hashing on geometrical attributes [77]. Hashing generally reduces the complexity of a typical search operation from $O(n)$ to $O(1)$ [107] and can be done on one or more (geometrical) key attributes: position, orientation, etc. One drawback of hashing is that it requires data structures that have high memory requirements. In practical applications, usually a trade-off is found between increase of speed of operations and availability of memory resources. In our utility-map interpretation application,

![Diagram](image)

**Figure 5.18:** Hashing on geometrical attributes. (a) Position of a virtual grid on a utility map. (b) Hashing reduces total interpretation time. The horizontal axis depicts the square number of grids; the vertical axis depicts total STM accessing time (measured in minutes) averaged over the interpretation of three utility maps.

hashing is implemented by placing a (virtual) grid over the utility map. An example of a grid on top of a utility map is given in Figure 5.18(a). Each grid cell contains a number of simple linked lists: one SLL for each LTM concept. Each list contains the instantiations of one LTM concept. The pointers to the simple linked lists for one grid cell are stored within an array which is indexed by the name string of the LTM concept. From Figure 5.18(b) we can see that for $16 \times 16$ grids and more we have an enormous decrease in STM accessing time. The use of $16 \times 16$ grids corresponds to using grids of size $1024 \times 1024$ pixels (our utility maps have sizes up to $16,000 \times 16,000$).
5.4 Detectors

The detectors of the simple objects serve as the interface between the low-level image processing and the intermediate- or high-level reasoning; they shift the focus from pixels to map semantics. Detectors are designed to recognize a simple object on the basis of the gray-scale input image, or any information source derived from this image: binary image, graphical primitives, vectorization, feature vectors, etc. To tune each detector in its parameter space, the BESSI toolkit [17, 18, 105] can be used. Our view on the concept of detectors is an object-oriented (programming) one, as proposed by Smeulders and Ten Kate [105]: the detector interface is fixed, facilitating separate design and easy replacement by new versions. Section 5.4.1 discusses experiments with different single detectors for simple map objects.

Our view on detectors is not only new in that it is object oriented; we also propose that in order to reduce the number of false positive detections, multiple detectors are to be used to generate instantiations of simple objects. Each single detector accompanies the detection of a simple object with a measure of confidence or judgment value. During interpretation, several of these detectors are combined into one judgment when the system evaluates an instantiation of a simple object. Because the interpretation of this judgment value can vary over different detectors, an ignorant combination of these measures yields no sound interpretation. Some detectors may give an approximation of a conditional probability as a measure, while others will provide a membership to a fuzzy set or a distance value from the model template. Furthermore, one detector is not as good as any other: we may have to weigh the judgment value of each detector. In order to fuse judgment values of different detectors into one judgment of a simple map object, we compared the use of different strategies. Experiments on them are discussed in Section 5.4.2.

5.4.1 Single-detector experiments

For the different map objects that exist in our utility-map application domain, we have developed different detectors that use different image-processing and pattern-recognition techniques. The time and effort put into the development of a specialized detector for a map object is based on the average frequency of the map object in a utility map and its importance to the interpretation. As such, the arrow map object has a far more sophisticated detector (which incorporates several techniques, one being [24]) than all the other detectors because of its high frequency of occurrence. Furthermore, some map objects are so morphologically distinct from other map objects that it is relatively easy to develop their detectors. Examples of those map objects are a connection, which is drawn as a large disk, and a conduit, which is drawn as a collection of thick line pieces. Other map objects need complex and time-consuming detectors. For instance, some map objects are drawn on the back of the utility maps and are hard to recognize. Besides a specialized detector for each map object, we have also developed (simple) detectors for each map object which use several well-known classifying algorithms such as artificial neural networks or (statistical) pattern-recognition techniques. These classifying algorithms take as input a (feature) vector with attributes of a graphical primitive, see Section 5.3.1. We have distinguished eight (simple) map objects: arrows (single- and double-headed),
<table>
<thead>
<tr>
<th>Map object</th>
<th>Special</th>
<th>NN</th>
<th>k-NN</th>
<th>ANN 14</th>
<th>ANN 16</th>
<th>ANN 18</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>digit</td>
<td>98.7</td>
<td>98.6</td>
<td>95.3</td>
<td>99.3</td>
<td>98.7</td>
<td>99.1</td>
<td>772</td>
</tr>
<tr>
<td>decimal dot</td>
<td>100.0</td>
<td>96.4</td>
<td>80.4</td>
<td>97.3</td>
<td>96.4</td>
<td>96.4</td>
<td>178</td>
</tr>
<tr>
<td>s.-h. arrow</td>
<td>56.4</td>
<td>38.5</td>
<td>46.2</td>
<td>64.1</td>
<td>76.9</td>
<td>74.3</td>
<td>55</td>
</tr>
<tr>
<td>dash</td>
<td>82.3</td>
<td>86.1</td>
<td>76.6</td>
<td>97.6</td>
<td>96.2</td>
<td>97.1</td>
<td>296</td>
</tr>
<tr>
<td>d.-h. arrow</td>
<td>92.4</td>
<td>90.9</td>
<td>66.7</td>
<td>81.8</td>
<td>78.8</td>
<td>81.8</td>
<td>91</td>
</tr>
<tr>
<td>arrow head</td>
<td>56.3</td>
<td>59.4</td>
<td>62.5</td>
<td>87.5</td>
<td>90.6</td>
<td>81.3</td>
<td>63</td>
</tr>
<tr>
<td>conduit piece</td>
<td>96.9</td>
<td>99.0</td>
<td>98.4</td>
<td>100.0</td>
<td>99.5</td>
<td>96.9</td>
<td>254</td>
</tr>
<tr>
<td>connection</td>
<td>95.5</td>
<td>100.0</td>
<td>86.4</td>
<td>100.0</td>
<td>97.7</td>
<td>100.0</td>
<td>52</td>
</tr>
</tbody>
</table>

Table 5.3: Percentage of correctly classified map objects; the *Special* detectors are the detectors developed by project partners, *NN* and *k-NN* are the nearest neighbor and k-nearest neighbor classifiers (optimized w.r.t. *k*). *ANN* 14 to 18 are the detectors with feed-forward network implementations, the number indicating the number of hidden units in the hidden unit layer. The column marked *Total* gives the total number of each map object in the test set.

<table>
<thead>
<tr>
<th>Map object</th>
<th>Special</th>
<th>NN</th>
<th>k-NN</th>
<th>ANN 14</th>
<th>ANN 16</th>
<th>ANN 18</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>digit</td>
<td>55</td>
<td>29</td>
<td>53</td>
<td>46</td>
<td>31</td>
<td>38</td>
<td>772</td>
</tr>
<tr>
<td>decimal dot</td>
<td>75</td>
<td>32</td>
<td>72</td>
<td>69</td>
<td>60</td>
<td>48</td>
<td>178</td>
</tr>
<tr>
<td>s.-h. arrow</td>
<td>42</td>
<td>25</td>
<td>14</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>55</td>
</tr>
<tr>
<td>dash</td>
<td>149</td>
<td>45</td>
<td>103</td>
<td>119</td>
<td>107</td>
<td>64</td>
<td>296</td>
</tr>
<tr>
<td>d.-h. arrow</td>
<td>30</td>
<td>18</td>
<td>10</td>
<td>9</td>
<td>6</td>
<td>23</td>
<td>91</td>
</tr>
<tr>
<td>arrow head</td>
<td>17</td>
<td>9</td>
<td>20</td>
<td>13</td>
<td>10</td>
<td>8</td>
<td>63</td>
</tr>
<tr>
<td>conduit piece</td>
<td>3</td>
<td>20</td>
<td>28</td>
<td>32</td>
<td>8</td>
<td>5</td>
<td>254</td>
</tr>
<tr>
<td>connection</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>52</td>
</tr>
</tbody>
</table>

Table 5.4: Number of false positives for each detector.

*arrow heads, dimension digits, decimal dots, dashes, conduit line pieces, and connections.* The artificial neural networks that we have used in our experiments were feed-forward networks with a varying number of hidden units in one hidden unit layer. The feature vector of a graphical primitive fed into an ANN includes 23 features, such as: area, length, minimum width, maximum width, average width, minimum-area encasing rectangle features, etc. Feature values were normalized to their means and their standard deviations in the learning set of examples. The networks have eight outputs, corresponding to the eight simple map objects. We used a learning set with 1822 examples and a test set with 1761 examples. Both sets cover all classes and were collected from a set of four utility maps. Table 5.3 gives the classification results and Table 5.4 gives the number of false positive detections of the different map object detectors.

From the experiment we can conclude that the detectors for conduits and connections have high performance rates and a low number of false positive detections. Detectors of other map objects combine high performance rates with high numbers of false positive detections. Because false positive detections are more expensive than false negative detections, this is not desirable. False positive detections have to be detected and solved by conflict resolution in the interpretation phase or, if
5.4 Detectors

<table>
<thead>
<tr>
<th>Map object</th>
<th>Average</th>
<th>Majority voting</th>
<th>Choquet integral</th>
<th>ANN</th>
<th>ANN</th>
<th>ANN</th>
<th>Total</th>
<th>Single</th>
</tr>
</thead>
<tbody>
<tr>
<td>digit</td>
<td>99.1</td>
<td>98.9</td>
<td>98.9</td>
<td>99.3</td>
<td>99.1</td>
<td>772</td>
<td>99.3</td>
<td>0</td>
</tr>
<tr>
<td>decimal dot</td>
<td>96.4</td>
<td>96.4</td>
<td>96.4</td>
<td>97.3</td>
<td>96.4</td>
<td>178</td>
<td>100.0</td>
<td>+</td>
</tr>
<tr>
<td>s.-h. arrow</td>
<td>56.4</td>
<td>51.3</td>
<td>68.2</td>
<td>59.0</td>
<td>66.7</td>
<td>61.6</td>
<td>55</td>
<td>76.9</td>
</tr>
<tr>
<td>dash</td>
<td>97.1</td>
<td>96.7</td>
<td>90.4</td>
<td>76.1</td>
<td>73.7</td>
<td>72.2</td>
<td>296</td>
<td>97.6</td>
</tr>
<tr>
<td>d.-h. arrow</td>
<td>81.8</td>
<td>77.3</td>
<td>86.4</td>
<td>84.8</td>
<td>83.3</td>
<td>91</td>
<td>92.4</td>
<td>-</td>
</tr>
<tr>
<td>arrowhead</td>
<td>81.3</td>
<td>84.4</td>
<td>84.4</td>
<td>84.4</td>
<td>84.4</td>
<td>63</td>
<td>90.6</td>
<td>-</td>
</tr>
<tr>
<td>conduit piece</td>
<td>99.0</td>
<td>99.0</td>
<td>98.4</td>
<td>99.0</td>
<td>99.5</td>
<td>99</td>
<td>254</td>
<td>100.0</td>
</tr>
<tr>
<td>connection</td>
<td>95.5</td>
<td>97.7</td>
<td>100.0</td>
<td>86.4</td>
<td>93.2</td>
<td>52</td>
<td>100.0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5.5: Percentage of correctly classified map objects for different combining techniques. ANN 2 to 8 are feed-forward neural network implementations, the number indicating the number of hidden units in the hidden unit layer. The column marked Total gives the total number of examples of each map object in the test set.

<table>
<thead>
<tr>
<th>Map object</th>
<th>Average</th>
<th>Majority voting</th>
<th>Choquet integral</th>
<th>ANN</th>
<th>ANN</th>
<th>ANN</th>
<th>Total</th>
<th>Single</th>
</tr>
</thead>
<tbody>
<tr>
<td>digit</td>
<td>29</td>
<td>28</td>
<td>27</td>
<td>3</td>
<td>36</td>
<td>35</td>
<td>772</td>
<td>29</td>
</tr>
<tr>
<td>decimal dot</td>
<td>51</td>
<td>52</td>
<td>45</td>
<td>50</td>
<td>178</td>
<td>32</td>
<td>19</td>
<td>-</td>
</tr>
<tr>
<td>s.-h. arrow</td>
<td>10</td>
<td>8</td>
<td>20</td>
<td>19</td>
<td>44</td>
<td>15</td>
<td>35</td>
<td>45</td>
</tr>
<tr>
<td>dash</td>
<td>81</td>
<td>84</td>
<td>5</td>
<td>35</td>
<td>256</td>
<td>45</td>
<td>19</td>
<td>+</td>
</tr>
<tr>
<td>d.-h. arrow</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>49</td>
<td>6</td>
<td>254</td>
<td>3</td>
</tr>
<tr>
<td>arrowhead</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>63</td>
<td>8</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>conduit piece</td>
<td>6</td>
<td>4</td>
<td>1</td>
<td>14</td>
<td>16</td>
<td>0</td>
<td>52</td>
<td>0</td>
</tr>
<tr>
<td>connection</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.6: The number of false positive detections for different combining techniques.

not detected, must be deleted and redrawn by the operator, whereas false negative detections only have to be redrawn by the operator. Furthermore, we may conclude that the choice of a detector depends on the map object.

5.4.2 Multiple-detector experiments

This section describes experiments with several combination functions for single detectors. The combination of classifiers for character- and digit-recognition purposes recently received a lot of attention in the literature [38,53,57,59,68,87,88]. It is assumed that the judgment values of the different single detectors are represented by real numbers in the interval [0,1]. The value 0 represents the worst judgment or no confidence, whereas the value 1 represents the best judgment or total confidence. Furthermore, we assume that judgment values are ordered, implying that if a judgment value for a possible instantiation i of object A (JV(A_i)) is smaller than a judgment value for another instantiation j of object A (JV(A_j)), we have less confidence in instantiation A_i than in instantiation A_j. The contribution of combining different single detectors lies in the property that different detectors possibly make different mistakes and that a combined classifier can compensate for these mistakes [60,115]. Another pleasant aspect of a combined classifier is that the building blocks may be weak classifiers while their combined effort stays strong. In this way, a tedious and difficult optimization scheme might be avoided. In order to fuse judgment values of different single detectors into one judgment of a simple map object, we compared the use of several combining strategies: averaging, a ma-
majority voting scheme (after a forced classification for each single detector), Choquet integral [38, 42], and implementations of feed-forward artificial neural networks with one hidden unit layer. Table 5.5 gives the classification results and Table 5.6 gives the number of false positive detections for different combination techniques.

The experiments show that combination techniques do not improve the classification rate. For instance, the single detector for single-headed arrows which is implemented as an ANN with 16 hidden units has a better performance than any combination function. Nevertheless, combination techniques do reduce the number of false positive detections for other map objects. For instance, the Choquet integral reduces the number of false positives in case of dashes with comparable performance.

