ISBN ISBN 978-82-471-1136-9 (printed ver.)
ISBN ISBN 978-82-471-1137-6 (electronic ver.)
ISSN 1503-8181

Doctoral theses at NTNU, 2009:79

Ole Christian Eidheim

New Approaches for Representation and Segmentation of Organs in CT and MR Scans

NTNU
Norwegian University of
Science and Technology
Thesis for the degree of
philosophiae doctor
Faculty of Information Technology, Mathematics and
Electrical Engineering
Department of Computer and Information Science

NTNU
Norwegian University of
Science and Technology

Doctoral theses at NTNU, 2009:79

NTNU

NTNU
Norwegian University of
Science and Technology

Ole Christian Eidheim

New Approaches for Representation and Segmentation of Organs in CT and MR Scans

Thesis for the degree of philosophiae doctor

Trondheim, May 2009

Norwegian University of
Science and Technology
Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Computer and Information Science

NTNU
Norwegian University of Science and Technology

Thesis for the degree of philosophiae doctor

Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Computer and Information Science

ISBN 978-82-471-1136-9 (printed ver.)
ISBN 978-82-471-1137-6 (electronic ver.)
ISSN 1503-8181

Doctoral Theses at NTNU, 2009:79

Printed by Tapir Uttrykk

Abstract

Analysis of medical images is resource demanding and time-consuming, and automatic procedures are needed to reduce the workload of medical staff in a pre-operative planning phase. In this thesis, the main focus has been on methods that automatically segment CT and MR volume data, in particular new approaches for representation and segmentation of the liver, hepatic vessels and the kidney.

The two main contributions in this thesis are a new 3D skeleton procedure and a texture-based segmentation method. The skeleton procedure is iterative, without user-defined parameters, and produces a minimalistic representation of binary objects without known artifacts. Compared to previous work, this skeleton method produces more reliable results and does not need tuning for each individual representation task.

The new texture-based segmentation algorithm is used to segment the selected organs, where only a few parameters influence the end result. Moreover, the parameters of this method are relatively easy to set, and a wide parameter range yields acceptable results. This method is more robust than popular previously published procedures that are typically based on edge information.

Additionally, there are two minor contributions in this thesis. A new general representation of binary objects with an interior is presented. This representation is used to automatically derive the parameters of the texture-based segmentation method based on a statistical template. Furthermore, parallel processing on modern graphics cards and multiple CPU processors have been studied and compared to serial algorithms. A significant decrease in runtime was shown on many common image processing techniques in addition to the proposed texture-based segmentation algorithm.

Even though the results are promising, more research is needed before reliable analysis of medical volume data can be performed. In particular, a combination of the proposed techniques incorporated with shape-based and statistical models is suggested for future research. The contributions in this thesis, however, are noticeable and represent a step forward in deriving complete automatic procedures for segmentation of medical volume data.

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) in partial fulfilment of the requirements for the degree Doctor of Philosophy (PhD). This work has been conducted at the Department of Computer and Information Sciences (IDI), NTNU. The doctoral study was funded by IDI.

Acknowledgements

First and foremost, I would like to thank my supervisors Keith Downing, Lars Aurdal and Thomas Langø for valuable guidance and support throughout the PhD period.

Roy Jakobsen and Håkon Heuch have provided most of the datasets I have worked on. I would like to thank you and your organisations; St. Olav Trondheim University Hospital and Rikshospitalet University Hospital, respectively. Many thanks also to Tom Mala at the Interventional Centre at Rikshospitalet for assisting on the medical background of the project.

I am much grateful to Rune Bakken, Per Bjarne Løvsletten, Jon Olav Hauglid, Jørgen Løland and Jeanine Lilleng for interesting discussion and comments during my time at NTNU. Additional thanks to Rune Bakken for helping me proofreading this thesis. I also much appreciated the collaboration with Jo Skjermo regarding visualisation and parallelisation of image processing tasks through the use of GPUs. Furthermore, I would like to thank Jan Christian Meyer and Rune Jensen for optimising my multi-threaded C code.

Finally, I would like to thank all my friends and family for their support, and most notably my daughter Ulrikke Ødegaard Eidheim who was born during my work on this thesis.

Contents

List of Figures

Chapter 1

Introduction

Medical imaging has become common in the diagnosis and treatment of diseases. While the image modalities are helpful, such equipment is expensive and resource demanding. Therefore, to reduce the workload of medical staff, automatic measurements and improved visualisations of the modality data is needed.

One main challenge in order to perform automatic analysis on medical volume data is to separate the various tissues from each other. Their boundaries may be diffuse and the medical volumes are typically distorted by noise. Furthermore, while the general anatomy is similar from person to person, previous interventions and genetic variations lead to variations in the size, shape, and location of organs.

To separate organ tissues well, advanced techniques based on pattern recognition are typically required. While humans are experts in pattern recognition, we have not been able to achieve results anywhere close to human recognition artificially. This is one of the reasons why there are presently few implemented applications for automatic measuring and 3D visualisations of medical volumes.

If automatic procedures do not achieve the level of accuracy of the results needed in medical science, it is likely that these methods can still prove helpful. Even if the distinction of tissues is poor, it can be used in a partially user guided system where the user defines the parameters needed to achieve a sound boundary between tissues semi-automatically.

The various modalities are constantly improved and higher resolutions increase the possibilities to help patients. Increased resolutions, however, also challenge the hardware needed to make the necessary computations. With respect to image processing, careful balancing must be made between the quality of the result, practical gain, processing time and cost.

Possibilities to greatly increase the processing speed inexpensively have re-

cently become available through the increasing number of cores, e.g. processing units, in personal computers or specialised parallel hardware. The cores are parallel processor units that can be exploited through multi-threaded programming. Such programming is typically harder to formulate and not all algorithms can be implemented efficiently in parallel. Specialised hardware such as GPUs on modern graphic cards are even more specialised and constrained.

The motivation behind this PhD thesis is to improve existing methods and to apply new techniques to solve a number of typical segmentation problems in medical imaging. A common factor in the segmentation techniques to be studied is that they use advanced techniques such as pattern recognition and models to guide the segmentation process. These methods are typically derived from various scientific fields such as mathematics, statistics, physics and biology, making the thesis highly interdisciplinary.

1.1 Research questions

The focus of this thesis is on the following research questions:

- Is it possible to achieve an accurate and robust segmentation of the liver, hepatic vessels, liver tumours and the kidney with previously derived image processing techniques?

  There have been many previous attempts to segment these organs. A study must be done, however, on the robustness of these methods. Typically, the segmentation methods are simple and perform adequately only in specific cases, and a large number of parameters must be set appropriately. These parameters are often difficult to derive, and must be fine-tuned for each segmentation task.

- How can we best segment the liver, hepatic vessels, possible tumours and the kidney from CT and MR scans?

  The main goal in this thesis is to derive robust methods that successfully segment these organs. It is unlikely that we can produce methods that perform well in all possible cases, and we aim therefore to find methods that perform better than the current state of the art. The proposed methods should not be highly dependent on numerous parameters and a high percentage of suitable parameter choices should yield good results.

- What statistical models can guide and verify the proposed segmentation techniques?

This question is closely related to question 2. Depending on the segmentation methods we propose, we need a robust representation that can be used to incorporate statistical data to guide the segmentation methods. There exist many previously presented models that can suit our purpose, but we must remain open-minded with respect to new solutions as well. A typical challenge with previous representations is to retain important properties that can be used to improve the segmentation, while disregarding unnecessary features to simplify the representation and make it more general.

1.2 Research goals

The primary research goal is related to the second and third research question, and consists of deriving and combining segmentation techniques that result in sound and robust solutions to the following segmentation tasks:

- Segmentation of the liver and liver tumours.