5.5 Judgment of complex objects

After discussing the judgment of simple objects in the previous section, we now focus on the judgment procedure of complex objects in this section. In our interpretation system, instantiations of a complex object are judged so a choice can be made between different instantiations if an interpretation conflict is apparent. As suggested in [78], judgment of a complex object is a combination function which takes the judgment of its parts or specialization and the judgment of its relationships as function arguments. Each argument has a different weight, which stresses its importance to the combination. Judgments of simple objects, i.e. the leaves in the object hierarchy, are given by the corresponding detectors (or combination of detectors) of that simple object. The judgment value of a standard (geometrical) relationship of a complex object is calculated from approximations of the conditional probability density functions of the standard relationship given the deviations of the parameters of the relationship, see Section 5.3.1. The calculation of a judgment value of a special (geometrical) relationship of a complex object is discussed in more detail in Section 5.5.1. Common techniques or uncertainty models for the computation of a judgment value of a complex object include the following:

- Probabilistic approaches:
  - Certainty-Factor model [104]
  - Bayesian belief networks [82]
  - Dempster-Shafer theory [22]

- Possibilistic approaches:
  - fuzzy logic [120]
  - fuzzy-set theory [121]
  - possibility theory [122]

A common drawback of all these uncertainty calculi is that they imply a syntax and semantics: the model prescribes a combination function for uncertainty and certain semantics of the uncertainty value. The learning phase of these calculi generally consists of estimating the (conditional) probability density functions from labeled
5.5 Judgment of complex objects

Figure 5.19: (a) The representation of a geometrical relationship between two objects uses the minimum-area encasing rectangles of the objects. (b) (c) Examples of geometrical relationships.

examples or manual specification by domain experts. We overcome this drawback by viewing each combination of judgment values as a unique case and treat the judging as a classification problem solvable by applying pattern-recognition techniques to adjust to this specific case by learning from examples without to be bothered by uncertainty model restrictions. In the following sections, we discuss our method to compute the judgment value of a complex object. Section 5.5.1 discusses the judgment of special geometrical relationships between objects; Section 5.5.2 discusses experiments of judging instantiations of complex objects.

5.5.1 Judgment of special (geometrical) relationships

For the judgment procedure of a special geometrical relationship, the relationship between two objects is represented with four parameters: \( d, dx, dy, \) and \( \alpha \), which are based on the minimum-area encasing rectangle of the two objects, see Figure 5.19(a). These parameters are with respect to the local coordinate system of the reference object, i.e. the first argument of the geometrical relationship, and are defined as follows:

- \( dx \) : translation along the \( x \)-axis (in pixels) between the points of gravity of the MAER of object 1 and the MAER of object 2.

- \( dy \) : translation along the \( y \)-axis (in pixels) between the points of gravity of the MAER of object 1 and the MAER of object 2.

- \( \alpha \) : difference in orientation angle (in radians) of the MAER of object 1 and the MAER of object 2. The difference in orientation angle is taken as the angle between the longest side of the MAER of object 2 and the shortest side of the MAER of object 1.

- \( d \) : the shortest distance (in pixels) between the MAERs of the two objects.
<table>
<thead>
<tr>
<th>Geometrical relationship</th>
<th>NN</th>
<th>k-NN</th>
<th>ANN 0</th>
<th>ANN 2</th>
<th>ANN 4</th>
<th>ANN 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_1$: digit to digit</td>
<td>97.9</td>
<td>97.9</td>
<td>96.9</td>
<td>96.9</td>
<td>97.9</td>
<td>96.9</td>
</tr>
<tr>
<td>$R_2$: digit to dot</td>
<td>82.4</td>
<td><strong>100.0</strong></td>
<td>82.4</td>
<td>88.2</td>
<td>94.1</td>
<td>94.1</td>
</tr>
<tr>
<td>$R_3$: digit to digit</td>
<td><strong>100.0</strong></td>
<td>100.0</td>
<td>78.9</td>
<td>89.5</td>
<td>89.5</td>
<td>89.5</td>
</tr>
<tr>
<td>$R_4$: arrow to conduit</td>
<td>96.2</td>
<td>98.7</td>
<td><strong>100.0</strong></td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>$R_5$: arrow to house</td>
<td><strong>100.0</strong></td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Table 5.7: Percentage of correctly classified geometrical relationships given different judging techniques.

<table>
<thead>
<tr>
<th>Geometrical relationship</th>
<th>NN</th>
<th>k-NN</th>
<th>ANN 0</th>
<th>ANN 2</th>
<th>ANN 4</th>
<th>ANN 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_1$: digit to digit</td>
<td>5</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>$R_2$: digit to dot</td>
<td>2</td>
<td>9</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$R_3$: digit to digit</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$R_4$: arrow to conduit</td>
<td>13</td>
<td>14</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>$R_5$: arrow to house</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 5.8: Number of false positive classified geometrical relationships given different judging techniques.

Notice that for some geometrical relationships which have a certain map object as argument, e.g., a connection or a decimal dot, the parameter $\alpha$ is obsolete. To judge geometrical relationships we applied pattern-recognition techniques with as feature vector the normalized (using the training set) $d$, $dx$, $dy$, and $\alpha$ parameters. Table 5.7 gives the classification results and Table 5.8 gives the number of false positive classifications. From the results, we may conclude that the classification of all geometrical relationships has a high classification rate and a low number of false positive classifications. As a result, no additional combining techniques are necessary.

### 5.5.2 Judgment of complex objects

For the judgment of instantiations of complex map objects, we have to combine judgments of lower-level map objects and geometric relationships. So, in this judgment, pixel info and contextual information is combined. Notice that in this combination all parts and relations are important. If only one of them is low in confidence there is enough reason to believe that the judgment of the complex instance should also be low in confidence, as a part may be missing or a geometrical relationship may be unfulfilled. This poses different restrictions on the combination techniques than multiple detectors do. Table 5.9 gives results for dimensions that occur most frequently in our utility map domain; the number of false positives are given in Table 5.10. We have used the following classifiers:

- **Linear classifier**: A linear classifier that is based on normal densities and Bayes rule.
- **Quadratic classifier**: A quadratic classifier that is based on normal densities and Bayes rule.
5.6 Interpretation conflicts

<table>
<thead>
<tr>
<th>Complex map object</th>
<th>Linear</th>
<th>Quadratic</th>
<th>k-NN</th>
<th>Parzen</th>
<th>Max</th>
<th>Min</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>dimension house ↔ conduit</td>
<td>60</td>
<td>13</td>
<td>40</td>
<td>40</td>
<td>100</td>
<td>27</td>
<td>100</td>
</tr>
<tr>
<td>dimension conduit ↔ arrow</td>
<td>83</td>
<td>78</td>
<td>100</td>
<td>83</td>
<td>100</td>
<td>33</td>
<td>100</td>
</tr>
<tr>
<td>dimension conduit ↔ dashed line</td>
<td>67</td>
<td>100</td>
<td>33</td>
<td>x</td>
<td>100</td>
<td>x</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 5.9: Percentage of correctly judged complex map objects given different judging techniques.

<table>
<thead>
<tr>
<th>Complex map object</th>
<th>Linear</th>
<th>Quadratic</th>
<th>k-NN</th>
<th>Parzen</th>
<th>Max</th>
<th>Min</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>dimension house ↔ conduit</td>
<td>16</td>
<td>0</td>
<td>6</td>
<td>6</td>
<td>69</td>
<td>0</td>
<td>69</td>
</tr>
<tr>
<td>dimension conduit ↔ arrow</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>8</td>
<td>74</td>
<td>4</td>
<td>73</td>
</tr>
<tr>
<td>dimension conduit ↔ dashed line</td>
<td>4</td>
<td>9</td>
<td>0</td>
<td>x</td>
<td>21</td>
<td>x</td>
<td>21</td>
</tr>
</tbody>
</table>

Table 5.10: Number of false-positive judged complex map objects given different judging techniques.

k-NN: k-nearest neighbor classifier optimized w.r.t. the parameter k.

Parzen classifier: A classifier that is based on Parzen densities and Bayes rule.

Max, Min, and Mean: A classifier that selects the maximum, minimum, or mean value of its inputs.

From the experiments we may conclude that the minimum operator has the best performance in terms of false positives, whereas the max and mean give the highest classification rates. The linear classifier combines the highest performance rate with a low number of false positives. The number of false positives can be further reduced by conflict resolution, as discussed in the next section.

5.6 Interpretation conflicts

During interpretation, conflicts between different instantiations of complex objects may arise. By interpretation conflicts we mean the notion that instantiations of simple or complex objects are part of (or specialization of) two or more different complex objects higher in the object hierarchy. Figure 5.20 gives an example of an interpretation conflict between two instantiations of the concept dimension. The two instantiations share the same arrow and house side instantiations but have have different instantiations of the conduit concept. Interpretation conflicts can have different causes:

1. Because of the applied tolerances in the instantiations of geometrical relationships, unrelated objects can become related. This is particularly true in dense and complex map situations.

2. The false positive detection of simple objects can cause interpretation conflicts higher in the object hierarchy. In this case, wrong instantiations of complex objects are generated on the basis of non-existent simple objects.

3. Application-domain specification.
Figure 5.20: Interpretation conflict: two instantiations of the concept \textit{dimension} have the same \textit{house side} and \textit{arrow} as parts. This conflict is not a drawing conflict because it is due to the specification of the application domain. (a) First instantiation of concept \textit{dimension}. (b) Second instantiation. (c) Both instantiations share the same \textit{house side} and \textit{arrow}, but have different instantiations of the \textit{conduit}. 
Figure 5.21 gives an example of the first cause of an interpretation conflict. Figure 5.21(a) shows an instantiation of a \textit{dimension} that is in conflict with the dimension shown in Figure 5.21(b). Both instantiations have the same \textit{arrow} and \textit{house side} as parts. The \textit{dimension} of Figure 5.21(a) is less probable than the \textit{dimension} of Figure 5.21(b) because the \textit{dimension leader} is closer in distance to the \textit{arrow} than the \textit{conduit} is. Because of the applied tolerances in the geometrical relationship between map objects, the relationship between the \textit{arrow} and the \textit{conduit} is still fulfilled, albeit with a low judgment value. This interpretation conflict must be resolved to prevent that during the map-reconstruction phase (Chapter 6) the \textit{conduit} and \textit{dimension leader} are placed on top of one another or overlap.

Due to the rules by which utility maps are drawn by the technical draftsmen of the public-utility organization and the specification of the application-domain, not all interpretation conflicts are conflicts in the drawing sense. In some map situations, a map object can be used multiple times by different instantiations of higher-level map objects. For example, Figure 5.20 shows an interpretation conflict between two conflicting instantiations of a \textit{dimension} that share the same \textit{house side} and \textit{arrow} but are not in conflict, so nothing needs to be resolved in this case. In order to cope with these \textit{context-dependent} conflicts we propose the use of so-called \textit{conflict rules} that explicitly state all possible conflicts between map objects. Figure 5.22 gives a (simplified) example of a file with a specification of a number of conflict rules.
5.6.1 Conflict resolution

In the previous section, we explained what we call an interpretation conflict and argued for the use of a conflict-rule file to distinguish between map situations that are conflicting in the drawing sense and map situations that are not. In this section, we discuss the resolution of interpretation conflicts: conflict resolution. Conflict resolution is mentioned sparsely in the literature. In some cases, it is called contextual-inconsistency detection [23] or external-incoherency resolution [79]. The main cause of inconsistencies or incoherences is the clash of pixel-based information with context-based information. In an abstract paper, Joseph et al. touch on the concept of conflict resolution [51] when they state: "It seems reasonable to expect that conflict resolution might be achieved by application of a (fairly) domain-independent conflict-resolution engine to a fairly domain-independent data structure." Their expectation is what we propose: a domain-independent conflict-resolution engine of which the actions are explicitly specified by means of a conflict-rule file. To resolve interpretation conflicts we propose a conflict-resolution strategy that takes into account the following questions:

Detection : At what map-object hierarchy level does the conflict arise?

Verification : At what map-object hierarchy level should it be resolved?

Resolution : How should it be resolved?

Conflict detection and verification For the detection and verification of conflicts, we propose that conflicts are described using the above-mentioned conflict rules. Conflict rules specify where, i.e. at what object-hierarchy level, a conflict can appear (detection) and at what level the conflict must be resolved (verification). When we delay the resolution of an interpretation conflict, the interpretation process has the opportunity to reason with conflicting instantiations (alternative interpretations)
until it has gathered more information (context) to resolve the conflict. The gathering of information is done by forming more complex objects and, as such, capturing more of the structure of the map. It is generally accepted that context (structure) is as important for the interpretation process as an information source, as the recognition of simple objects is [52, 106]. To choose a verification level for a conflict is a task for the knowledge engineer. When conflicts are resolved at the same level as where they appeared, no contextual information is incorporated in the resolution scheme. On the other hand, when we resolve a conflict at a high level in the map-object hierarchy, it may give rise to a combinatorial explosion of possibilities. Clearly, there is a trade-off in between the two extremes and the specification of conflict rules provides us a means to control conflict resolution.

**Conflict resolution** For conflict resolution, we propose the use of a pattern classifier. Usually, in (image) interpretation, conflicts are resolved by comparing the two instantiations of concepts which are in conflict, and choosing the instantiation which has a higher judgment value than the other. If we have two different instantiations of the same concept this causes no problems, because the judgment values of both instantiations are calculated in the same manner. However, if we have a conflict between an instantiation of a concept, denoted as $A$, and an instantiation of a distinct concept, denoted as $B$, this causes a potential problem because the way in which the two judgment values are calculated can be entirely different. The object composition of complex object $A$ can have more or less hierarchy levels than the object composition of complex object $B$. Furthermore, object $A$ can have different part and specialization links, different standard or special geometrical relationships, and different simple objects than object $B$. Different part and specialization links, geometrical relationships, and simple objects have their different judgment functions and detectors and therefore have different confidences and accuracies. In other words, the calculation of a judgment value for a certain complex object may be biased optimistically or pessimistically and may have a variance. If we apply a pattern classifier to this conflict-resolution task then we are not only able to linearly discriminate between the two judgments, as with common uncertainty calculi, but we can also possibly compensate for any apparent biases (optimistic, pessimistic) and variances of the judgment values. Joseph et al. state in [51] that: "Conflict resolution must be done on the basis of heuristic knowledge of the ways in which drawing entities might combine to form complete drawings." We have the opinion that this heuristic knowledge can be captured most easily by learning conflict classifiers from a set of manually labeled examples of (conflicting) complex objects. Figure 5.23 depicts the paradigm we follow for conflict classification, it is shown for the special case of two conflicting instantiations, labeled as $A$ and $B$, of the concept dimension as given in Figures 5.2 and 5.3. The complete scheme for a conflict classification forms a cascade of classifiers. At the first level of the cascade, we have detectors for the simple objects arrow, conduit, and house side, a classifier as a judgment function for the perpendicular relationship, and the standard judgment function for the in-line relationship. For simplicity, we assume that the arrow concept has a detector instead of specializations double- and single-headed arrow. The outcome of the detectors and the judgment functions for the geometrical relationships are mapped to judgment values in the interval $[0, 1]$. These five judgment
values form the input vector for the classifier of the complex dimension object at the second level of the cascade. By classification and consecutive judgment-value mapping we obtain a judgment value for both instantiations of a dimension. The two judgment values form on their terms the input for the third level of classification in the cascade: conflict classification. The conflict classifier determines which of the two conflicting instantiations should be kept.