  The main challenge in liver segmentation is to separate the liver tissue from surrounding tissue. These regions generally have non-existing borders and edge-detecting methods are ineffective to separate the tissues. Statistical methods are also difficult to apply, since tumours in the liver and previous interventions may result in non-typical liver shapes. Segmentation of the tumours shares the same challenges, but the tumour and liver tissues are generally separable by their texture.

- Segmentation of blood vessel structures, in particular liver vessels.

  Segmentation of hepatic vessels is a problem that is closely related to segmentation of the liver. A liver segmentation may provide a good starting point for segmentation of these vessels. However, it is conceivable that a stand-alone vessel segmentation may be conducted. The texture of liver and vessel tissue is generally different, but appropriate CT or MR scan parameters are required.

- Segmentation of the kidney.

  This task is easier to accomplish than the previous two due to the difference between the kidney tissue and surrounding tissue in CT scans. If a suitable method is found for segmenting the liver, it is likely that the same technique can be applied to kidney segmentation with an even higher success rate.

We have defined two secondary goals as well.

- Visualisation of the segmentation results.

  Visualisation of the final result is needed for presentation, interpretation and analysis. Especially blood vessels may prove difficult to visualise efficiently, due to the need of a suitable way to compute a triangle mesh of branching structures.

- Parallelisation of the proposed methods to decrease the processing time.

  Image processing techniques are often time consuming and inefficient, especially for real time processing. Therefore, to significantly increase the processing speed, we developed parallel algorithms through the use of Graphics Processing Units (GPUs) on inexpensive modern graphic cards or through utilising the increasing number of cores in personal computers.

1.3 Research approach

I intend to answer the given research questions through the following research methods:

- Literature review (Hart, 1998; Kitchenham, 2004)
- Design science (Hevner and Ram, 2004; Vaishnavi and Kuechler, 2007)
- Controlled experiments (Wikipedia, 2007; Frigon and Mathews, 1997)

An overview of previous proposed solutions to our research goals will be achieved through literature reviews. Next, the solutions will be evaluated, and on this basis improved solutions will be derived to solve the given tasks and implemented through design science. Controlled experiments will follow, and the derived solutions will be modified and improved if needed.

As previously stated, we do not hope to find complete solutions for the tasks at hand. The goal is to advance a step further with respect to previously published material. Large scale verification of the methods will therefore not be performed. However, we will discuss the quality of the proposed methods with respect to previously published material.

1.4 Published articles

The following published conference articles are related to this PhD work.

- Eidheim, Ole Christian and Aurdal, Lars and Omholt-Jensen, Tormod and Mala, Tom and Edwin, B. (2004). Interconnecting segmented hepatic vessels in adjacent CT slices. *NOBIM*, pages 91-97.

  Eidheim, Ole Christian and Aurdal, Lars and Omholt-Jensen, Tormod and Mala, Tom and Edwin, B. (2004). Segmentation of liver vessels as seen in MR and CT images. *Computer Assisted Radiology and Surgery*, pages 201-206.

  These two articles represent early work from the beginning of the PhD period. The articles describe techniques that reconstruct hepatic vessels from CT scans through the use of anatomical knowledge. The methods work on individual slices, however, instead of applying 3D processing methods on the CT volume as presented in our later work, Eidheim (2005). The two articles can be found in Appendix A and B, respectively.

- Skjermo, Jo and Eidheim, Ole Christian (2005). Polygon Mesh Generation of Branching Structures. *14th Scandinavian Conference on Image Analysis*.

  This article presents a method for visualisation of the hepatic vessel results from the two first articles and my master thesis. The article describe a mesh producing method for visualisation of branching structures that was written in collaboration with Jo Skjermo. The article can be found in Appendix C.

- Eidheim, Ole Christian and Skjermo, Jo and Aurdal, Lars (2005). Real-time Analysis of Ultrasound Images Using GPU. *Computer Assisted Radiology and Surgery*.

  In this article, we studied the processing speed of parallelised algorithms on modern graphics cards in comparison to serial processing on a CPU. This work is presented in chapter 7.

1.5 Thesis organisation

This thesis is divided into three parts:

Part 1. Introduction and background

The first part outlines the research motivation, goals and previous work performed in the field of medical image processing. This part includes the following chapters:

- Chapter 1 contains this introduction.
- Chapter 2 outlines previous image processing methods that are related to the given segmentation tasks.

Part 2. New approaches, results and discussion

This part consists of our proposed methods to solve our research goals and to answer the given research questions. The following chapters are included in this part:

- Chapter 3 presents a 3D skeleton algorithm for retrieving more compact representations of 3-dimensional objects where the homotopy of the volume is kept.
- Chapter 4 outlines a texture-based segmentation method for creating a more robust segmentation of organs, in particular the liver, from CT and MR scans.
- Chapter 5 demonstrates the texture-based technique presented in chapter 4 on hepatic vessel, liver tumour and kidney segmentation.
- Chapter 6 presents a new general representation of objects that may not have a simple interior. The method is demonstrated used to automate the technique given in chapter 4.

Part 3. Assessment

- Chapter 7 contains the final discussion, conclusion and proposed future work.

Chapter 2

Related work in medical image segmentation

Due to the importance of medical applications, medical images have been of particular interest to the image processing community. A vast number of proposed methods have been published that process medical images to detect important structures or to do automatic measurements and diagnostics. Still, very few image processing tasks have been implemented and used in medical practice. The main reason behind this is the uncertain results of the methods and the high number of parameters that need to be appropriately set in order for the methods to work soundly. The problem of segmenting an organ from a medical modality is highly complex, especially because of diffuse boundaries between the different tissue types and the great anatomical variation from person to person.

In this chapter we will review the typical image processing tasks used in medical imaging. The focus will be on tasks that are related to the given research goals. Preprocessing tasks will be covered first, and they are typically performed prior to a segmentation algorithm to detect data to make the segmentation easier. Segmentation is the task of labeling similar regions, and the regions are typically represented by a feature vector, that is a point in a hyper-dimensional space. The classifier groups these points into separate classes and uses these groupings to classify new points.

The image processing methods in this chapter are divided into the previously mentioned approaches, however, the approaches may slightly overlap. For instance, segmentation and classification are similar operations and the latter techniques are commonly used to execute the former even though it is common to distinguish segmentation and classification. The same is true for preprocessing techniques, which sometimes can be extended to a segmentation algorithm, additionally to be used to improve a segmentation result in a

postprocessing stage. Nevertheless, we attempt to place the methods where they are most commonly associated in the literature.

After presenting the common image processing techniques, we will discuss the execution time of image processing methods with respect to being implemented and used in medical applications. Finally, we will discuss the presented techniques and their applicability to the task at hand.

2.1 Preprocessing

Preprocessing is usually applied before executing a segmentation algorithm. Generally, the objective is to make important regions more readable by the segmentation algorithms.

After the segmentation, corrections to the results are sometimes needed in a stage called postprocessing. These corrections are often performed using the same techniques that are used in the preprocessing stage.