5.6.2 Conflict-resolution mechanism

Our conflict-resolution mechanism is an integral part of our interpretation mechanism, as depicted in Figure 5.9. Before interpretation starts, the conflict-resolution engine is loaded with the conflict-rule database specified in the conflict-rule file. Because the database is considerably large in our application, we use a hashing table as a data structure for the conflict-rule database. Hashing is performed with objects that can be in conflict and objects forming the cause of conflicts as key attributes.

The conflict-resolution engine comes into effect after the verification step in the interpretation cycle. The conflict-rule base is used to check whether the instantiations of LTM concepts might get involved in interpretation conflicts; the (instantiations of) LTM concepts are stored in the list of instantiations of the working
5.6 Interpretation conflicts

<table>
<thead>
<tr>
<th>Complex map object</th>
<th>Linear</th>
<th>Quadratic</th>
<th>k-NN</th>
<th>Parsen</th>
<th>Max</th>
<th>Min</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>dimension house ⇐ conduit</td>
<td>73</td>
<td>81</td>
<td>61</td>
<td>79</td>
<td>51</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>dimension conduit ⇐ arrow</td>
<td>83</td>
<td>82</td>
<td>82</td>
<td>82</td>
<td>28</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>dimension conduit ⇐ dashed line</td>
<td>37</td>
<td>72</td>
<td>37</td>
<td>37</td>
<td>63</td>
<td>17</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 5.11: Percentage of correctly classified conflicts for different classifiers.

<table>
<thead>
<tr>
<th>Complex map object</th>
<th>Linear</th>
<th>Quadratic</th>
<th>k-NN</th>
<th>Parsen</th>
<th>Max</th>
<th>Min</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>dimension house ⇐ conduit</td>
<td>4</td>
<td>24</td>
<td>2</td>
<td>23</td>
<td>45</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>dimension conduit ⇐ arrow</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>48</td>
<td>10</td>
<td>26</td>
</tr>
<tr>
<td>dimension conduit ⇐ dashed line</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 5.12: Number of false positive classified conflicts for different classifiers.

hypothesis. If so, and interpretation takes place at the specified hierarchy level of conflict resolution, which is set by the working hypothesis, actual resolution takes place. If interpretation does not take place at the right level of resolution, no action is taken and actual resolution is postponed until the right level has been reached. Resolution of conflicts at an interpretation level is performed by checking the instantiations of the working hypothesis. If any conflicting instantiations are found, they are stored in a conflict-resolution set. From the resolution set two conflicting instantiations are taken. The two instantiations are judged and one instantiation is marked for deletion by the conflict classifier. The other instantiation is put back into the resolution set. The procedure ends when the resolution set is consistent, i.e. contains no more conflicting instantiations. The instantiations marked for deletion are deleted from the list.

5.6.3 Conflict-resolution experiments

In Sections 5.4.1, 5.4.2, 5.5.1 and 5.5.2 we gave the results of experiments with single detectors, multiple detectors, judgment of geometrical relationships and judgment of complex objects. These experiments concerned the detection and judgment of objects and relationships. In this section, we present the results of experiments with conflict resolution. Conflict resolution is applied to the complex objects used in the experiments on judging complex objects (Section 5.5.2). It is shown that conflict resolution can further reduce the number of falsely positive judged complex objects. In order to do so, the conflicts between the different complex objects are identified and conflict classifiers (see Figure 5.23) are learned from examples. We use the classifiers for judging complex objects as conflict classifiers. Table 5.11 gives the conflict classification results for conflicts of some complex objects; the number of false positive classified conflicts is given in Table 5.12. Surprisingly, application of maximum resolution, i.e. choosing the instantiation with the highest judgment value, does not give the best performance. The results show that the use of a linear or quadratic classifier is to be preferred. Table 5.13 gives the classification results for complex objects after application of conflict resolution with the best conflict classifier. When we compare Table 5.13 with Tables 5.9 and 5.10, we notice that the number of false positive complex objects can be reduced if we accept some decrease in performance.
<table>
<thead>
<tr>
<th>Complex map object</th>
<th>Correct</th>
<th># false positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>dimension house ↔ conduit</td>
<td>47</td>
<td>3</td>
</tr>
<tr>
<td>dimension conduit ↔ arrow</td>
<td>89</td>
<td>7</td>
</tr>
<tr>
<td>dimension conduit ↔ dashed line</td>
<td>67</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 5.13: Percentage of correctly classified complex objects after conflict resolution and number of false positives.

5.7 Discussion: current limitations and drawbacks

In this section we discuss our interpretation system and pinpoint its current limitations and drawbacks.

Knowledge specification In our application, using a semantic network to represent knowledge seems the best choice. The functionality and expressiveness given by part and specialization links is adequate to represent the document structure. One missing functionality, however, is the Kleene star operator on part links. In our semantic-network approach we have to specify beforehand exactly how many parts a complex object can have. There are cases in which we do not know how many parts can occur. For instance, how many dashes does a dashed line have? We have made a first implementation of the Kleene star operator on part links, but it is not used in the experiments for this chapter. Some of the necessary Kleene star operators on part links have been moved into an image detector (dashed-line detector).

The use of explicitly specified knowledge has its advantages and drawbacks. One major advantage is that the bulk of tuning parameters is editable in a specification file. This reduces the number of test-rewrite-compile cycles of a system. Another advantage is that when application-domain knowledge is explicitly specified, the system is, in principle, applicable in other areas without requiring a complete redesign of the system. In our framework, for example, it is sufficient to write new detectors, design a new object hierarchy by means of part and specialization links, specify geometrical relationships and possible interpretation conflicts. This specification phase is then followed by a system-learning phase in which judgment functions for concepts and relationships and conflict classifiers are learned from labeled examples. One drawback of explicit knowledge specification is that the syntax and semantics of the specification are usually not as well defined as, for instance, the C programming language is. As a result, the idea that application-domain knowledge is easily editable by ordinary users may prove to be false.

Abstraction level of detectors Difficulties also lie with the choice of abstraction level of image-analysis algorithms: which objects are chosen to be the simple objects which have a detector? For instance, do we choose a dash as a simple object or do we make a dashed-line detector? In our application, some of these choices are made on the basis of heuristics.
Detectors versus context  Given the fact that we have to combine two information sources in our interpretation system, one originating from the detectors and one coming from the context, we must choose which information source dominates the other. Do we choose a system design with strong detectors (having a low number of false positives) and weak context, or a design with weak detectors (having a higher number of false positives and higher classification rate) and a strong context that can possibly delete the false positives by means of conflict resolution? This choice is application dependent. In unambiguously structured drawings (or documents), which contain a small set of easily detectable objects, one can let the context prevail. On the other hand, in less structured drawings with a rich set of graphical entities that can easily be confused, one can opt for strong detectors and a weak context. In our utility-map conversion system, the false positive detection of a simple object or the false positive interpretation of a complex object has to be deleted and redrawn by the operator. A false negative detection of a map object is considered to be less costly in man-machine interactions as it only involves redrawing the missed map object. In order to reduce the number of false positive detections, we have chosen a system with strong detectors and a conflict-resolution scheme that detects and solves any remaining false positive detection.

Conflict specification  One drawback of our interpretation system is the amount of knowledge specification it needs. Besides a description of the domain model in terms of simple and complex objects, geometrical relationships, part and specialization links, and a description of control knowledge by means of assigning object priorities, an additional specification of all possible interpretation conflicts is necessary. If the number of specified objects grows, the number of possible interpretation conflicts grows as well. Although a fair portion of the specification process can be automated (if we use drawing rules and conventions), some of the conflicts still have to be specified and adjusted manually. Furthermore, the conflict specification is limited to the case in which precisely two complex objects are in conflict over one or two instantiations. Complex situations may exist where more than two complex objects are in conflict over more than two instantiations. In that case, we need an extended specification, possibly a logical specification with an arbitrary number of conjunctions or disjunctions of conflicts and a different conflict-resolution engine which takes into account the number of conflicting complex objects.

Learning phase  Apart from the job of knowledge specification mentioned above, another time-consuming job is the training of all classifiers in the learning phase of the system. As we use classifiers for the detection of the simple objects, judgment of geometrical relationships, judgment of complex objects and conflict classification, we have a large amount of classifiers to be trained. Training of classifiers requires the manual labeling of examples as right or wrong. Considering the amount of classifiers to be trained in our framework, the job of manually labeling examples seems grotesque. But fortunately, the job of labeling examples can be restrained to labeling simple and complex objects. The labeling of simple objects can be done by assigning each example to a simple object class. By means of these examples, we train the (multiple) detectors. Using these detectors, the interpretation can generate some instantiations of complex objects. By labeling the instantiations of complex
objects as right or wrong with a differentiation of the cause, we can automatically generate labeled examples of simple objects, other complex objects, geometrical relationships, and conflicts. This implies that if an example of a complex object is labeled as wrong, we additionally input:

- which simple object is wrong
- which complex object, part, or specialization of this complex object is wrong
- which geometrical relationship is wrong

Furthermore, we need a tool which visualizes the examples of complex objects in such a way that we can differentiate between the different causes. Although the learning phase of our interpretation system takes a considerable part in the overall system design process, we state that the learning phase of an interpretation system must not be underestimated. Any interpretation system without an adequate learning phase cannot be taken seriously, let alone an interpretation with no learning phase.

5.8 Conclusions and summary

In this chapter we described the interpretation part of a conversion system for public-utility maps. The interpretation is fitted into a general framework for (image) interpretation, which was described in detail. We extended the common semantic-network approach with the object-oriented programming concept of (image) detectors. We carried out experiments on combining different detectors for the detection of simple map objects and showed that the use of multiple detectors can reduce the number of false positive detections. We gave a representation of geometrical relationships and carried out experiments on classifying some common geometrical relationships. The experiments showed satisfying results. A new uncertainty calculus was proposed for the judgment of complex objects, based upon the use of pattern classifiers as uncertainty propagators. Their use was illustrated with experiments on judging complex map objects. Concerning the interpretation mechanism, we argued for the use of a priority queue as a data structure to store and retrieve hypotheses of objects. A queue enables a correct interpretation order in a mixed bottom-up and top-down approach. Further, we have discussed interpretation conflicts and gave arguments for the necessity of explicitly specified context-dependent conflict rules and the appliance of a pattern classifier as a conflict-resolution means. Experiments showed that maximum resolution is not optimal. The general paradigm we follow is to treat each combining function of judgments and each comparison function between judgments as a classifying problem, enabling to learn the utility-map domain in terms of objects, geometrical relationships and conflicts from examples.
Chapter 6

Utility-map reconstruction

In this chapter we describe a new system for the reconstruction of geometrical map objects as developed in the Dutch TopSpin-PNEM project "Knowledge-based conversion of public-utility maps". Vectorized map objects, which come from an image-interpretation system, are reconstructed and reference objects are aligned with respect to a base map in such a way that geometrical inference is made possible in a geometrically correct topographic base map. This is done using a map correspondence between the public-utility map and the base map that is based on mutual topography and relational information supplied by a map-interpretation module which applies a novel conflict-resolution strategy. With the map correspondence, the relationships, and several object-specific functions, the map objects are reconstructed relationally and geometrically correct.

This chapter is based on the following publication:
6.1 Introduction

In the industrialized world there is currently a large interest in the conversion of engineering drawings to database files for CAD systems and the conversion of utility maps to digital maps which are suitable for geographical information systems (GIS). To avoid tedious and expensive conversions by hand, research is done on automatic conversion of engineering drawings and utility maps. We collaborate in a research project that investigates the possibility of automatic conversion of Dutch public-utility maps. The main objective of this project is to achieve a gain in speed of the entire conversion process. As fully automatic conversion is not feasible yet, a semi-automatic conversion-process is proposed, in which the operator corrects and appends the results of the automatic image-interpretation process. For the system layout of the complete conversion process we refer to [26,105]. A simplified overview of the proposed conversion system is shown in Figure 6.1.

Figure 6.1: Conversion-system overview.

This chapter focuses on the automatic map reconstruction of this system. The goal of the reconstruction is to correctly position the map objects of the utility map in the coordinate system of the base map because the utility map suffers from local scale, translation, and rotation distortions. The reconstruction takes four types of input:

- A utility map consisting of vectorized map objects.
- A base map consisting solely of vectorized houses.
- A set of point pairs that forms a map correspondence between the houses in the utility map and the houses in the base map.
- Geometrical relationships between map objects in the utility map.

The algorithm to find a map correspondence is introduced in Section 6.2, whereas the interpretation module, which produces geometrical relationships between map objects, is discussed in Section 6.3. The object-specific reconstruction is discussed in Section 6.4. Results of experiments with the reconstruction are given in Section 6.5.
6.2 Map correspondence

Before we can reconstruct map objects, we have to establish a correspondence between the utility map and the base map. The correspondence is found with an algorithm that matches points in the utility map and the base map by comparing polygonal objects present in both maps. The only objects present in both maps are houses. The matching starts by globally transforming the map objects in the utility map using an affine transformation that is determined by manually identified control points. This step is introduced to reduce the number of possible matching candidates in the next step. After this global transformation the following differences between the houses in both maps can be observed:

- Different number of houses in both maps: missing or additional houses in one or both of the maps.
- Differences in local scale, translation, and orientation.
- Differences in the number of vertices of corresponding polygons due to a difference in abstraction, e.g. detailed description of house sides versus square representation of a house, or a difference in completeness, see Figure 6.2.

Figure 6.2: Two examples of a difference between houses on a base map and a globally transformed utility map: (a) completeness difference, (b) abstraction difference.
To deal with these differences we use an inexact matching procedure on the basis of the vertices of the house polygons as features to estimate the local scale, translation, and orientation. The matching procedure, described in more detail in [21], consists of the following steps:

**Search candidates**: Starting with an object in one of the maps, candidate matching objects are selected within a spatial region of interest in the other map.

**Permutate**: For every candidate match, we generate all permutations of vertices of both polygons which are compatible with each other. In this way, we can handle incomplete polygons and spurious details in polygons. The compatibility measure is based on two measurements: the distance between vertices and the orientation difference of the corners of vertices.

**Apply evaluation of likelihood function**: The generated permutations of vertices are evaluated with a likelihood function based upon the least-squares residual error of a local affine transformation. The candidate match with the maximum likelihood is selected as a match and establishes a *local* correspondence between the utility map and the base map.