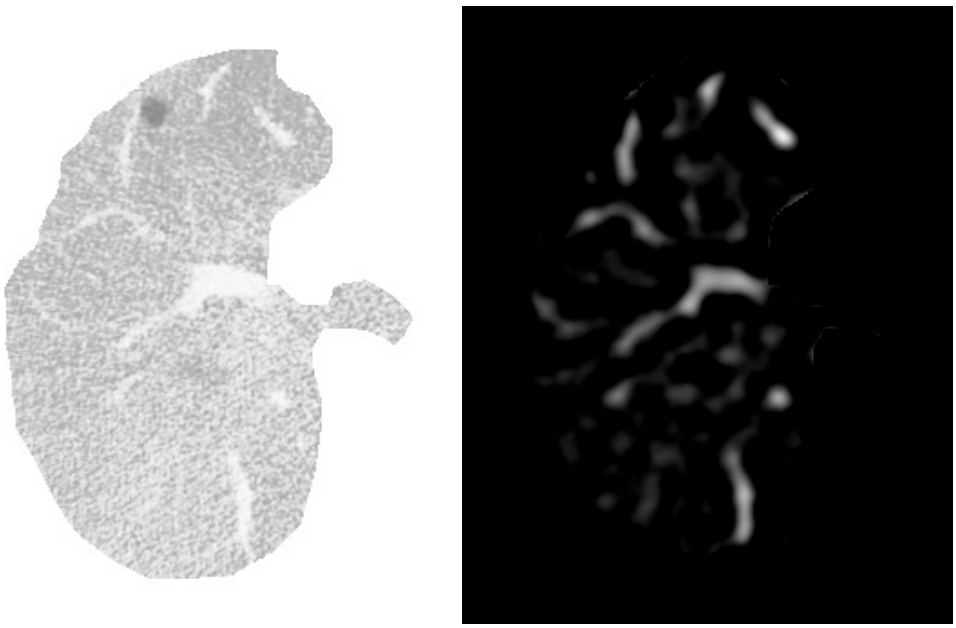
2.1.1 Spatial filters

Spatial filtering through convolution is commonly used in image preprocessing (Gonzalez and Woods, 2008; Sonka et al., 1999). A frequent application is to remove noise by smoothing or blurring an image. Another common use is called matching, which is basically finding regions in an image that correspond to a given template.

Matched filtering was used to locate hepatic vessels from CT scans in Chaudhuri et al. (1989). The idea was that vessels in CT images have a Gaussian profile, and thus a Gaussian hill filter could be used to match the blood vessels. Since blood vessels have distinct sizes and are oriented in separate directions, the filter has to be scaled and rotated, and the filtering results from all the scaled and rotated Gaussian hill templates were summed up. Omholt-Jensen (2002) also used matched filtering in segmentation of hepatic vessels, where the purpose was to use anatomical knowledge to derive an improved vessel segmentation. Figure 2.1 shows an example result of matched filtering applied to a CT slice.

2.1.2 Mathematical morphology

Mathematical morphology is a general framework that can be used to perform various image processing tasks. Two basic operators, namely dilation and erosion, form the basis of morphological methods (Gonzalez and Woods, 2008; Sonka et al., 1999; Soille, 2003). A structure element is used to define which



(a) (b)

Figure 2.1: Example matched filtering result where the blood vessels within a liver are emphasised. a) Original CT slice masked with a liver mask. b) Matched filtering result of a) using several templates matching different sized blood vessels in separate headings.

neighbouring pixels should be included during filtering. Using a flat structure element (Soille, 2003), dilation and erosion are local maxima and minima filters, respectively.

Most image processing libraries contain these two operators, making the mathematical morphology methods highly available. Some example uses are region filling, matching through the hit-or-miss transform, boundary extraction, finding object skeletons, and pruning unimportant pixels. Mathematical morphology can also be used to implement the distance transform, which is used to compute the distances to the nearest background pixels in an image.

In medical imaging, Aykac et al. (2003) make use of morphological closing and erosion in a preprocessing step to identify candidate airway locations. Further, morphological opening, closing, and skeletonisation (thinning followed by pruning) are used by Thomas et al. (1991) to measure the fetal femur length in ultrasound images.

2.1.3 Bayesian image processing

Bayesian image restoration was introduced by Geman and Geman (1984). The idea was to make use of the Bayes formula (Duda et al., 2001) to reconstruct images:

$$P(X|Y) = \frac{p(Y|X)P(X)}{P(Y)} \tag{2.1}$$

In this formula, $p(Y|X)$ is called the likelihood distribution, which specifies how X has been degraded from Y. The last part of (2.1), $P(X)$, is the prior probability that defines how neighbouring pixels are related in X. Figure 2.2 shows an example of vessel restoration that we computed while testing various preprocessing algorithms.

This technique can be used to reconstruct images in a preprocessing stage as well as for segmentation purposes. The unknown reconstructed image is then corresponding to X, and the original image is set to be Y. The pixels in the new image are updated iteratively after the likelihood distribution and the prior probability have been defined.

In Bayesian image processing, the two models, namely the likelihood distribution and the prior probability, are derived so that they define the end result. $p(Y|X)$ corresponds to the probability of Y typically based on surrounding pixels in X. The second model, $P(Y)$, defines the probability of Y with respect to prior knowledge of the dataset to be processed.

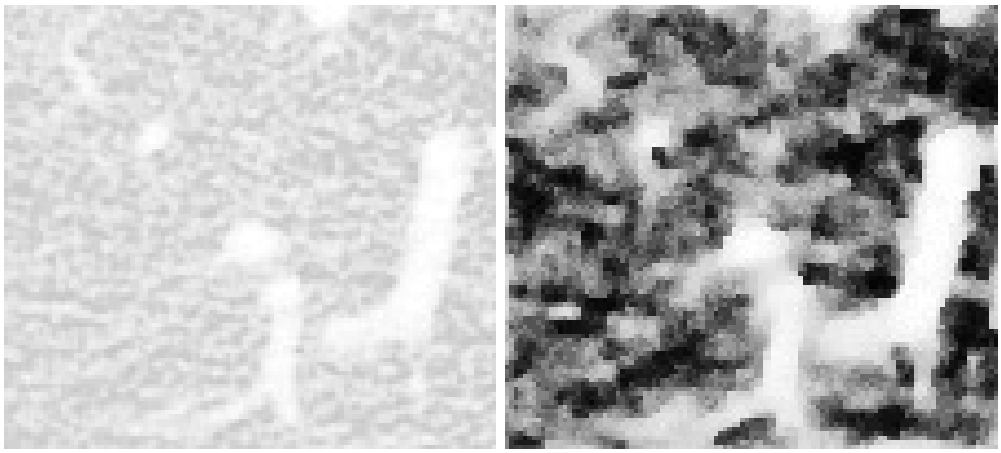
Energy minimising methods such as simulated annealing is ordinarily used to find the optimal estimation of image X. Refer to Winkler (1995); Geman and Geman (1984); Hokland (2002) for further reading.

2.1.4 Level set methods

The level set method introduced in Sethian (1997, 1996), is primarily a model based segmentation algorithm and will be more fully described in section 2.2.2. This approach, however, can also be used in noise removal as proposed by Malladi and Sethian (1996). The idea of the method is to move the intensity values of the image in the direction of the curvature as shown in Sethian (2004b). An example is preprocessing of a digital subtraction angiogram (DSA) as shown in figure 2.3.

2.1.5 Anisotropic diffusion

The use of anisotropic diffusion in image processing was introduced by Perona and Malik (1990), and the technique has since been used frequently in image



(a) (b)

Figure 2.2: Example restoration using Bayesian image analysis. a) A part of the liver. b) Result of restoration.

(a) (b)

Figure 2.3: Example usage of motion by curvature. a) A digital subtraction angiogram of an artery. b) Preprocessing result of a) using motion by curvature. Images courtesy of Sethian (2004b)

analysis. The anisotropic diffusion equation for an image I is as follows:

$$\frac{\partial I}{\partial t} = \nabla(c\nabla I) \tag{2.2}$$

where the diffusion coefficient c varies in space but not in time. If c is instead constant, the equation is reduced to the isotropic heat diffusion equation:

$$\frac{\partial I}{\partial t} = c\nabla^2 I \tag{2.3}$$

Applying isotropic heat diffusion to an image is equivalent to running a Gaussian filter on the image. By using anisotropic diffusion with varying c, however, it is possible to specify the magnitude of blurring with respect to the contents of an image.

Typically during image preprocessing, we want to blur roughly homogeneous regions, while preserving the edges. This can for instance be achieved in anisotropic diffusion by setting $c = g(||\nabla I(x, y, t)||)$, and thus vary c with respect to the edges in an image. Additionally, if the function g is chosen appropriately, the edges in an image can be sharpened as well (Perona and Malik, 1990).

There are several example uses of anisotropic diffusion in medical imaging. For instance, Soler et al. (2001) uses anisotropic diffusion in a preprocessing phase before segmentation of the liver, blood vessels, and possible liver tumours from CT scans. Another example is Chung and Sapiro (2000), where anisotropic diffusion is used before segmenting skin lesions.

Anisotropic diffusion is also used in the gradient vector flow procedure (Xu and Prince, 1998, 2000), which is typically used before applying a deformable model segmentation (See section 2.2.2).

2.2 Segmentation

Image segmentation refers to the process of finding regions in an image that have one or more common properties. Example common properties are colour, texture and shape. The segmentation result is typically a binary image where the regions are represented by value one and the background by value zero. The next step after the segmentation is to extract a meaningful representation such that a classifier can distinguish the separate regions.

Generally, we group segmentation techniques into two main groups, local and global methods. Local approaches are exclusively based on information contained in the image itself. The image is assumed to be "self-contained", i.e. it has all the information necessary to retrieve the objects of interest. On the other hand, global methods utilise additional related knowledge about the image in the segmentation approach such as statistical templates or physical models.

2.2.1 Local approaches

Local approaches are often simpler than global approaches. However, there are exceptions like mean shift segmentation where data in the image is used in a classifier to segment regions in the image. There is much information in an image that can be exploited by a segmentation algorithm, and better results are not necessarily achieved by using global methods. One theoretical example is when the statistical variance is too high to make a suitable model of the data to be segmented. As common elsewhere in image processing, local and global approaches can be combined to solve a segmentation problem.

Thresholding

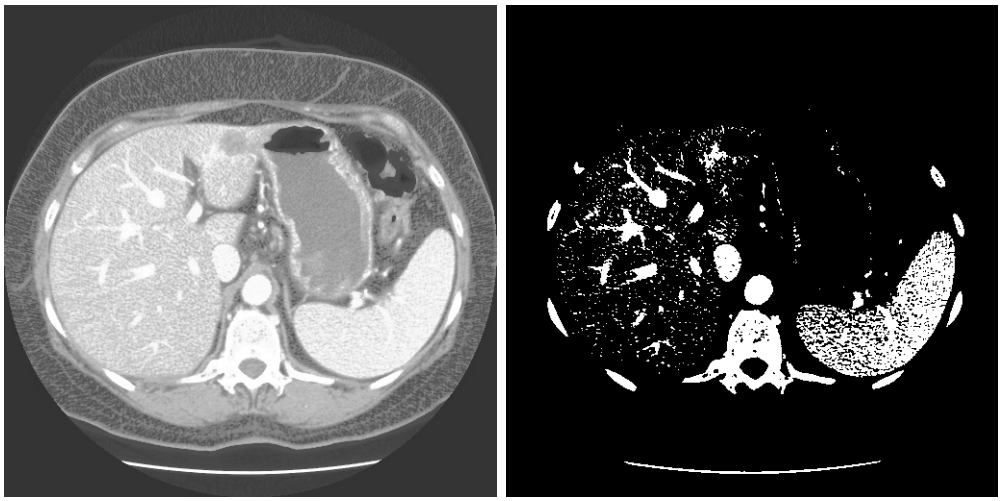
Thresholding is the simplest and most frequently used segmentation algorithm. The basic idea is to mark pixels having intensity values within a predetermined range (Gonzalez and Woods, 2008; Sonka et al., 1999). A slightly more advanced usage of this algorithm is described in Székely and Gerig (2000). Here, a two dimensional intensity distribution from a spin-echo MR image pair is computed, and the two distributions are used to segment the tissues. A more accurate segmentation is usually accomplished using more than one spectrum of the same scene, such as for instance colour images.

The main challenge concerning thresholding is to select the most desirable intensity range. One way to solve this is through an approach called optimal thresholding (Gonzalez and Woods, 2008; Sonka et al., 1999). The optimal threshold is said to be the threshold that causes the smallest number of pixels to be incorrectly segmented. In optimal thresholding, Gaussian curves are fitted to the histogram of an image, and thresholds are set where the curves cross. Soler et al. (2001) makes use of this method in segmenting specific tissues in CT images.

Another significant thresholding algorithm is based on the entropy of an image's histogram (Kapur et al., 1985). An entropy diagram is obtained by an average entropy measure (Gonzalez and Woods, 2008), and each local maximum on the entropy diagram represents a potential best threshold. This method is reported to be successful in segmenting liver vessels in Glombitza et al. (1999) and Omholt-Jensen (2002). See figure 2.4 for an example use of entropy based thresholding.

Mean shift segmentation

Mean shift segmentation is an interesting, recently proposed segmentation technique. Here, pixels in an image are instead represented as points in



(a) (b)

Figure 2.4: Example segmentation using thresholding based on entropy. a) A CT image. b) Resulting segmentation of a) using thresholding based on entropy.

a feature space. Density estimation in the feature space is processed using the Parzen window technique described in Comaniciu and Meer (2002); Duda et al. (2001), and a mean shift procedure is used to follow the density gradient in the feature space to a local maximum (Comaniciu and Meer, 2002).

These maxima represents segmentation identities, and all pixels leading to the same local maximum is grouped into the same segment. The number of maxima is dependent on the Parzen window size, and thus the number of segments does not need to be known beforehand.

Although not mentioned in section 2.1, the mean shift procedure can be used in image preprocessing as well (Comaniciu and Meer, 2002; Fernández et al., 2003).

Edge based segmentation

Edge based segmentation involves following edges in an image and mark pixels within closed boundaries. The simplest approach is to analyse the immediate neighbourhood of each pixel in an image and label pixels having similar gradient magnitude and direction (Gonzalez and Woods, 2008). Closed contours, containing equally labelled pixels, are eventually filled and transformed into segmented regions.

More advanced examples of local edge detectors are the Canny edge de-

tector and the Marr-Hildreth algorithm (Gonzalez and Woods, 2008). The Canny edge detector searches for local directional maxima in the gradient magnitude, while the Marr-Hildreth algorithm detect edges by locating zero-crossings in the result from a second order differential operator such as the Laplacian.

The procedures above, however, considers only adjacent pixels when tracking the edges in an image. Instead, graph theoretic techniques (Gonzalez and Woods, 2008; Sonka et al., 1999) can be utilised to find the least expensive path according to a given cost function. Dynamic programming (Bellman, 1957; Sonka et al., 1999) are typically used to find the global minimum, and thereby the most appropriate edge graph from a given starting point.