### 6.3 Interpretation

The interpretation module takes labeled map objects as input and produces their geometrical relationships, see Figure 6.3. The input map objects are *conduit pieces*, *connections*, *house sides*, *arrows*, *dashed lines*, and *roadside pieces*. All map objects are represented as a labeled vector \( [x_1, y_1, x_2, y_2, \mathcal{L}] \), where \((x_1, y_1) \in \mathcal{Z} \times \mathcal{Z}\) represents the starting point, \((x_2, y_2) \in \mathcal{Z} \times \mathcal{Z}\) the end point, and \(\mathcal{L} \in \mathcal{N}\) the label of the object, except for connections which are represented as a labeled point \([x_1, y_1, \mathcal{L}]\) with \((x_1, y_1) \in \mathcal{Z} \times \mathcal{Z}\) and \(\mathcal{L} \in \mathcal{N}\) the label. There is no uncertainty concerning the labels of the map objects; only their spatial position is unknown. The desired output geometrical relationships between these map objects are: in-line, perpendicular, T-junction, and with-distance-of. To find these geometrical relationships, the interpretation module groups the input map objects into *complex* map objects, like *dimensions* and groups of *dimensions*. A basic *dimension* typically

---

**Figure 6.3**: The interpretation module takes labeled map objects as input and produces their geometrical relationships. The symbol || stands for an in-line relationship, > stands for a perpendicular relationship, and \(\perp\) for a T-junction.
consists of an arrow specifying a distance between two map objects. The arrow has geometrical relationships with the two map objects. For instance, an arrow is always drawn in line of a house side. These geometrical relationships stem from the rules by which the technical draftsmen draw utility maps. Figure 6.4 gives two examples of the model specification of a dimension. The interpretation process

![Diagram](image)

**Figure 6.4**: Two example specifications of dimensions: (a) dimension that measures the distance between a house side and the conduit, (b) dimension that measures the distance between a roadside and a connection.

finds the required geometrical relationships for the reconstruction by automatically hypothesizing and verifying the dimensions. The knowledge about dimensions and their geometrical relationships in our utility-map domain is represented in a semantic network [78]. Our utility-map interpretation module differs from (utility-map) interpretation systems described in the literature [23,27,35,49,52,55,97] and in Chapter 5 in the sense that the input of the interpretation is not an image, but a set
of correctly classified vectors and points. Although its input differs, the inner workings of the interpretation remain the same as those described in Chapter 5. Besides the type of input there are some other differences between the two interpretation systems, which we discuss in the following paragraphs.

6.3.1 Knowledge representation

The main difference between the knowledge representation from Chapter 5 and the representation used here is that the concept of image detectors has been abandoned for this application. This stems from the fact that the input of the interpretation is not an image but a set of correctly classified vectors and points, which makes detection superfluous.

Simple and complex concepts  When there are no detectors, the distinction between simple and complex concepts is based upon the fact that a simple concept has no parts or specializations while a complex concept does have parts or specializations. All simple concepts have a corresponding input vector or point with an associated label. As a result, there are simple LTM concepts defined for conduit pieces, connections, house sides, arrows, dashed lines, and roadside pieces in our application domain. Complex concepts in our application domain define dimensions and groups of dimensions. All simple concepts in LTM have a vector or point attribute. Because the input is already (correctly) classified, there is no need for additional attributes.

Geometrical relationships  The specification of the application-domain knowledge uses only standard geometrical relationships and their corresponding standard judgment functions. As mentioned in Section 5.3.1, the standard geometrical relationships include in-line, perpendicular, parallel, T-junction, and distance, which cover all required geometrical relationships. The standard judgment functions used within our application are slightly modified: the judgment value of a geometrical relationship solely depends on the derivation of the relationship parameter $d$, i.e. on the distance between the two arguments of the relationship. As such, the judgment function of a geometrical relationship equals the following conditional probability:

$$P(\text{relationship} \mid \Delta d) = L_1(\Delta d, \mu_d, \sigma_d) = L_1(d, 0, \sigma_d)$$  \hspace{1cm} (6.1)

We use a $L_1$-norm as function $L_1(d, 0, \sigma_d)$, which is parametrized by $\sigma_d$. Section 6.3.4 discusses the motivation behind omitting the derivation of the angle parameter $\alpha$ in the calculation of the judgment value of the relationship.

6.3.2 Interpretation mechanism

The loss of detectors implies that the interpretation process must be initiated in a different way. The process is initiated by converting the input data into corresponding instantiations of simple LTM concepts, which consequently are stored in STM. Because there is no uncertainty in the label of these instantiations, the interpretation process can continue strictly top-down without resulting in an enormous number of possible interpretations. After the input data is converted, the
interpretation hypothesizes all dimensions and dimension groups. The generated hypotheses are processed in a top-down fashion, resulting in the instantiations of dimensions and dimension groups. The geometrical relationships between arrows and other map objects can then be derived from these resulting instantiations.

6.3.3 Judgment functions

The judgment values of the instantiations of the simple LTM concepts are all set to one: they are certain without doubt. As a result, the judgment values of complex concepts solely depend on the judgment values of the geometrical relationships of the complex concept. For the judgment function of a complex object we take the minimum operator. Because instantiations of simple objects all have maximum judgment values, the judgment of a complex object is thus the minimum of the judgment values of the geometrical relationships. As mentioned earlier, the judgment value of a relationship solely depends on the distance between the two arguments of the relationship. Consequently, the judgment value of a complex object expresses the maximum distance between two of its parts, which are geometrically related.

6.3.4 Interpretation conflicts

In our proposed conversion system, the system operator has some flexibility to adjust and append the image-interpretation results. The operator has the possibility to position a missed map object with a certain tolerance in its position and in its geometrical relationships with other map objects. Due to this flexibility, interpretation conflicts may arise during interpretation. All possible interpretation conflicts between dimensions and dimension groups are specified in a conflict-rule file, which is consulted by the conflict-resolution scheme during interpretation. The conflict-resolution scheme is a simplified version of the scheme introduced in the previous chapter, i.e. the conflict resolutions are not learned from examples but are based upon a selection of the instantiation which has maximum judgment value. Because the judgment value of a complex object depends on the judgment value of geometrical relationships, the conflict-resolution scheme is based upon how well geometrical relationships are fulfilled.

6.4 Object-specific reconstruction

The results of the map correspondence and the interpretation serve as input for the object-specific reconstruction. With the corresponding point pairs we are able to determine a reliable local transformation of the utility map in the coordinate system of the geometrically correct base map. Furthermore, with the geometrical relationships, it is possible to form groups of interrelated objects. In the reconstruction, two main types of objects are distinguished: dimension groups and conduits. Dimensions specify a direction and/or a distance. Dimension groups are two or more geometrically related dimensions that are geometrically related to the topography, i.e. houses, and determine the location of points of the conduit. The reconstruction is based on the dimension groups according to which the conduits are aligned. It is important to note that the geometrical accuracy of objects in the utility map is
relatively low. Consequently, a visually appealing reconstruction of dimensions and conduits has a higher priority than the geometrical accuracy, which eventually may lead to local scale differences.

6.4.1 Dimension groups

The goal of the reconstruction of dimension groups is the placement of the individual dimensions in the coordinate system of the base map, while preserving the geometrical relationships found in the utility map. The procedure we have developed to achieve this goal uses a sequential reconstruction approach. Correction of a group of dimensions only takes place when the points of that group are in correspondence with the points of the dimension group in the base map. By starting from these points, it is possible to place the dimensions in the base map, using the actual information about the directions and the distance of the dimension. The order in which the dimensions in the group are placed is determined based upon the type of dimension as well as the type of its geometrical relationship with previously placed dimensions in this group and their associated degree of freedom. From this ordered set, first the dimension with the lowest degree of freedom is placed and afterwards removed from the ordered set. Based on the new set, new dimensions may be added to the set and the steps are repeated until all dimensions have been reconstructed. When the dimensions in a group form a loop, there often is a residual coordinate difference in the computation of the location of the dimensions. This residual difference can only be resolved if the loop contains a dimension that only specifies a direction or an intersection relationship, as it are only those objects which can be adjusted in length or coordinate position.

6.4.2 Conduits

In the reconstruction of the conduits, the main condition for success is the preservation of the shape of the conduit as it is found in the utility map. The shape of the reconstructed conduit is determined by the angles and distances between geometrically related dimensions and connecting conduits. Obviously, a conduit is reconstructed on the basis of the points of its corresponding base-map dimensions. Local scale differences that are associated with the dimensions lead to the conclusion that one cannot simply use a similarity transformation for the reconstruction of the conduits. Therefore, it is necessary to formulate a model which can preserve the shape of the conduit while bringing the points into correspondence. Thereto a least-squares parameter-estimation problem is chosen such that the final position of the points of the conduit are the unknowns. The input parameters of the estimation problem are the shape-preserving quantities describing the conduit. In order to deal with local scale difference, only the angles between the points of the conduit(s) are considered as input parameters as well as a local-scale factor. In the parameter estimation we assign weights to different types of angles. We distinguish the following angles, in descending order of the assigned weight: from or to a geometrically related dimension point, from or to a connecting conduit point and from or to another point. In this way, the angles at the most important parts of the conduit are preserved. A complete description of this procedure can be found in [19].
6.5 Experiments

Our system for the reconstruction of geometrical map objects has been tested on three utility maps with varying complexity. Utility maps are typically A0-sized, scale 1 : 1000, and have approximately 500 dimensions. The error measures determined are the percentage of correctly reconstructed dimension groups and correctly positioned conduit points. Section 6.5.1 discusses the results for dimension groups and Section 6.5.2 elaborates on the results for conduits.

### 6.5.1 Dimensions

In Table 6.1 the reconstruction results are given for dimension groups. The reconstruction results have been visually inspected and each dimension or dimension group has been labeled as correct or incorrect depending on whether the geometrical relationships are similar to the ones in the original utility map. To determine the influence of the number of dimensions in one group on its reconstruction result, we have divided the reconstruction results of dimension-group reconstruction into two parts: results for groups with less than four dimensions and results for groups with more than three dimensions. One can see that if there are less than four dimensions in a group, reconstruction goes well in 80% of the cases, as opposed to the reconstruction of groups with more than three dimensions, which goes well in 55% of the cases. To evaluate this huge difference we look at specific causes:

Object-specific reconstruction error: Errors made by object-specific reconstruction. This type of error can occur when loops in dimension groups become too complex for the reconstruction algorithm to handle. As such, these errors mainly occur for large groups (≥ 4 dimensions).

<table>
<thead>
<tr>
<th>Utility map</th>
<th>Number of dimensions in one group</th>
<th>R1272w</th>
<th>R1272z</th>
<th>R1318a</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt; 4</td>
<td>≥ 4</td>
<td>&lt; 4</td>
<td>≥ 4</td>
</tr>
<tr>
<td>correctly reconstructed</td>
<td>88%</td>
<td>67%</td>
<td>85%</td>
<td>52%</td>
</tr>
<tr>
<td>correctly reconstructed (unbiased)</td>
<td>90%</td>
<td>71%</td>
<td>90%</td>
<td>60%</td>
</tr>
<tr>
<td>correspondence error</td>
<td>1%</td>
<td>0%</td>
<td>1%</td>
<td>4%</td>
</tr>
<tr>
<td>interpretation error</td>
<td>6%</td>
<td>17%</td>
<td>7%</td>
<td>13%</td>
</tr>
<tr>
<td>object-specific reconstruction error</td>
<td>3%</td>
<td>10%</td>
<td>1%</td>
<td>18%</td>
</tr>
<tr>
<td>inherent error</td>
<td>2%</td>
<td>6%</td>
<td>6%</td>
<td>13%</td>
</tr>
</tbody>
</table>

Table 6.1: Reconstruction results for three utility maps (R1272w, R1272z, R1318a). The percentage of correctly reconstructed dimension-groups and the percentage of the cause of errors are depicted. The unbiased percentage of correctly reconstructed dimension groups does not take into account inherent errors because they are not errors of the reconstruction system. All results are given for each utility map for dimension groups smaller than four dimensions (first column) and dimension groups greater than or equal to four dimensions (second column).
<table>
<thead>
<tr>
<th>Utility map</th>
<th>R1272w</th>
<th>R1272z</th>
<th>R1318a</th>
</tr>
</thead>
<tbody>
<tr>
<td>relative performance</td>
<td>88%</td>
<td>94%</td>
<td>95%</td>
</tr>
<tr>
<td>absolute performance</td>
<td>88%</td>
<td>65%</td>
<td>60%</td>
</tr>
</tbody>
</table>

Table 6.2: Reconstruction results for three utility maps. Depicted are the relative and absolute percentage of correctly positioned conduit points.

**Interpretation error**: Falsely accepted (false positive) and missed (false negative) geometrical relationships in the interpretation. Because large groups incorporate more geometrical relationships, the probability of error is higher than with small groups (< 4 dimensions).

**Correspondence error**: Incorrect matching results. The map-correspondence process can generate wrong corresponding point pairs. As a result, the local transformation will not be valid, which in turn results in incorrectly reconstructed dimension groups.

**Inherent error**: Inherent impossible reconstructions. It is possible that the map objects form a semantically impossible situation to (re)construct, e.g. drawing errors made by the technical draftsman or changed topographic situations in the base map.

Because inherent errors are not errors made by the reconstruction system, Table 6.1 also presents unbiased reconstruction results when these errors are not taken into account. Then reconstruction goes well in 90% and 70% of the cases. The three utility maps are of different complexity. This is clearly visible in the percentage of correctly reconstructed dimension groups, the correctly positioned conduit points in Section 6.5.2, and the inherently impossible reconstructions. From the results we may conclude that the performance for the map correspondence is satisfying. The enormous number of inherently impossible reconstructions for the map R1318a with high complexity is troublesome, as these cannot be resolved, neither automatically, nor by the operator. Furthermore, it is clear that the size of a dimension group increases the probability of an error made in its reconstruction.

### 6.5.2 Conduits

In Table 6.2, the results are depicted for the reconstruction of conduits in the case of correctly reconstructed dimension groups (relative performance) and the percentage of correctly positioned conduit points overall, i.e. w.r.t. all conduit points (absolute performance). The absolute performance is an indication of how well the conduit shapes are preserved and how much work is left for the operator after the reconstruction process. From the results, we may conclude that the reconstruction of the conduit depends heavily on the success of the reconstruction of the dimensions.
6.6 Reconstruction example

An example of utility-map reconstruction is illustrated in Figure 6.5, where a small part of a utility map is shown before and after reconstruction. Figure 6.5(a) shows the (scanned) utility map, Figure 6.5(b) shows the corresponding part of the base map which here only has one house, depicted in black on the utility map (gray). Figure 6.5(c) shows the utility map in gray on top of which the interpreted map-objects have been plotted in black. The conduit appears as a thick line in the utility map, whereas dimensions can be recognized by their associated dimension number, with the exception of dimension leaders which are depicted as dashed lines. Figure 6.5(d) shows the reconstruction result. As stated above, the goal of map reconstruction is to calculate the new locations of the map objects in the coordinate system of the base map. The output of the reconstruction process in Figure 6.5(d) shows the reconstructed map objects. All map objects are now represented in the coordinate system of the base map. In the middle of Figure 6.5(d), a shortcoming of the map reconstruction is visible. At that position, a dimension leader connects to the conduit. In our system, dimension leaders are rigid, implying that the length of a dimension leader cannot be changed in the reconstruction process. In this case, the end point of the dimension leader is a fixed control point, which causes the shape of the conduit to be altered w.r.t. the shape before reconstruction. Because dimension leaders only specify a direction and not a distance, it should be possible to alter the length of dimension leaders (up to a certain amount) in this case in order to preserve the shape of the conduit. We are currently implementing non-rigid dimension leaders in our map-reconstruction system.

6.7 Conclusions and summary

In this chapter we presented a new system for the reconstruction of geometrical map objects. The reconstruction module used a local transformation as determined by a map-correspondence module and relational information from an interpretation module. The map-correspondence module could handle differences in local scale, translation, rotation, abstraction, completeness, and number of houses by applying an inexact-matching procedure. The interpretation module could handle context-dependent interpretation conflicts by using an explicitly specified conflict-resolution strategy. The object-specific reconstruction focused on dimensions and conduits. The proposed reconstruction procedure preserved the shape of the objects, despite local scale differences. Experiments showed satisfying results.
Figure 6.5: Reconstruction example. (a) Scanned utility map. (b) Utility map (gray) and base map (black). (c) Utility map (gray) and interpretation of utility map (black). (d) Reconstructed utility map (black).
Chapter 7

Music-score interpretation

This chapter describes a system for music-score interpretation. This system serves as an example application of our general framework for document-image interpretation. This framework was proposed in Chapter 5 for the specific case of interpreting public-utility maps. It is shown that by tailoring the general framework, the design of a system for music-score interpretation can be shifted from a complete system design from scratch to a more abstract level of design. At this abstract level of design, the focus is on writing image detectors for simple graphical objects, on specifying music-score structure and on tuning interpretation knowledge to ensure an efficient interpretation order. We shortly describe our interpretation system and show the design steps taken to come towards a music-score interpretation system.
7.1 Introduction

A machine that reads and plays music has fed the imagination of many. Although the actual usefulness of such a machine seems less important than seeing the machine playing, the development of a optical music-interpretation system is one of the scientific challenges in computer era. Even though, at this time, there are image-interpretation systems that can handle far more difficult input images, the complexity of optical music interpretation is far better comprehended (and appreciated) by the general public. Although the scientific challenge legitimates the research on optical music interpretation in its own right, there are quite a number of useful applications. Among these are applications with editorial, archival, and scientific purposes:

Archiving music: An unmeasurable amount of music scores is stored as paper documents in libraries and private archives.

Adapting music to other instrumentations: Music scores arranged for certain instruments can be arranged for different instruments. For example, a full orchestra score might be rewritten such that it can be played on a single piano.

Transposing music: Transposing of music involves writing it in a different key; shifting the symbols of an already existing score and adding or removing accidentals.

Using automatic harmonization: In automatic harmonization, chords are added to a given monophonic melody.

Renewing music: Renewal involves repagination and reprinting of old scores.

Synthesizing compositions: The computer can also aid with the synthesis of existing compositions on piano by means of attached mechanical devices. Using the piano characteristics and an analysis of the piano piece to be played, it can set the parameters of the mechanical device to correctly perform the dynamics of the piano piece.

Analyzing music: A large digital database with music scores aids doing scientific research on music theory. For example, to discover relationships in structure or style between compositions of some historical time period, one can perform data-mining on a database with all the works of that period.

Related work on music-score interpretation has been reported by d'Adency et al. [16], Andronic et al. [6], Biostein et al. [8], Carter et al. [11, 12], Coûasnon et al. [13-15], Fujinaga et al. [36, 37], Kato et al. [56], Martin et al. [72], Pennycook [83], Prerau [85, 86], Roach et al. [91] and Stückelberg et al. [108]. The majority of the approaches are low level and start with the recognition of specific music symbols, like staff lines, to bootstrap the recognition of other symbols. A more high level and less rigid approach is proposed by Coûasnon et al. They state that there is much knowledge on music-score structure in the form of rules for musical notation, and they propose a music-recognition system which uses a grammar to represent this knowledge. The grammar guides the segmentation and recognition of
the graphical objects. A drawback, however, is that the grammar is implemented in \texttt{λProlog}, which is not appropriate to implement low-level algorithms like pattern recognition or object classification. As a result, these algorithms are written in a different programming language and are applied before the grammar parser, making the approach strictly bottom-up. Furthermore, no methodology is presented to deal with interpretation conflicts.

In this chapter, a music-score interpretation system is presented that is based on the general framework for document interpretation proposed in Chapter 5. It combines a bottom-up and top-down interpretation strategy, and deals with interpretation conflicts by means of explicitly specified conflict rules. Section 7.2 gives a short description of our general framework for document interpretation. Section 7.3 describes our music-score interpretation system. Some examples of detectors for simple music objects and specifications of complex music objects are given.

### 7.2 Interpretation system

This section describes our general framework for document-image interpretation. A more detailed description of this framework is given in Chapter 5. The input of our interpretation system is the gray-scale image obtained by scanning the paper document. The input image is binarized by sharpening, thresholding and post-processing, as discussed in Chapters 2, 3, and 4. The interpretation process starts by detecting simple objects in the image data with specialized detectors. These detectors are built from morphological binary or gray-scale image operators (opening and closing), template matching, pattern classifiers on feature vectors, or a combination of the above. The interpretation process continues by grouping these simple objects into complex objects by exploiting their geometrical relationships. The knowledge about the application domain governing the interpretation is represented explicitly in a semantic network by means of knowledge files. Within this semantic network, specific simple and complex objects, part and specialization links, and geometrical relationships are represented. Additionally, possible interpretation conflicts are described by specifying the context in which a conflict may appear as well as when and how it should be resolved.

### 7.3 Interpretation of music scores

This section shows the design steps taken to arrive at a music-score interpretation system by using the above-described interpretation system. Section 7.3.1 describes the specialised detectors for the various simple music objects, followed in Section 7.3.2 with a specification of the complex music symbols. To facilitate understanding of the discussion, we give in Figure 7.1 the music score symbols the final interpretation system can recognize. Figure 7.2 shows an example input image and desired interpretation results.
7.3.1 Detectors for simple music objects

This section describes some specialized detectors for various simple music objects. We have chosen these simple objects for a detector because they are relatively easy to detect.

**Staff-line detector** A staff line defines a time-pitch coordinate system and is a long, thin line running horizontally across most of the page of the music score. Along the staff line, the time axis (roughly linear) is represented, while perpendicular to the staff lines the dimension of pitch is represented. Higher positions in this direction denote higher pitches. Note that these positions are quantized by five staff lines that together form a stave. The specialized staff-line detector assumes that the document is horizontally aligned and uses a binary morphological opening operator [46, 103] with a horizontal line piece as structuring element. The width of the line piece equals 1 pixel, the length is smaller than the length of a typical staff line. This operator results in a set of horizontal line pieces, and for each of them, an instantiation of the simple staff-line object is created. Upon creation, the vector attribute is calculated.
Figure 7.2: Music-score interpretation. (a) Part of a music score for piano: first bar of the sonata in C-major K545 by Mozart. This part was scanned from the MusicTeX manual [109]. (b) Desired interpretation of music score; denoted are recognized simple and complex objects like clefs, time signatures, notes, note heads, note stems, and note beams. The symbol E(g1) denotes an eighth note g in the first octave as set by the clef, E(g2) denotes an eighth note g in the second octave, etc.
that represents the staff line. Because staff lines are morphologically very distinct
from other simple music objects, the specialized staff-line detector rarely generates
many false positive or false negative detections. Figure 7.3 shows found staff lines
in the music score of Figure 7.2(a).

![Staff lines](image)

(a) ![Detected staff lines](image)

(b)

Figure 7.3: Detection of staff lines. (a) Music score. (b) Detected staff lines.

**Bar-line detector**  A bar line is a short vertical line placed on the stave that separates
notes into bar units. In the case of a music score with multiple simultaneous voices
(as shown in Figure 7.2(a)), this vertical line is longer and connects the staves of the
different voices. The specialized bar-line detector again assumes that the document
is horizontally aligned and uses a binary morphological opening operator with a
vertical line piece as structuring element. The length of the line piece is set in such
a way that it is smaller than the width of a typical stave. The opening results in
a set of vertical line pieces, that all are instantiated as a simple bar-line concept.
In contrast to the staff lines, the bar lines are not morphologically distinct from
other simple music objects, for example the stems of the notes also contain vertical
lines, see also Figure 7.4. Consequently, the bar-line detector may generate many
false-positive or false-negative detections. Selection of true bar lines is dealt with in
the next phase of interpretation that exploits the context of these vertical lines.

**Note-stem detector**  A note stem is the vertical line segment of a note symbol. Apart
from the whole note, all notes have a stem. A note stem has a note head attached
to it at one of its endpoints and possibly one or more note flags at the other,
opposite endpoint. The note-stem detector is similar to the bar-line detector and
uses a binary morphological opening operator with a vertical line piece as structuring
element to detect all vertical line pieces, see Figure 7.4. The length of the line piece
is somewhat smaller than the structuring element for a bar line because note stems
may be smaller than the width of the stave. Each of these pieces are hypothesized
to be an instantiation of the simple note-stem object and the next phase of the
interpretation should distinguish the false positive bar lines from the note stems.

**Clef detectors**  A clef symbol is placed at the beginning (left side) of a stave and aligns
the stave to a particular note. A clef symbol can be a treble, bass, or alto clef symbol.
Figure 7.4: Detection of note stems. (a) Music score. (b) Detected note stems.

Figure 7.5: Detection of solid note heads. (a) Music score. (b) Detected solid note heads.

For all three clef symbols, a separate specialized detector has been developed. The detectors are based on binary template matching, i.e. an exact template of the symbol is created and matched for each pixel of the image. The best matches (according to some threshold on the number of matching pixels) are selected. The templates are acquired by copying and averaging some sample clefs from a number of digitized scores. Because the position of the clef is always at the beginning of a stave, the search area for the clef can be reduced considerably. For each detection, an instantiation of a clef concept is created. The detector does not generate many false positive or false negative detections because clefs are morphologically distinct from other music objects and their search area is kept small.

Note-head detector A note head is the elliptic part of a note symbol. It can be positioned on top of, or at the bottom of the note stem. The vertical position of the note head relative to the stave determines the pitch of the note symbol. In case of whole notes and half notes, the note heads are blank, as can be seen in Figure 7.1(a). For all other notes, the note head is a solid ellipse. The alignment of the ellipse is horizontal in case of whole notes. For all other notes, the ellipse is slightly tilted (five degrees) counter-clockwise, as can be noted from Figure 7.1(e). Two specialized
note-head detectors are developed: one for blank note heads and one for solid note heads. The solid note head detector uses a binary morphological opening operator with a small disk as structuring element. The size of the disk is set in such a way that it fits within the average note head of some sample scores. The resulting objects are all considered instantiations of the simple solid note head object. The specialized blank note head detector actually is the solid note head detector only applied after the holes in the binary image are filled with a closing operator \[46, 103\] with a disk as structuring element. The size of the disk is set in such a way that it is larger than any blank note head of some sample scores. Solid note heads are morphologically rather distinct from other music objects. Combined with a reduced search area for note heads (derived from the note stems) there are not many false positive or false negative detections, see Figure 7.5. The blank notes, however, are morphologically not that distinct and may cause a large number of false positive detections, see also Figure 7.6. For these notes the next phase of the interpretation process must resolve these ambiguities by exploiting the context of the note head.

![Figure 7.6: Detection of whole-note heads. (a) Music score. (b) Detected whole-note heads.](image)

**Note-flag detector** A note flag is a small flag-like symbol which is attached to the note stem at the endpoint opposite to where the note head is attached. A note flag determines (among other symbols) the note duration. Whole-, half-, and quarter-note symbols never have flags. An eighth note has one note-flag symbol, a sixteenth note has two note-flag symbols, and so on. Note that a note flag is attached to one note stem, as opposed to note beams which group and determine eighth, sixteenth, or smaller notes. In our application only eighth-note flags are considered. The detector is based on binary template matching. The procedure of acquiring the templates is the same as for the clef detectors.

**Note-beam detector** A note beam is a long and relatively thick line segment which connects eighth and smaller note symbols with the same time duration. Not all connected note symbols have to share the same time duration; differences in duration are expressed by removing or adding beams to one or more note stems. The line segment can be slightly tilted upwards or downwards, in general depending on
the melody going up or down in pitch. The note-beam detector uses a binary morphological opening operator with a line piece having a certain minimum thickness (determined by the scale of the document) as a structuring element. To account for different melodies we have to perform the opening repeatedly, each time with a slightly rotated version of the structuring element (within a range of allowed angles). All resulting line pieces are instantiations of the simple note-beam object. Figure 7.7 shows found note beams in the music score of Figure 7.2(a).

![Figure 7.7: Detection of note beams. (a) Music score. (b) Detected note beams.](image)

Rest detectors A rest symbol dictates that a particular voice in a music score is silent for some measures of time. The duration of silence is expressed in different rest symbols. In music scores, four different rest symbols are distinguished representing whole, half, quarter, and eighth rests. Rests with smaller time durations than the eighth rest are constructed by attaching additional flags to the eighth rest in the same way as for note symbols, see Figure 7.1(b). In our application, only whole, half, quarter, and eighth rests are considered. The detectors are based on binary template matching, in which all rests have a different template. Again, the templates are acquired by copying and averaging some samples from a number of digitized scores.

Accidental detectors An accidental symbol indicates that a particular note symbol must be taken as it is (opposed to the key signature) or raised or lowered in pitch with a half or full semitone. Accidentals are placed some distance to the left of the note head of the note symbol. Accidentals are represented with different symbols, as can be noted from Figure 7.1(c). In our application only the flat, natural, and sharp accidentals are considered. The detectors are based on binary template matching, in which all accidentals have a different template.

### 7.3.2 Complex-object specification

In this section the specification of various complex music objects are given. Complex objects are defined by part and specialization links as well as by their geometrical relationships. Complex objects are specified for different purposes. First, specification provides a means of abstraction from a set of similar objects, e.g. one can
specify a complex rest object that is a generalization of whole, half, quarter, or eight rests. As such, in specifications of other complex objects one can simply use the complex rest object and not bother with the different possibilities of simple rest objects, allowing for efficient specification. Secondly, the specification of complex objects allows interpretation of a group of simple objects into one complex object and allows exploitation of their geometrical relationships. For example, if a vertical line piece does not have a certain geometrical relationship with a note head, then it cannot be a note stem and part of a note symbol. In this way, the number of false positive detections of simple objects can be reduced. Finally, the specification of complex objects makes it possible to calculate restricted search areas for the detection of simple objects from geometrically related (simple) objects. For instance, if a vertical line piece is detected then the search area for a note head can be restricted to small areas around the top and bottom of the line piece.

Rest  A rest symbol can be a whole, half, quarter, eighth, or sixteenth rest, as shown in Figure 7.1(b). Hence, the complex rest object is specified as a generalization of these rest objects, which all have a specialized detector. Figure 7.8 gives a visualization of the specification. The point attribute of the complex rest object represents the position of the rest symbol on the time axis of the stave. During interpretation it is calculated, i.e. copied from its instantiated specialization, according to an attribute-calculation function, see also Section 5.3.1.