Another example of an edge based segmentation algorithm is the Hough transform (Hough, 1959; Gonzalez and Woods, 2008). Using various models, it is possible to search an image for simple geometric shapes. For instance, to search an image for circles with radius 1 we calculate the following Hough transform:

$$H(a,b) = \int\int f(x,y)\delta((x-a)^2 + (y-b)^2 - 1)dxdy \tag{2.4}$$

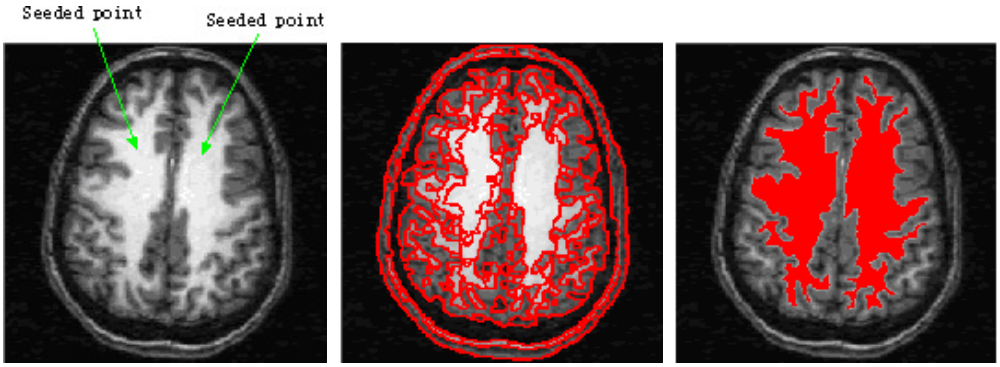
where $f(x,y)$ represents the gradient of an image, and (a,b) corresponds to the positions of the sought circles. The commonly used delta function, $\delta(u)$, returns 1 when u is 0, and 0 otherwise. High values in $H(a,b)$ represents positions (a,b) where circles are found in $f(x,y)$.

An elliptical Hough transform was used to identify axon centres in Fok et al. (1996). The purpose of this paper was to count the number of axons in nerve cells as well as extract each axon's size and shape. After the initial identification of the axon centres, an active contour model (McInerney and Terzopoulos, 2000) was used to refine the axon contours.

Region based segmentation

Region based procedures rely on common properties between adjacent pixels (Gonzalez and Woods, 2008; Sonka et al., 1999). Starting with a few seed points, regions are typically expanded until a property criteria is no longer met or until a region boundary collides with another region boundary. A similar method is watershed segmentation (Vincent and Soille, 1991) where regions are grown from local intensity minima of an image. An example watershed segmentation is shown in figure 2.5.

Example applications of region growing can be found in Martinez-Perez et al. (1999) and Tuduki et al. (2000) where blood vessels were segmented.



(a) (b) (c)

Figure 2.5: Segmentation of the brain white matter. a) Seed points. b) Resulting over-segmentation using watershed. c) A level set method, which will be described in section 2.2.2, is used to make the final segmentation using the watershed results from b). Images courtesy of MIPG medical image processing group.

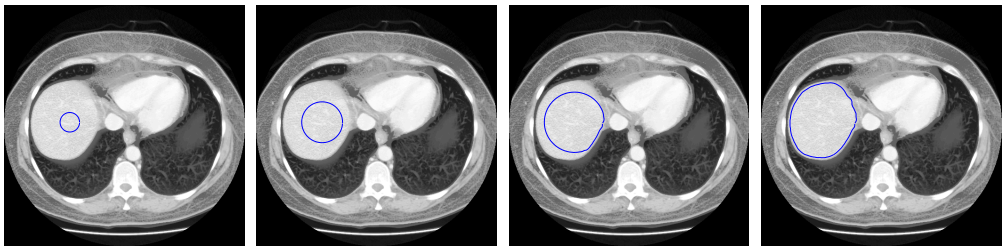
In (Krivanek and Sonka, 1998), watershed segmentation was used to automatically measure the size and shape of follicles from ultrasound images. Furthermore, watershed segmentation was used to segment the coronary arterial tree in Haris et al. (1999).

Additionally, several articles propose region based methods for delineating the liver vessels automatically through region based segmentation algorithms (Chaudhuri et al., 1989; Kapur et al., 1985; Inaoka et al., 1992; Zahlten et al., 1995; Soler et al., 2001). Most promising, with respect to creating a 3D model of the liver vessel structure, are the methods by Zalthen et al. and Soler et al.

Zalthen et al. use a 3D region growing algorithm to extract the portal vein, but a seed point must be given. In the article by Soler et al., the portal trunk is located using its general anatomical position. The portal vein skeleton is calculated using skeletonising methods from Bertrand and Malandain (1994); Malandain et al. (1993) and is corrected by pruning vessel segments that do not confirm with a set of predetermined properties.

2.2.2 Global approaches

Global methods utilise additional knowledge concerning the problem at hand. Physical models, for instance, make use of the knowledge of the general shape of the objects to segment. Another example is statistical models that represent the statistical variations of the object to be segmented. These



(a) (b) (c) (d)

Figure 2.6: a) Initialisation of the contour model, in blue, on a CT image. b), c), d) The contour alters shape to fit the liver .

methods are typically applied when segmentation is problematic based on data exclusively from the image.

Deformable models

The first article on this topic was Kass et al. (1988). Here, a model called *the snake model* was introduced. In short, a snake is a spline influenced by internal and external forces seeking an energy minimum. Internal forces typically control the tension and rigidity of the snake, and external forces draw the snake towards edges in an image (McInerney and Terzopoulos, 2000; Heuch, 2003). External forces from the image $I(x, y)$ are usually derived from:

$$P(x, y) = -c|\nabla[G_\sigma * I(x, y)]| \tag{2.5}$$

where c is the magnitude of the force, ∇ is the gradient, and G_σ is a Gaussian smoothing filter. $G_\sigma * I(x, y)$ means that $I(x, y)$ is convolved with G_σ. External forces can also include the so-called balloon force that expands the snake to find far edges. Another way of pulling the snake to remote edges is to implement gradient vector flow (Xu and Prince, 1998, 2000) that was briefly mentioned in section 2.1.5.

An interesting alternative to the original snake model is the discrete dynamic contour model (Lobregt and Viergever, 1995). The structure of this model is a set of interconnected vertices that is transformed directly through simple vector operations. Figure 2.6 shows an example segmentation of a liver slice using a deformable model.

In Montagnat and Delingette (1997), a hybrid model was described that is similar to the snake model. This model, however, is restricted to a reference model that to a high degree define the end result. The model was used to segment the liver and brain ventricles from CT volumes.

A similar method that is worth mentioning is described in Tsagaan et al. (2002). Here, NURBS (Non-Uniform Rational B-Spline) surfaces are used to segment the kidney guided by statistical information. The representation is of particular interest since it may be easier to represent varying surfaces using NURBS surfaces than attempting to locate corresponding points that are evenly spread on two surfaces.

An extended use of the deformable model was also demonstrated in Székely and Gerig (2000), where intensity profiles along the boundary were used to guide the segmentation. Such incorporation of gray-level appearance of the anatomy in combination with a shape model is known as active appearance models (Cootes et al., 1998).

Snakes are becoming more and more common in medical imaging. For instance, Heuch (2003) used snakes to segment the liver from CT images, and Kelemen et al. (1998); Kelemen and Székely (1999); Székely and Gerig (2000) utilised snakes to segment the basal ganglia of the human brain. Moreover, snakes with a modified gradient vector flow were used to track white blood cells in Ray et al. (2002).

Level set method

Deformable models are difficult to handle when the topology of the contour changes (Sethian, 1997). The level set method (Sethian, 1997, 1996) solves this issue by working with shapes one dimension higher than the dataset to segment. For instance, if a 2D image is being segmented, a 3D shape is deformed to match the objects of the image. Intersections of the shape and the image, also known as the zero level set, are regarded as the contours. The advantage of level set segmentation is that splitting and merging of contours happen automatically with no additional processing.

Level set methods have been demonstrated to be useful in outlining the stomach from CT images, and in segmenting the structures of arterial trees from DSA images (Malladi et al., 1995; Malladi and Sethian, 1996). Examples of beating heart segmentation, femurs and surrounding soft tissue segmentation, and brain reconstruction can also be found in Sethian (2004a).

Statistical models

Statistical models represent the statistical variation of a sought object. Statistical models are typically more resource demanding to implement since they commonly require hand segmented samples. Many segmentation tasks, however, require statistical models in order to be adequately solved.