![Figure 7.8: Specification of complex rest object.](image)

Accidental  Similar to the specification of the rest object, the accidental object may consist of double-flat, flat, natural, sharp, or double-sharp accidentals, as shown in Figure 7.1(c). This again is modeled using specialization links. Figure 7.9 shows the specification. In our application, the double-flat and double-sharp symbols are omitted for reasons of simplicity. The point attribute of the complex accidental object represents the position of the accidental. During interpretation, it is calculated from its specialization object and is used to determine its geometrical relationship with the note symbol.

Note symbol  In the same way as the specification of the rest and accidental objects, the note object may consist of whole, half, quarter, eighth, or smaller notes, see
Figure 7.9: Specification of complex accidental object.

Figure 7.10: Specification of complex note object.

Figure 7.1(a). This is also modeled using specialization links. Figure 7.10 shows the specification. For simplicity, sixteenth and smaller notes are omitted in our application. The half, quarter, and eighth notes are complex objects themselves and are discussed in the following paragraphs. The complex note object has a point attribute that represents the position of its note head. It is used to calculate the note's pitch and to position the note on the time axis as well as to relate the note with accidentals.

Half-, quarter-, and eighth-note symbols The complex half-note and quarter-note objects both consists of two parts: a note head (blank for half notes and solid for quarter notes) and a note stem, see Figure 7.1(a). The geometrical relationship between the note stem and the note head is specified with the standard distance relationship, as introduced in Section 5.3.1. Furthermore, the standard judgment function for this relationship is used as well as the standard pruning function vector end points to specify the search area for the note head on basis of the note stem. Figure 7.11(a) shows the specification of a quarter-note object, an application of the pruning function is given in Figure 7.12(a). The complex eighth-note object comprises three parts: a solid note head, a note stem, and a note beam (or flag). Two geometrical relationships are defined for this object. One geometrical relationship defines a standard distance relationship between the note head and the note stem,
Figure 7.11: Specification of complex object (a) quarter note and (b) eighth note.

Figure 7.12: (a) Quarter note: application of the standard pruning function vector end points to set the search area for the note head on basis of the note stem. (b) Staff lines: application of the standard pruning function vector end points to set the search area for the next staff line of the stave.

the second relationship defines a standard distance relationship between the note stem and the note beam. Again, standard judgment and pruning functions are used. Note that the eighth note is an extension of the quarter note with an additional beam or flag, as a result this may cause interpretation conflicts when quarter and eighth notes are found at the same position. The false positive detections of quarter notes in eighth notes are resolved by conflict resolution during the interpretation phase, as discussed in Section 7.3.4.

Stave A stave consists of five parallel running staff lines, a clef, and two bar lines. To form such a complex object, a complex staff-lines object is created that has five staff lines as parts, which are geometrically related with standard parallel geometrical relationships. The relationships have standard judgment and pruning functions. Figure 7.13 gives a visualization of the specification. An example application of the pruning function that sets the search areas for staff lines on the basis of other staff lines is depicted in Figure 7.12(b). The staff-lines object is geometrically related with the clef to form a complex stave object, which is in its turn related with two
bar lines to make a complex Stave object, see also Figure 7.14. A Stave object has a MAER attribute that represents the minimum-area encasing rectangle around the five staff lines and is used in the calculation of the pitch of a note.

**Additional complex objects** In order to determine the pitch of a note symbol, we must relate the note to the stave. To do so, a complex Stave-Note object is created which inhibits a Stave and one Note object, see Figure 7.14. This complex object has a pitch attribute that is calculated from the relative position of the note head (represented with the point attribute) to the stave (represented with the MAER attribute). The Note and Stave object have a standard distance relationship which specifies that a note must be positioned within some distance of the stave. A similar Stave-rest object is defined to relate rests to one stave.
7.3.3 Interpretation order

Interpretation starts with the detection of staff lines, as they are the most easy to detect and do not generate many false positive detections. Five parallel staff lines are then grouped into one staff-lines object, which is related with the clef to form an instantiation of the stave. When the Stave object has been determined by finding two additional bar lines, interpretation continues with hypothesizing note symbols to form instantiations of Stave-Notes. Note symbols are detected by first hypothesizing their note stem and using the resulting instantiations to look for note heads. The same procedure is performed for rest symbols. This interpretation order is accomplished by assigning the appropriate priorities to the different symbols, see Figure 7.14. After the interpretation, the resulting instantiations of Stave-Notes and Stave-rests are sorted horizontally by means of the point attribute, and vertically by means of the MAIR attribute of the Stave. The sorted notes and rests can then be converted to a standard MIDI file for music synthesis.

7.3.4 Conflict resolution

As mentioned earlier, during interpretation conflicts between different instantiations of music objects may rise. Some of these conflicts are due to the specification of the application domain. For example, the specification of a quarter note and an eighth note are apart from the eighth-note beam (or flag) similar. This may result in conflicting instantiations of quarter notes and eighth notes at the same position. Figure 7.15 depicts an example of such a conflict. For this kind of conflicts, the conflict-resolution engine is instructed (by the conflict-rule file) to choose for the eighth-note symbol and delete the instantiation of the quarter note. This choice is motivated by the fact that when both instantiations are possible the eighth note must prevail because it is an extension of a quarter note. Similar conflicts are specified

\[ \text{Figure 7.15: Interpretation conflict between a quarter note and an eighth note.} \]
for notes and dotted notes, i.e. notes with an extended (50%) time duration, which is represented with an additional dot to the right of the note head. In this case, the conflict-resolution engine is instructed to choose for dotted notes.

7.4 Examples

In this section interpretation results are shown of some samples of music scores. The music-score images used in the experiments are scanned from the book *The Joy of First Classics*. This book contains easy (piano) pieces by master composers for beginners and early-grade pianists [2]. The images are scanned with a Mustek ScanExpress 600 SEP at 300 dpi and quantized to 256 gray values. When necessary an image was manually rotated until it was horizontally aligned. The binarization of the images were obtained by sharpening, thresholding, and pre-processing (Chapters 2, 3, and 4) the gray-scale images. A part of the binary images were used to set the parameters of the specialized detectors, i.e. size of the structuring elements, selection of binary templates, etc. The other part was used to test our music-interpretation system. Figure 7.16 shows some interpretation results.

In these examples, conflict resolution was needed in Figures 7.16(a) and 7.16(b) to produce the final interpretations. In Figure 7.16(a) conflict resolution was needed to resolve the conflicting instantiations of quarter notes and eighth notes at the same position. As mentioned in Section 7.3.4, the conflict-resolution engine is instructed (by the conflict-rule file) to choose for the eighth-note symbols (E(b2), E(a2), E(g1) and E(a2)) and delete the instantiations of the quarter notes. In Figure 7.16(b) conflict resolution was needed to resolve the conflicting instantiation of a half note and a three quarters note (H.(c1)) at the same position.

Although the results are encouraging, some final remarks concerning the applicability of the music-interpretation system must be made. Firstly, the interpretation system requires that the key and time signature of a music score are entered manually before processing the score. Secondly, it is unable to recognize that bars or segments must be repeated. An example can be seen in the score of Figure 7.16(a). Finally, the system has great difficulties with interpreting scores that are not horizontally aligned or that have a different scale than the scores used to optimize the system. This stems from the fact that most detectors are not scale invariant. The horizontal alignment and scale of the score may be obtained by first detecting the staff lines and measuring their angles and the distance between two adjacent lines.

7.5 Conclusions and summary

In this chapter we described the implementation of a music-score interpretation system which uses our general framework for document interpretation. We showed the specification of complex music objects by means of part and specialization links and geometrical relationships, and the implementation of simple music objects detectors using image-processing and pattern-recognition techniques. Although our main design goal was to demonstrate the general applicability and flexibility of our document-interpretation system, we also showed music-score interpretation results that are satisfying for demonstration purposes.
Figure 7.16: Music-interpretation results. (a) First bar of Canario. (b) First bar of Minuet. (c) First bar of Old German dance. The symbol E(g1) denotes an eighth note g in the first octave as set by the cleff, E(g2) denotes an eighth note g in the second octave, etc. The same applies for half (H(.)) and quarter (Q(.)) notes.
Chapter 8

Discussion

In this thesis a knowledge-based framework for document interpretation is proposed. The main input of the system is a binary image of a scanned document. As such, a fair amount of our effort has been devoted to the development of a binarization technique. The developed technique includes a sharpening and thresholding step followed by pre- or post-processing of binarization errors. Our contribution to document-image interpretation is the extension of the common semantic-network approach with (multiple) detectors and conflict resolution, which both can aid in the pursuit of lower numbers of false-positive interpretations of application-domain objects.

8.1 Reflections on binarization

The importance of the binarization step is not acknowledged in many of the current document-processing systems. The binary output of a scanner is considered adequate for the interpretation of a document. Furthermore, a number of approaches apply vectorization on the binary image data, which blocks interpretation based on morphological features. We have the opinion that a binary image is one of the important inputs. Hence, the design of a binarization method must not be taken lightly. The advantages of binary image data are twofold: it requires less computer memory than gray-scale image data and it may provide sufficient information to discriminate between application-domain objects that are morphologically very distinct. For example, it is computationally more efficient to bootstrap the interpretation with a detector which takes only binary image data as input than with a detector which needs gray-scale image data. In addition, the bootstrap results can then be used to constrain the search areas for (more specialized) detectors that operate on gray-scale image data to find objects that are harder to detect solely on the basis of binary image data.

The sharpening operator is the first step in our binarization technique. The main assumption about it is that the image to be sharpened represents a black-and-white document that is discretized by a scanner with an isotropical PSF. Problems can arise if the image contains more than two principal brightnesses or if regions have texture or a non-uniform gray value. Although, the sharpening operator can
possibly cope with more than two principal brightnesses, it assumes that regions have a uniform brightness. As a result, highly textured images are hard to sharpen. An obvious solution is to let a derivative operator determine whether it is possible to sharpen, and to incorporate this derivative in the PDE. The approach by Perona and Malik [84] uses a PDE which has a similar effect: edges remain untouched and noisy areas are blurred on the basis of a first-order derivative. The assumption of isotropy of the PSF can easily be violated if two different physical processes determine the horizontal and vertical spatial resolution of the scanner. To sharpen an anisotropically blurred document, one can alter the shape of the structuring function. We suggest the application of quadratic or ellipse-shaped flat structuring functions. A last point of criticism is the influence of noise on the sharpening process. The dilation and erosion operators are highly sensitive to noise by definition: they are a local maximum and minimum operator. The effects of noise are most noticeable when we sharpen with a large number of iterative applications. The sequence of images then reveals many “walking” minima and maxima that corrupt the image. A solution is to replace the local maximum and minimum operators with percentile filters, which are less sensitive to noise. A further extension of the sharpening operator may be found in the use of structuring functions that are neither flat nor parabolic. We showed that as long the structuring function is concave, the operator has a sharpening effect on an image. Other candidates for the structuring function are, for example, cones, poweroids, and Gaussian functions. The list of concave structuring functions is of course limitless. Note that not all of these are dimensionally decomposable w.r.t. dilation and erosion, as the parabolic structuring function is, and therefore their implementation may be less computationally efficient. However, the different shapes of the structuring functions may give better sharpening results for certain applications.

The second step of our binarization technique, i.e. automatic thresholding with hysteresis, is an improvement on all automatic global-thresholding techniques. It assumes, however, that the image to be thresholded consists of two principal brightnesses with normally distributed gray values. If this assumption is not true, then performance is suboptimal. Although thresholding with hysteresis is introduced to obtain a binary segmentation of a gray-scale image, we see no barriers preventing further extensions of the method to produce segmentations into more than two regions by the use of more thresholds. This implies, however, a modification of the error functions to introduce more than two classes of pixels.

We showed that binarization errors, like the adjoining of digits, are inevitable when one binarizes a document automatically. There is no way to circumvent them because they originate not only from imperfections of the binarization method, but also from imperfections of the document itself. We gave a correction method for binarization errors that is based on (geometrical) relationships between objects that exist in an application domain. The method assumes that adjoining objects can be characterized by one geometrical relationship. If similar objects, for example conduit pieces differ in size, the proposed representation of a geometrical relationship may prove inadequate. For the case that the correction method cannot be applied successfully, we proposed a prevention method that does not require a model of objects and relationships in the application domain. Its major drawback, however, is that it cannot solve imperfections of the document, while the correction method can.
This limits the applicability of the prevention method. Another point of discussion is when to correct binarization errors. Do we opt for a data-driven (bottom-up) approach or a correction scheme that is driven by the interpretation process (top down)? In a data-driven approach we check all binary image data for binarization errors. This method is not efficient and can cause unwanted corrections. A top-down approach can restrain the search area to regions where detectors failed to detect an object. A point of concern w.r.t. the top-down approach is that the bootstrap of the interpretation may be affected if a number of objects cannot be instantiated due to binarization errors.

By now it is clear that the main assumption of the binarization technique is that the image to be binarized is a scanned black-and-white document. Further extensions of the technique may be found in assuming more principal brightnesses or different degradation models of the image. One must be aware, however, that a segmentation of an image into meaningful regions may not be acquired on pixel statistics alone; it can be beneficial to use a priori domain knowledge on the objects to be found. Then, we arrive at the point of detectors and interpretation. In our application the binarization of the image can be found relatively easy. For other image-interpretation systems this may not be as simple. As such, one must weigh the effort put in the design of a binarization method and the effort devoted to implement specific object detectors. If the images to be processed are easy to segment, binarization greatly simplifies the detector design. On the other hand, if the images to be processed are hard to segment (texture, non-uniform illuminated regions), it is better to skip the segmentation entirely and devote all effort to object-specific detector design.

### 8.2 Reflections on interpretation

Many of the current document-interpretation systems employ a knowledge-based approach to perform their task. In such systems, application-domain knowledge is represented in a form that can capture knowledge on domain objects as well as knowledge on document structure. A large number of approaches facilitate an additional representation for control knowledge and include an interface to procedurally embedded knowledge. The profit gained by applying a knowledge-based approach is that the system can be modular and adaptable to other application domains because it provides the domain knowledge in an easy-to-use specification. The semantic-network implementation we have adopted proved to be adequate for specifying knowledge on our application domains (utility maps and music scores). Declarative knowledge can easily be fitted into our system by means of specifying concepts, links, and geometrical relationships. Further, procedural knowledge can be provided with special detector, judgment, and attribute libraries. The detectors as well as the judgment and attribute functions conform to a pre-defined interface. Consequently, they can easily be replaced by new versions and the interface facilitates a separate design. Special toolboxes exist that can aid in the optimization of the performance of detectors. Besides an object-oriented view on detectors, we argued for the use of multiple detectors to increase their robustness. We employed classifier-combination techniques to reduce the number of false positive detections.
Concerning the interpretation mechanism, we chose for a priority queue as a fundamental data structure. It facilitates a mixed bottom-up and top-down approach to image interpretation and can be controlled if different priorities are assigned to application-domain objects. Although we showed that multiple detectors increase the system's robustness, we provided an additional conflict-resolution scheme to fully exploit the knowledge on document structure, resulting in a greater reduction of false positive detections and interpretations. The scheme proposed is highly flexible: one can specify whether an interpretation conflict is occurring, as well as when and how to resolve it. This flexibility is made possible by explicitly specifying interpretation conflicts.

The use of detectors in our interpretation system implies that one has to choose which objects in an application domain are simple and have a detector, and which objects are complex and are made out of other objects. In our application, some of these choices were made on basis of heuristics, but in general, two approaches to the endowing of detectors can be distinguished: one approach is to think in terms of meaningful objects in the application domain; the other approach focuses on the capabilities of detection techniques. Writing detectors for meaningful objects in the application domain may cause implementation problems, but makes the connection with more complex application-domain objects simple. Furthermore, it allows the use of application-domain models that already exist. The other approach has the benefit that it may result in higher detection rates. The connection to the application-domain model is harder to make, however. For example, one can design detectors for objects which correspond with a graphical primitive, which may have no direct correspondence to a meaningful object in the application domain. In this way, a detector can be built that classifies on morphological features of the graphical primitive; it is easy to implement and will probably have high detection rates. Besides having to choose objects which have a detector, we have to make a choice w.r.t. the implementation of the detectors. In our system, most of the detectors use graphical primitives from the binary image data to detect objects. Calculation of geometrical and morphological features from graphical primitives is easy and make it simple to use (a combination of) pattern-recognition techniques. Further extensions may be found by incorporating more single detectors or applying different combination techniques. For the case that the binary image data is unreliable, e.g. for objects drawn on the back of the utility map, we implemented more specialized detectors using gray-scale filters and local adaptive thresholding techniques. When for an application domain a binarization of the image cannot be made, or when it is unreliable, one has to opt for this latter, more time-consuming option.

Conflict resolution proved to increase the flexibility and reliability of our system. The extension of a conventional system with an explicitly specified conflict-resolution scheme implies that a specification of all possible interpretation conflicts is necessary. The number of possible interpretation conflicts grows exponentially w.r.t. the number of application-domain objects. Although a fair portion of the specification process can be automated by means of drawing rules and conventions, some of the conflicts still have to be specified manually. Furthermore, the conflict specification is limited to the case in which precisely two complex objects are in conflict over one or two instantiations. Complex situations may exist where more than two complex objects are mutually in conflict over more than two instantiations. In such cases,
we need an extended specification; possibly a semantic network which can represent an arbitrary number of conjunctions or disjunctions of conflicts as well as a different conflict-resolution engine which takes into account the number of conflicting complex objects.

A final point of discussion concerns the integration of the two sources, i.e. detectors and complex objects, that give information on the map contents in our interpretation system. Detectors provide information which is based on morphological features of objects, whereas interpretation of complex objects gives information on the context of objects. Although in the examples of applications of our interpretation framework, objects and context were clearly defined, future applications may encounter images in which the two sources are unbalanced. In such an application, one has to make a choice between two different approaches based on the reliability of the information source. If the objects are easily detectable and the object structure is ambiguous, one can choose for an approach with strong detectors and a weak context. In this approach, one puts the largest amount of effort in the design of the detectors, whereas conflict resolution is curtailed to a minimum because context information is not reliable. On the other hand, if objects are hard to detect and the object structure is unambiguous one can choose for an approach with weak detectors and a strong context. In this approach, detectors are designed to have a high detection rate with possibly a high number of false positives; the conflict-resolution scheme is used to eliminate the large number of false positives.

8.3 Further research in computer vision

If we shift our focus from interpretation of documents towards computer-vision applications, the applicability of our framework is less obvious. The three application domains for which our interpretation system was tested were (closely) related and regarded interpretation of highly structured documents, in which objects and geometrical relationships can be interpreted reasonably well. The task of interpreting natural images is much harder than processing documents. In computer-vision applications, the objects and the (geometrical) relationships between objects are generally not clear and often ambiguous, making interpretation a difficult job. For example, in face-recognition applications [89] the objects of interest, i.e. eyes, nose, mouth, hairline, can have a wide variety of appearances: eyes may have different colors and shapes (they can even be closed). Fortunately, the relationships between the different face features are strong and may provide just enough information to solve any ambiguous object detections. As a result, some researchers emphasize the importance of context in computer-vision applications, claiming that some images can be interpreted better when structure is recognized instead of entities [106]. We have the opinion that our conflict-resolution scheme can be of great interest to such structure-based approaches: our scheme provides additional specification of knowledge on context. It provides a greater functionality to reason with context than conventional systems because it allows the specification of how much context information must be gathered to resolve interpretation conflicts due to falsely detected entities.
Another approach to the difficult task of computer vision is found in the synergy of different detection and interpretation techniques: multi-agent systems. They embody a less rigid attitude to interpretation and rely on the trials of a lot of different algorithms which communicate with each other either by means of a shared data structure (blackboard) or directly. The system does not have an exact scheme to follow, but relies on a number of different agents, which together have sufficient functionality to arrive at an interpretation. Again, robustness of a system is found in the combination of different algorithms, as with multiple detectors. In such systems, different autonomous agents exist that can detect specific features in an image using several detection techniques (artificial neural networks, template matchers). Furthermore, some agents can detect geometrical relationships between detected objects or can group different detection results to obtain more complex interpretations. The advantage of multi-agent systems is that the architecture allows for direct communication between the different agents to exchange information on certain detections. This can be valuable or even essential to setting the algorithm's parameters. However, a multi-agent system is hard to control because of the autonomous behavior of its agents. Much time must be devoted to letting the agents operate and communicate with each other to accomplish a certain task. It is easy to see that apart from its control architecture, multi-agent systems are similar to conventional knowledge-based systems: the same methods and knowledge are employed. Consequently, parts of our system can easily be converted to agent-like structures. The concept of multiple detectors is already present in most multi-agent systems by definition, but a conflict-resolution agent could greatly improve the performance of such multi-agent systems.
Bibliography


The field of scientific research in computer vision has become mature. Current achievements, which mainly consist of model- or knowledge-based approaches, demonstrate to be applicable in a wide variety of purposes, ranging from medical applications to automated document processing. As such, it can provide solutions to complex problems which exist within a large number of companies. One of these problems is the conversion of paper public-utility maps to a digital format. The digital storage of maps has a number of advantages, the most important being the efficient retrieval and maintenance of information. Public-utility organizations still have a large number of paper maps with information relevant to their service networks that is not available in a digital format. Current conversion techniques which public-utility organizations use require a lot of human participation along the complete conversion process, and are therefore very expensive. For these reasons, an automated conversion technique is desired.

The main contribution of this thesis concerns the introduction of a knowledge-based system for the interpretation of public-utility maps. Because there are a great number of different types of maps, which may be of varying quality and drawn with different conventions, a flexible interpretation technique is required. We address flexibility by proposing a knowledge-based system in which the application-domain knowledge can be adapted to the specific type of maps to be interpreted. Our knowledge-based approach makes it simple to specify the application-domain knowledge on map objects, the geometrical relationships between map objects, and to establish a hierarchy of objects. Our approach is extended with the possibility to specify control knowledge and knowledge on interpretation conflicts. This extension also enhances the reliability of the conversion. The reliability requirement must be met to ensure acceptance of the system within an organization and to achieve a low level of operator-machine interactions. The concept of multiple detectors which incorporate multiple object-detection techniques, and the application of conflict resolution within the interpretation increase recognition accuracy and reliability of the system.

Because the majority of documents consist of black text and black objects on a white paper background, the binarization of a scanned document is an important input for any document-interpretation system. As such, a fair amount of our contribution lies within the development of a binarization technique. The developed technique includes a sharpening and thresholding step followed by pre- or post-processing of binarization errors.
The sharpening step suppresses the blur, which is due to the scanner's lens system, in the scanned document. As a result, the choice of threshold(s) in the following thresholding step becomes less critical. The sharpening operator we describe is an extension of the operator as defined by Kramer. We show that Kramer's operator is actually an element of a general class of morphological image operators for sharpening digitized images. Furthermore, we show that image operators using a concave structuring function have sharpening properties and can be characterized by a partial differential equation that incorporates the sign of the Laplacian of the image to be sharpened. We give proofs of the sharpening properties as well as a derivation of the PDE for which we use the recently introduced slope transform. The parameters, i.e. the number of iterations and the size of the structuring function of the image operator, can be determined on the basis of an estimate of the amount of blur present in the image. We show for discrete algorithmic implementations of the image-operator class that image operators using a parabolic structuring function have an efficient implementation and isotropic sharpening behavior.

There exists an enormous number of (automatic) global thresholding techniques, which makes it hard to choose a suitable technique. The disadvantage of all these techniques is that they solely rely on histogram information and use one global threshold. An alternative to global thresholding techniques, thresholding with hysteresis, has proven to perform better than global thresholding for some practical applications. Thresholding with hysteresis differs from global thresholding in the sense that thresholding with hysteresis uses two thresholds instead of one. Furthermore, thresholding with hysteresis also incorporates local spatial information, as opposed to global thresholding, which solely relies on pixel statistics. Unfortunately, an automatic selection procedure of the two thresholds of the hysteresis was lacking. In this thesis, we outline a novel procedure that automatically sets the two thresholds and show that this procedure is optimal. In order to do so, we define thresholding with hysteresis within the framework of mathematical morphology and we derive an error function that measures the number of misclassified pixels on the assumption that the image contains a mixture of normally distributed object and background pixels. Experiments show superior results for automatic thresholding with hysteresis when compared with the minimum-error thresholding algorithm.

Even when we have a perfect binarization method, binarization errors can occur. This is due to the fact that binarization errors occur because of an imperfect document as well as an erroneous binarization method. It is obvious that refinement of the binarization method is not sufficient to abandon the problem if the assumption on the first cause is not made. In our utility-map domain, the biggest problem of binarization errors is the problem of overlapping characters. As above, two causes for adjoining objects can be distinguished: objects already adjoin on the paper document, or the binarization method is erroneous. Therefore, we proposed two new solutions to solve problems of both the causes. One solution works on the gray-scale image data (pre-processing method: prevention); the other solution operates on the binarization result (post-processing method: correction). The difference between them is that while the prevention method cannot solve all occurrences of adjoining objects, the correction method can, but it requires a model of the geometrical relationship of the adjoining objects to do so. Experiments on utility maps show a high performance for the correction method.
After describing the steps taken to obtain a binarization of a scanned document, we describe a document-interpretation system. The interpretation system is fitted into a general framework for (image) interpretation. Within the framework, it is possible to represent a variety of knowledge types by means of a semantic network. The knowledge types range from specification of the application-domain knowledge by specifying objects and geometrical relationships between objects, to specification of procedural knowledge: interpretation order and specification of interpretation conflicts. For each object in an application domain, one can specify whether it is a simple object and has a specialized detector, or whether it is a complex object and is composed of a number of objects having a specific geometrical relationship. We propose a new concept which involves multiple detectors. The concept proves to result in fewer false positive detections of objects. Additionally, interpretation conflicts can be represented in the framework by specifying the context in which a conflict may appear and when and how it should be resolved. We show the necessity of explicitly specified context-dependent conflict rules and we introduce a conflict-resolution scheme based on learning from examples. Within this scheme, we propose a new judgment function for the judgment of instances of the specified application-domain objects. We demonstrate the general applicability of the system with two additional applications: utility-map reconstruction and music-score interpretation.

When a utility map is interpreted successfully, it is still not suitable for digital storage in a GIS, because the map and its interpretation suffer from local scale, translation, and rotation distortions. We propose a reconstruction of utility maps that uses a correspondence between a utility map and a geometrically correct base map based on mutual topography as well as relational information to reconstruct the utility map such that its topographical position matches the actual position of the network. A correspondence cannot be found easily because of differences between the topography on the utility map and topography on the base map. The differences originate from local scale differences on the utility map and the use of different drawing conventions for both kinds of maps. A solution is found with an inexact matching procedure. The interpretation system provides the necessary information about geometrical relationships between map objects for the reconstruction process. The interpretation applies conflict resolution to deal with uncertainty in the location of the objects on the utility map. To preserve the shape of the reconstructed network, we reconstructed dimensions and conduits in a different manner.

Besides applications of the interpretation system in the utility-map domain, we describe a third application of our system: music-score interpretation. By modification of the knowledge files and by implementing detectors for simple music objects, the interpretation system can be changed to interpret music scores. Some example implementations of detectors are described in detail. They are based on binary morphological image operators and template-matching techniques. Additionally, the specification of complex music objects and conflicting interpretations are discussed and we show some examples of interpretation of (easy) piano pieces.

The thesis ends with a discussion on the proposed binarization and interpretation techniques. For the binarization, we identify the assumptions about the image characteristics, and discuss whether they can be generalized to provide an applicability of the binarization method to a larger class of images. The main assumption about the binarization technique is that the image to be binarized is a scanned
black-and-white document. Further extensions of the technique may be found in assuming more principal brightnesses or different degradation models of the image. The discussion on the interpretation part focuses on the contraposition between detectors and conflict resolution, corresponding with the interpretation of objects and structure in the image. We discuss the general applicability of the system by noting that the three application domains for which our interpretation system was tested are (closely) related and regard interpretation of highly structured documents, in which objects and geometrical relationships can be interpreted reasonably well. The task of interpreting natural images is considered to be much harder than that of processing documents. Shifting the focus towards computer vision, a point is made by stating that context-dependent conflict resolution may prove to be a valuable tool when the importance of image structure prevails over the identification of meaningful objects or regions in the image.

Summary of the thesis: "Document interpretation applied to public-utility maps"

Samenvatting

Computer vision is een volwassen onderzoeksgebied geworden. De huidige syste- men, voornamelijk model- en kennisgestuurd, laten een toepasbaarheid zien voor een groot aantal deelgebieden, variërend van medische applicaties tot automatische documentverwerking. Als zodanig kan het oplossingen bieden voor complexe pro- blemen die bestaan in een groot aantal bedrijven. Eén van deze problemen is de omzetting van papieren beheerkaarten van leidingnetten van nutsbedrijven naar een digitaal formaat dat geschikt is voor een Geografisch Informatie Systeem (GIS). Het digitaal opslaan biedt veel voordelen, de belangrijkste is het efficiënte beheer van informatie over het netwerk van leidingen. Nutsbedrijven hebben nog steeds een groot aantal kaarten met relevante informatie over hun netwerken die nog niet be- schikbaar zijn in een digitaal formaat. De conversietechnieken die de nutsbedrijven gebruiken zijn echter erg arbeidsintensief en daarom erg duur. Een automatische conversietechniek is dus gewenst.

De belangrijkste wetenschappelijke bijdrage van dit proefschrift betreft de in- troductie van een kennisgestuurd systeem voor de interpretatie van beheerkaarten. Omdat er een groot aantal verschillende typen kaarten zijn, die ook nog van kwaliteit kunnen verschillen, is een flexibel interpretatiesysteem vereist. Flexibiliteit wordt geboden door een kennisgestuurd systeem te introduceren waarin domeinkennis kan worden aangepast aan het type kaarten dat moet worden geïnterpreteerd. Een kennisgestuurd systeem maakt het eenvoudig om domeinkennis over kaartobjecten en geometrische relaties tussen kaartobjecten te specificeren en een hiërarchie van com- plexe objecten te vormen. Verder kan er kennis over controle van het systeem en interpretatieconflicten worden gespecificeerd wat tot een grotere betrouwbaarheid van het systeem bijdraagt. Aan de systeemis van betrouwbaarheid moet worden voldaan om acceptatie van het systeem binnen een organisatie te waarborgen. Naast deze mogelijkheden van specificatie introduceren wij het concept van gecombineerde objectdetectoren. Deze detectoren kunnen objecten in het beeld betrouwbaar her- kennen door een combinatie van detectietechnieken te gebruiken. Verder introdu- ceren wij de toepassing van conflictresolutie binnen het interpretatiesysteem om de nauwkeurigheid van de interpretatie en de betrouwbaarheid van het systeem nog verder te verhogen.