In applications, statistical models have two common uses. First, they can

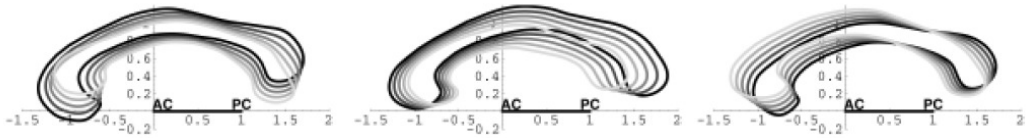


Figure 2.7: Statistical models built by principal component analysis. The figures show the models reconstructed from the three eigenvectors with the highest corresponding eigenvalues. Images courtesy of Székely and Gerig (2000).

be used as a segmentation initialisation by matching reconstructed models onto structures in an image. After this, the segmentation can be refined using another segmentation method, for instance active contours. The second use of statistical models are in criteria functions to compute the statistical validity of an already computed segmentation result.

Székely and Gerig (2000); Kelemen et al. (1998); Kelemen and Székely (1999) use principal component analysis (Jolliffe, 2002) to create a smaller set of statistical templates, eigenmodes, expressing the main variations of a larger training set. A best match is found and refined using a deformable model. In (Sclaroff and Liu, 2001), a multidimensional unimodal Gaussian distribution of the training set is assumed. Deviation from the mean is penalised by an amount that is proportional to the Gaussian distribution function. More example uses of statistical models are Székely and Gerig (2000); Kelemen et al. (1998); Kelemen and Székely (1999) where 2D and 3D models of the corpus callosum are used (see figure 2.7). In Soler et al. (2001), a statistical model of lesions were defined to locate tumours in the liver. Furthermore, statistical models were used to automatically segment the three main structures of the heart from MR scans in Frangi et al. (2002).

In section 2.1.3, we described methods that could be used to restore and improve images using Bayesian decision theory. Similar methods are proposed in medical image segmentation (Choi et al., 1991). First, a manual segmentation and classification is conducted, and Gaussian curves are fitted to the probability distributions for each class. In addition to the derived mean and variance, a model of the regions known as the prior probability is used. For each pixel, an a posteriori probability is calculated for each class, and an appropriate class is selected typically through inversion (Ripley, 1987).

Case-based reasoning

Case-based reasoning, related to using statistical models, solves new problems based on previous similar problems. For purposes of computer reasoning, the process has been formalised as a four-step process (Aamodt and Plaza, 1994):

1. Retrieve: Given a target problem, retrieve cases from memory that are relevant to solving it. A case consists of a problem, its solution, and, typically, annotations about how the solution was derived. For example, suppose Fred wants to prepare blueberry pancakes. Being a novice cook, the most relevant experience he can recall is one in which he successfully made plain pancakes. The procedure he followed for making the plain pancakes, together with justifications for decisions made along the way, constitutes Fred's retrieved case.

2. Reuse: Map the solution from the previous case to the target problem. This may involve adapting the solution as needed to fit the new situation. In the pancake example, Fred must adapt his retrieved solution to include the addition of blueberries.

3. Revise: Having mapped the previous solution to the target situation, test the new solution in the real world (or a simulation) and, if necessary, revise. Suppose Fred adapted his pancake solution by adding blueberries to the batter. After mixing, he discovers that the batter has turned blue - an undesired effect. This suggests the following revision: delay the addition of blueberries until after the batter has been ladled into the pan.

4. Retain: After the solution has been successfully adapted to the target problem, store the resulting experience as a new case in memory. Fred, accordingly, records his newfound procedure for making blueberry pancakes, thereby enriching his set of stored experiences, and better preparing him for future pancake-making demands.

Case-based reasoning has been proposed used in image processing as well. For instance Perner (1999) uses case-based reasoning (CBR) to automatically find the brain/liquid ratio from CT scans, and M. Frucci and di Baja (2008) use CBR to automatically find the most suitable parameters for watershed segmentation based on previous similar problems.

2.3 Object representation and description

Prior to making a classification of an object, the object is typically represented as a numeric feature vector. A point in an n-dimensional space is then representing the shape, and classification is achieved by partitioning the space into separate classifications. Other ways to represent knowledge exist, such as predicative logic, structural descriptions, production rules and semantic nets. Most generic classifiers, however, have a numerical feature vector as input.

The feature vector is a more compact representation of an object. Important elements that can help separate different objects are kept, while unneeded properties are disregarded.

Representations has other uses in image processing as well. One example is object skeletons that have been used to identify blood vessel centres and branches.

2.3.1 Contour based representation and description

There are several simple features that can be used to describe the boundary of an object. Some examples are chain codes, boundary length, curvature, bending energy, signature, and chord distribution (Sonka et al., 1999; Gonzalez and Woods, 2008). Fourier descriptors represent a slightly more complex method where frequencies along one direction of a contour are obtained instead of using the contour positions directly. Consequently, it is straightforward to disregard high frequencies and thereby compress the representation depending on the needed level of detail (Gonzalez and Woods, 2008).

There are, however, two main problems with contour based descriptors. First, every similar shape must have a corresponding starting point of the contour if they are to be classified equally. There exist no general way to find such a starting point, and one has to choose a normalisation technique depending on the problem at hand. The second problem is finding corresponding points along the contours of dissimilar shapes.

2.3.2 Shape based representation and description

When describing an object it is important that the description is invariant to especially affine transformations. Shape descriptors are in general invariant to these transformations, making them typically more useful than boundary descriptors for recognition purposes.

Some simple example shape based descriptors are area, Euler's number,

projections, eccentricity, elongatedness, rectangularity, shape direction, compactness, and convex hull (Sonka et al., 1999).

Statistical characteristics

Statistical moments are an important shape based descriptor (Papoulis, 1991). Here, the image function is interpreted as a probability density of a 2D random variable. Properties of this probability density can be described using statistical moments. Different degrees of invariance can be achieved depending on which moments are used. Scaled central moments, for instance, are translation and scale invariant.

Other statistical characteristics also exists such as maximal probability, element difference moment of order k, inverse element difference moment of order k, uniformity and entropy (Gonzalez and Woods, 2008).

2.3.3 Texture based description

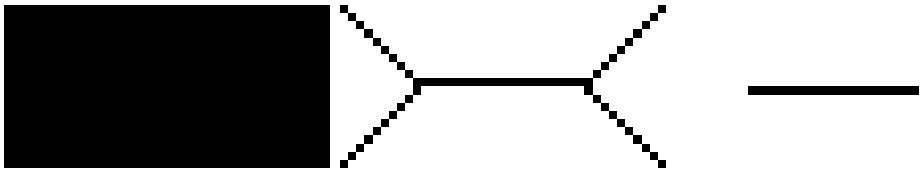
Texture may contain valuable information that can help the classifier to separate objects of different classes. This is especially true with regards to volumes from medical modalities where texture is the main source of information.

A simple way to represent texture is through the region's histogram, but this representation may not be adequate for classification purposes (Bevk and Kononenko, 2002). A more useful representation is therefore the co-occurrence matrix (Gonzalez and Woods, 2008), where pixel setup occurrences constitute a second order distribution. Commonly, statistical characteristics are computed from the distribution to define a more compressed feature vector.

Texture based descriptions will be discussed further in chapter 4.

2.3.4 Shape skeleton

A shape skeleton represents a compact representation of an object without changing its topology. In 2D, the skeleton is typically defined as the medial axis of a shape. Blum (1967) proposed a medial axis transform, where each pixel in a shape is marked as a medial axis if the pixels have two or more smallest distances to the background. An extension of this definition to 3D is possible, however, surfaces instead of lines may then be characterised as medial axes. It is therefore common to distinguish 3D skeletons into medial surfaces and medial lines.