Het grootste deel van alle documenten bestaat uit documenten met zwarte tekst, symbolen en grafieken op een witte papieren achtergrond. Wanneer deze documenten worden gedigitaliseerd met behulp van een scanner dan is de segmentatie van het digitale beeld in beeldpunten van voorgroond (inkt) en achtergrond (papier) een belangrijke bron van informatie voor ieder systeem voor documentinterpretatie. Als
zodanig bestaat een deel van onze wetenschappelijke bijdrage uit de ontwikkeling van een segmentatietechniek. De ontwikkelde methode bestaat uit het opscherpen en drempelen van het digitale beeld, gevolgd door correctie van segmentatifouten.

Het opscherpen van het digitale beeld gebeurt met een filter dat de onscherpte in het beeld als gevolg van het lensysteem van de scanner onderdrukt. Het filter maakt de drempelkeuze in de volgende stap minder kritisch. Het filter dat wij beschrijven is een uitbreiding van het filter dat door Kramer is geïntroduceerd. Wij laten zien dat het filter van Kramer eigenlijk een element van een klasse van opscherpfilters is. Verder laten wij zien dat concave filters opscherpeigenschappen hebben en kunnen worden beschreven met een partiële differentiaalvergelijking welke het teken van de Laplacian van het op te scherpen beeld bevat. Bewijzen van de opscherpkwaliteiten en de afleidingen van de differentiaalvergelijkingen worden gemaakt met behulp van de recent geïntroduceerde Slope transformatie. De parameters van het filter, die zijn het aantal iteraties en de grootte van het filter, kunnen worden bepaald aan de hand van een schatting van de onscherpte in het beeld. Voor discrete implementaties van het filter laten wij zien dat parabolische filters een efficiënte implementatie hebben en een gelijkmatig rond opscherpgedrag vertonen.

Er bestaan veel globale drempeltechnieken, een keuze voor een geschikte techniek is daarom moeilijk. Een nadeel van al deze technieken is dat zij slechts één drempel gebruiken en deze bepalen aan de hand van de statistiek over het voorkomen van grijswaarden. Een alternatief voor globale drempeltechnieken, drempelen met hysteresese, heeft voor praktische toepassingen bewezen beter te presteren dan globaal drempelen. Drempelen met hysteresese verschilt van globaal drempelen in het feit dat het twee drempels gebruikt in plaats van één. Verder gebruikt het ook spatiale informatie in tegenstelling tot globale drempeltechnieken die alleen statistiek van grijswaarden gebruiken. Tot nu toe ontbrak een automatische procedure voor de selectie van de twee drempels. In dit proefschrift beschrijven wij een nieuwe procedure die automatisch de twee drempels bepaalt en wij laten zien dat deze procedure optimaal is. De procedure wordt beschreven aan de hand van een definitie van drempelen met hysteresese in de mathematische morfologie. De procedure gebruikt een foutmaat dat het aantal fout geclassificeerde beeldpunten telt onder de veronderstelling dat het beeld bestaat uit een mengvorm van normaal verdeelde voor- en achtergrondbepunten.

Zelfs wanneer we een perfecte segmentatiemethode hebben, kunnen er segmentatifouten voorkomen. Dit komt doordat fouten niet alleen ontstaan door een imperfecte segmentatiemethode maar ook door een imperfect document. Het is duidelijk dat het verbeteren van de segmentatiemethode niet voldoende is om fouten veroorzaakt door een imperfect document op te vangen. Voor onze kaartentoepassing is het grootste probleem twee overlappende karakers. Overlappende karakers hebben eveneens twee oorzaken: de karakers overlappen al op de kaart of de segmentatiemethode is fout. Daarom bieden wij twee oplossingen om beide oorzaken op te vangen. Eén oplossing werkt op het digitale grijswaardenbeeld (voorbewerking: preventie), de andere methode werkt op het resultaat van de segmentatiemethode (nabewerking: correctie). Het verschil tussen deze twee is dat de preventiemethode niet alle voorkomende overlappende karakers kan oplossen, terwijl de correctiemethode dat wel kan maar een model van de geometrische relatie van de twee karakers nodig heeft om dat te doen.
Na de beschrijving van de stappen van de segmentatiemethode wordt het interpretatiesysteem beschreven. Het interpretatiesysteem wordt ingeënt in een algemeen raamwerk voor beeldinterpretatie. Binnen het raamwerk is het mogelijk om een variëteit aan kennis typen te representeren met behulp van een zogenaamd semantisch netwerk. De kennis typen kunnen lopen van domeinkennis over objecten en geometrische relaties tussen objecten tot procedurele kennis: volgorde van interpretatie en conflicten in interpretatie. Een semantisch netwerk is uitstekend geschikt om de domeinkennis over de te interpreteren documenten te representeren. Voor elk object in het toepassings domein is men namelijk speciferen of het een simpel object is welke een gespecialiseerde detector heeft, of dat het een complex object is welke is samengesteld uit een aantal objecten welke één of meer geometrische relaties delen. Wij introduceren een nieuw concept voor detectoren: gecombineerde detectoren. Gebruik van dit concept resulteert in minder foutpositieve detecties van objecten. Naast objecten en relaties, kunnen ook conflicten worden geregistreerd in het raamwerk door te specifieren in welke context conflicten kunnen voorkomen en wanneer en hoe ze moeten worden opgelost. Wij laten zien dat deze specificatie van context-afhankelijke conflicten nodig is om conflictiserende en niet-conflicterende situaties op de kaart te kunnen onderscheiden. Verder introduceren wij een schema om conflicten te lossen dat is gebaseerd op leren uit voorbeelden van conflicten. Als onderdeel van dit schema, stellen wij nieuwe functies voor de waardering van instanties van domeinobjecten voor. Wij demonstreren de algemene toepasbaarheid van ons systeem met twee extra toepassingen: reconstructie van kaarten en interpretatie van muziekschrift.

Wanneer een beheerkart van leidingnetten succesvol geïnterpreteerd is, dan is het nog steeds niet geschikt voor digitale opslag in een GIS omdat de kaart lijkt aan lokale verstoringen in schaal, translatie en rotatie. Om de kaart zo goed mogelijk op schaal te reconstrueren stellen wij een procedure voor die gebruikt maakt van een overeenkomst tussen een kaart en een geometrisch correcte basiskaart van het Kadaster en van relationele informatie over kaartobjecten. Deze overeenkomst berust op topografische informatie (huizen) die aanwezig is op beide kaarten. De procedure zorgt ervoor dat de reconstructie zodanig is dat de gecreëerde kaart de werkelijke positie van het netwerk beschrijft. Een overeenkomst kan helaas niet gemakkelijk worden gevonden omdat de topografie op de nutskaart en basiskaart behoorlijk verschillen vanwege verschillende tekens en conventies. Een oplossing is het gebruik van een inexact overeenkomst om met de verschillen om te gaan. Het interpretatiesysteem geeft informatie over de geometrische relaties tussen kaartobjecten voor het reconstructieproces. Voor het omgaan met onzekerheid in de lokatie van objecten op de kaart wordt conflictresolutie gebruikt. Om de vorm van het netwerk van leidingen te behouden, worden bematingen en leidingnetten op verschillende manieren gereconstrueerd.

Naast toepassingen van het interpretatiesysteem voor kaarten, beschrijven wij een derde toepassing van het systeem: herkenning van muziekschrift. Door het wijzigen van de gespecificeerde domeinkennis en het implementeren van nieuwe detectoren voor simpele objecten in muziekschrift kan het interpretatiesysteem worden aangepast om muziekschrift te interpreteren. Enkele implementaties van detectoren worden in detail beschreven. Ze zijn gebaseerd op binair morfologische beeldoperatoren en template-matching technieken. Verder worden de specificaties van
complexe muzieksymbolen behandeld en aan het eind worden een aantal illustraties van interpretaties van pianowerken gepresenteerd.

Het proefschrift eindigt met een discussie over de voorgestelde technieken voor segmentatie en interpretatie. Voor de segmentatie worden de aannames over de karakteristieken van het beeld geïdentificeerd en er wordt besproken of deze kunnen worden versoepeld om een bredere toepassing van de segmentatiemethode te verkrijgen voor een grotere klasse van beelden. De belangrijkste aannames van de segmentatiemethode is dat het digitale beeld dat moet worden gesegmenteerd een gedigitaliseerd zwart-wit document is. Verdere aanpassingen van de segmentatiemethode kunnen worden gevonden in het veronderstellen van meer grijswaardentinten in het beeld of andere methoden voor digitalisatie. De discussie over het interpretatiesysteem spist zich toe op de verhouding tussen detectoren en conflictresolutie, die corresponderen met de interpretatie van objecten en structuur van het beeld. Wij stellen de algemene toepasbaarheid van het systeem ter discussie door te concluderen dat de drie toepassingsgebieden waar het systeem op is getest erg gerelateerd zijn en interpretatie van gestructureerde documenten betreffen waar de objecten en geometrische relaties redelijk goed kunnen worden geïnterpreteerd. De taak om natuurlijke beelden te interpreteren wordt als veel moeilijker beschouwd dan een document automatisch te verwerken. Maar voor toepassingen in computer vision voorzien wij toch een grote waarde van context-afhankelijke conflictresolutie wanneer structuur van het beeld belangrijker wordt dan de identificatie van betekenisvolle objecten of gebieden in het beeld.

Samenvatting van het proefschrift: “Documentinterpretatie toegepast op beheerkaarten van leidingnetten van nutsbedrijven”

Acknowledgments

I have to acknowledge that I did not do this all by myself. I am in debt to so many, yet will only mention a few. Starting with the initiators of my Ph.D. position, I would like to express my thanks and gratitude towards professor Eric Backer and my supervisor Jan Gerbrands for the great opportunity they gave me to do scientific research in the Information and Communication Theory Group. Although both were occupied with positions outside the group, and I enjoyed the resulting freedom, I have to admit that I feel sorry that our co-operation did not start the way it ended. Furthermore, I am glad that Marcel Reinders became an additional supervisor; the scientific plural I adopted in this thesis is because of you. Thank you for rigorous proofreading of this thesis.

The first two years of my Ph.D. position, I worked in the TopSpin-PNEM project together with people from the Pattern Recognition Group, the University of Amsterdam, and TNO-TPD. As such, I spent some time at the other side of the Mekelweg and I would especially like to thank Jurrien, Jurgen, and Bernardus for the co-operation, which I enjoyed. The project supervision by Ton and Arnold was sometimes intense, but did leave me with some space to develop my own scientific ideas. Further, I would like to thank the people of TopSpin and the PNEM for sponsoring the project.

The project meetings were not the only occasions where I met project colleagues; I also met some of them at our monthly discussion group on knowledge-based image processing with people from the Pattern Recognition Group. As such, the discussion was mainly focussed on maps and I have to thank Albert Vossepoel and Rob Duin for remaining interested.

It seems funny that just half an hour can separate nerds having lunch from dining philosophers, something we have to blame on the great concentration of doctoranduses in our lunch group, I guess. Erik V., you are one of the most social persons I have ever known, you even lent me your dancing partner for two months! Cor, fellow-doctorandus in a tower of engineers, by now I know why you quote so much from the Intermediair: it is so much easier to read in the train. Edgar, you don’t say so much, but the way you can move a discussion from mortgage boredom to our favorite 13.00 subject reveals the hand of a master. Philip, you are the living proof that something good can come from below the rivers. From the other lunch group I would like to mention my former room mate Erik Mouw (I hope you stopped wondering about my musical taste), Peter (I really like the way you dance), Gerhard (living price list of computer parts), and André (wait, wait, wait, I know it!).
The reader of this thesis has been saved from the horror of my punctuation and
cyphenation habits by Mirjam Nieman, who shed a light on them. Your visualization
of technical terms (bottom-up processing, touching characters, top hat) would make
a very nice comic for engineers! Thank you.

From outside the office, I would especially like to thank my friends, once gathered
in and around the legendary band the Space Chickens, for their support and just
for the great times. I really enjoyed all our performances and rehearsals, and I
think we surely changed the definition of space rock! Edo, do not think we are sad
because you are again leaving to some distant place, no, we are crying because we
cannot come along. Mac, now again abroad, our lively discussion on shit shovelers
and people who spend all their lives in submarines and suffer from periscope vision
remain precious to me. Carlo, I learned a lot of you on how to play my guitar; I
hope you keep on rocking. Fabian, I liked traveling with you and hope you break
through as a photographer. Jeroen, you choose politics, no comment. Sheila, you
have proven to be a real friend over the last years; I hope you can keep some space
in your busy agenda for me.

My family gave up understanding what exactly I was doing some time ago, but
that was no real problem for me. It was always nice coming home, whether that was
necessary (dirty clothes) or not. Thank you for your support.

Infine, la mia seconda famiglia in Italia: Marina e Nonna Nanda grazie per
l’ospitalità durante il mio soggiorno in Italia. Mi dispiace che ancora non so fare i
cappelletti. Ornella, so quanto sei contenta che questa tesi sia finita. Non dimenti-
care, tesoro, che tu sei più importante per me.
Curriculum Vitae

Johannes Gerardus Maria (John) Schavemaker was born in Nibbixwoud, the Netherlands, on July 7th, 1970. In the year 1988, he obtained his Atheneum diploma from the Oscar Romero Scholenenmeenschap in Hoorn, the Netherlands. Directly after his secondary education, he went to Amsterdam to study Computer Science at the University of Amsterdam. Schavemaker carried out his M.Sc. project at the Koninklijke/Shell Exploratie & Produktie Laboratorium in Rijswijk, the Netherlands under supervision of Rein van den Boomgaard, Willem Epping, and Arnold Smeulders. Results of the project were presented at the International Symposium on Mathematical Morphology (ISMM94) in Fontainebleau, France. His thesis was entitled Segmentation in morphological scale-space. He received his M.Sc. in October 1994.

From October 1994 to October 1998, Schavemaker worked as a Ph.D. student at the Information and Communication Theory group of the Faculty of Information Technology and Systems at the Delft University of Technology. This work was partly carried out as part of the Dutch TopSpin-PNEM project Knowledge-based conversion of public-utility maps. The project resulted in a semi-automatic conversion system at the end of 1996. Journalistic publications about the project appeared in Delft Integraal, Delft Outlook, and De Ingenieur.

In February 1999 Schavemaker started working as a researcher at the Electro Optical Systems group of TNO-FEL.