(a) (b) (c)

Figure 2.8: The skeleton of a rectangle using two different algorithms. a) The rectangle to be thinned. b) Skeleton of a) using morphological thinning. c) Skeleton of a) using the algorithm given in Guo and Hall (1989).

2D skeletons

We briefly mentioned skeletonisation through morphological thinning in subsection 2.1.2. In morphological thinning, hit-and-miss templates are first used to find shape boundaries (Soille, 2003). The boundaries are removed successively until only the skeleton of the shape remains. Post-processing of the skeleton may consist of a pruning phase where small end branches of the skeleton are removed.

Another example of an iterative skeleton algorithm is described in Guo and Hall (1989). This algorithm produces a more compact skeleton that in most cases does not need pruning. Refer to Lam et al. (1992) for further reading on additional iterative 2D skeleton algorithms.

Non-iterative skeleton algorithms have also been proposed. These are typically based on medial axis and distance transforms (for instance Blum (1967)), or line following and run length encoding (Lam et al., 1992).

Figure 2.8 shows a rectangle and its two resulting skeletons using thinning by morphology and the skeleton algorithm given in Guo and Hall (1989).

3D skeletons

Existing templates or criteria functions used in most 2D skeleton algorithms cannot be directly extended to 3D. However, much research effort has been put into 3D skeletons recently due to the increased availability of 3D imaging data. Lobreget et al. (1980) first presented a skeleton algorithm based on preservation of Euler characteristics. Further, Ma and Sonka (1996) proposed a boundary thinning algorithm utilising deleting templates, and Saha et al. (1997) developed an algorithm that preserves the number of object components, cavities, and tunnels. More recent example publications of 3D thinning algorithms can be found in Borgefors et al. (1999); Palagyi et al. (2001); Xie et al. (2003).

Palagyi et al. (2001); Xie et al. (2003) both make use of a simple point definition derived in Malandain and Bertrand (1992). A point is simple if its removal does not affect the topology of the shape. The thinning result also depends on the deletion order of the simple points. Palagyi et al. (2001), for instance, propose a method that mark all voxels for deletion in a separate pass before deleting them. This procedure is repeated for each heading; northern boundary voxels are deleted first, then southern, and so forth. Nevertheless, testing has shown that current simple point definitions are erroneous in certain voxel setups and the deletion order must be improved to compute a more compact skeleton with fewer unnecessary or faulty branches. Figure 2.9 shows an example medial surface and medial line of a cuboid. The medial line is produced using the simple point definition in Malandain and Bertrand (1992) and by thinning boundary voxels a predetermined number of iterations.

Skeletonisation through the distance transform has also been proposed in numerous articles such as Niblack et al. (1992); Zhou and Toga (1999); Sato et al. (2000); Jiang and Alperin (2004). Local maxima are here located from the distance transform, and these maxima are interconnected to form an object skeleton. The interconnection of the local maxima, however, is difficult and highly parameter controlled. To compute a minimised skeleton such as is done in the 2D algorithm given in Guo and Hall (1989), skeletonisation through thinning may most likely produce the best result (Palágyi and Kuba, 1999).

2D and 3D skeletons in medical applications

2D and 3D skeletons have been used extensively in medical image analysis. In Yim et al. (2000), vessel skeletons were used to analyse vessel paths and branching patterns of vascular trees from magnetic resonance angiography (MRA). Similar operations were performed in Palagyi et al. (2001), but from Spiral Computed Tomography (S-CT) volumes. More recently, Volkau et al. (2005) uses 3D shape skeletons to construct the human normal cerebral arterial system from various 3D datasets. Other examples include Nyström and Smedby (2001); Tom et al. (1994); Gomberg et al. (2000).

2.3.5 Principal component analysis

The use of principal component analysis to create a small model template base from a larger model database was mentioned in section 2.2.2. Samples are represented as points in an N-dimensional space, and eigenmodes are

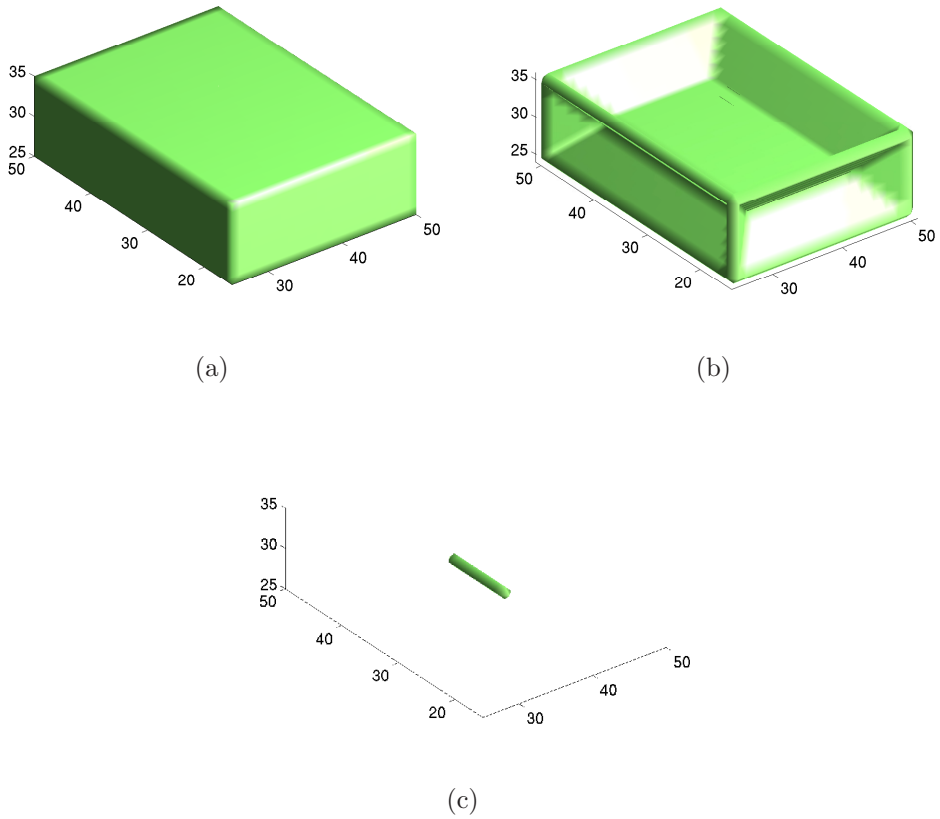


Figure 2.9: This figure shows an example medial surface and medial line of a cuboid. a) The cuboid before 3D thinning. b) Medial surface of a). c) Medial line of a) using the simple point definition described in Malandain and Bertrand (1992).

created using the eigenvectors with the largest corresponding eigenvalues (Székely and Gerig, 2000; Kelemen et al., 1998; Kelemen and Székely, 1999).

Principal component analysis is similarly used to reduce the number of dimensions of a dataset and making a more general and compressed representation of the data (Jolliffe, 2002). Instead of computing eigenmodes, the points are projected onto an N-dimensional hyperplane defined by the eigenvectors with largest eigenvalues.

2.4 Classification

After a meaningful description is made, a classifier typically groups the data into clusters. The number of classes may or may not be predetermined, and

the classifier may be guided by known classifications or operate unguided. There are many types of classifiers and we will divide them into two groups, namely feature vector and structural classifiers. Feature vector classifiers separate data represented as points in a hyperspace into a number of groups based on their positions. An example feature vector classifier is neural networks, and its input neurons define the number of dimensions of the hyperspace. Structural classifiers work on more complex data structures that also include descriptions of relationships within the input data. These descriptions cannot be represented simply as points in an N-dimensional hyperspace, and other comparison criteria of the data is therefore needed (Duda et al., 2001).

There are three major paradigms for training the classifier (Russell and Norvig, 2002). In supervised learning, a set of already classified samples is used to modify the classifier such that it classifies equivalent data similarly. Unsupervised learning consists of clustering similar feature sets automatically into a known or unknown number of classes. Finally, in reinforced learning the classes are usually unknown, but the action of the classifier is altered over time to minimise a global cost function.

2.4.1 Feature vector classifiers

Subdividing the feature space into regions of classes can be challenging to perform properly since the data typically needs to be divided nonlinearly. One possible solution is to use a kernel function that transforms the data through a nonlinear function, and then separates the feature points linearly such as in support vector machines (Shawe-Taylor and Cristianini, 2000).

Neural networks represent a set of classifiers that directly divide the feature space nonlinearly depending on the activation functions. Artificial neurons, that is simple functions, are interconnected with input and output connections. By combining a number of such simple functions, a more complex function is realized. There are several types of artificial neural networks and the most well known are feed-forward neural networks, recurrent networks, stochastic neural networks, and modular neural networks (Wikipedia, 2007).

Another possibility is to take advantage of parameter estimation techniques if the distribution of the data is known (Duda et al., 2001). After the distribution of the training samples are made, Bayesian decision theory can be applied to classify new samples. Non-parametric techniques like the Parzen window method can be used to approximate the distribution if the distribution of the data is unknown (Duda et al., 2001). The Parzen window technique approximates the unknown distribution typically by adding points as Gaussian functions where the width is varied according to the number of

training samples. Depending on the implementation of the Parzen window method, it requires more computing power or memory usage than methods that derive the grouping of samples through hyperplanes. This drawback may pose a problem, but the classification result may be easier to predict and analyse. Moreover, samples that do not correspond to any previous training samples are classified as uncertain instead of being given an incorrect classification as a result of complex hyperplanes dividing the feature space.

Examples of simpler feature vector classifiers are minimum distance, k-nearest neighbourhood, Fisher's linear discriminant, k-means algorithm, hierarchical clustering, clustering by criterion functions, and classification and regression trees (CART) (Duda et al., 2001).

2.4.2 Structural classifiers

Structural classifiers are more distinct than the feature vector classifiers due to the various ways the objects are described. In graph matching (Sonka et al., 1999) for instance, an object is initially divided into interconnected nodes. The nodes and their interconnections may have individual properties, and the graph is typically matched to known graph templates to classify the new graph. The match is performed by finding the global optimum of a cost function, but the execution of graph matching can be time-consuming. There exist, however, examples of non-deterministic algorithms finding sub-optimal solutions that run in polynomial time.

Another example of structural classifiers is recognition with strings. Here, the properties of an object are represented as a string with variable length, and two objects can be compared by finding the edit distance (Duda et al., 2001) between their two respective strings.

2.5 Execution time

CT and MR scans are not necessarily analysed by radiologists and surgeons immediately after they are taken. This leaves some time for computational processing before the results have to be presented for human interpretation. On the other hand, modalities such as ultrasound presents a real-time video stream that makes the possibilities for data processing limited.

With respect to classification, processing of data volumes can be time-consuming. Many algorithms, however, can be parallelised through the use of certain hardware, computer clusters and multiple processor cores. An example hardware well suited for parallelising is modern graphics cards, which are

inexpensive and constantly improved due to the competitive gaming industry. Although this technology was not made for signal processing, improvements are planned that will make this hardware even more applicable to image processing and similar applications.

Parallel processors called multiple core processors are getting increasingly more common as well. CPUs are less restrictive than specialised hardware and may thus represent a cost effective way to improve the processing speed through multi-threaded programming. The number of cores on typical processors are still few, but the trend is moving towards a large increase of parallel cores on common and inexpensive hardware in the near future. For instance, Intel has already made a 80-core processor prototype that they plan to release by 2012 (Intel, 2007).

Computer clusters, on the other hand, are software implementations of parallelisation where tasks are spread over a network of computers. Many limitations of the hardware are overcome by adding more nodes to the cluster network, making this solution highly flexible. Additionally with the increasing performance of international communication networks, computer clusters may be situated virtually anywhere and specialised parallel hardware is therefore not needed on the end user side. In theory, this method is the most inexpensive and practical solution for most computationally expensive tasks that can be parallelised. There are still, however, limitations on the availability of such networks, performance issues with respect to the degree data needs to be shared between the cluster nodes, and transfer speeds between the end users and the computer cluster.

A significant disadvantage in parallelised computing is that it requires a higher skill in software engineering to develop such software than typical serial software. Currently, very few software programmers have the required skills to write decent parallel software. Optimisation of the software is also an issue since it is dependent on the hardware and the tasks must be set up to compute and communicate optimally with respect to the runtime of the application.

The choice of a parallel solution depends on the problem at hand, financial budget and available systems. These solutions are becoming increasingly available, however, and there are significant advantages in exploiting parallel processing.

2.6 Discussion

In this chapter, we have presented techniques that are typically used in medical image processing. With respect to the tasks at hand, however, none of the

techniques achieve acceptable and sufficiently robust results. The segmentation algorithms represent unreliable shortcuts to finding desired structures in an image. It is conceivable, however, that these techniques have been developed simply because the computational power needed for more advanced solutions is only now becoming available through parallel computing.

There is a significant increase of available computation power due to the growing number of cores of inexpensive processors. This opens up opportunities to perform more advanced calculations than before within a predetermined time limit. There are limitations, however, in memory and disk bandwidth, but if one can divide the computation tasks appropriately, these limitations can be overcome by utilising a computer cluster.

Another challenge lies in representing objects in a sufficient way such that important features are kept while disregarding unnecessary information. Especially 3D skeleton representations need to be improved since existing methods either have artifacts or are dependent on user-defined parameters. Moreover, most of the published representations disregard the interior of objects, and represent only the outer shape. This may pose a problem when many of the important features are located within an object.

There exists a sound foundation in all of the presented feature space classification algorithms. They also scale well with the increasingly available computational power. The challenge lies therefore in representing the data in such a way that the classifier can be utilised in an acceptable period of time and still produce reliable results. This is accomplished by extracting meaningful features that are invariant to unnecessary dimensions and thereby reducing the number for input arguments of the classifier.

Chapter 3

New skeleton method for graph based segmentation of hepatic vessels

The liver is a vital organ with vascular, metabolic, secretory, and excretory functions. It is extensively perfused and during liver surgery special care has to be taken in order to avoid bleedings.

Prior to liver surgery, the patient will typically be examined using CT scans, in particular the position of large hepatic vessels must be determined. The relative position of, for instance, tumours to these vessels is of great importance when planning the procedure and in evaluating the operability of the patient.

Surgeons and radiologists will typically base their evaluation on a visual inspection of the 2D slices produced by a CT scan. It is difficult, however, to deduce a detailed liver vessel structures from such images. Surgeons at the Intervention Centre at Rikshospitalet have found 3D renderings of the liver and its internal vessel structure to be a valuable aid in this complex evaluation phase. Currently, these renderings are based on a largely manual segmentation of the liver vessels. This procedure is time consuming and error prone, and we have sought a way to extract the liver vessel structure automatically from CT scans.

Several articles propose methods to delineate the liver vessels automatically (Chaudhuri et al., 1989; Kapur et al., 1985; Inaoka et al., 1992; Zahlten et al., 1995; Soler et al., 2001). Most promising, with respect to creating a 3D model of the liver vessel structure, are the methods by Zahlten et al. (1995); Soler et al. (2001).

Zahlten et al. (1995) use a voxel based region-growing-algorithm to extract the portal vein starting at a given seed point. A similar method is given

in Soler et al. (2001), but the portal trunk is located using its general anatomical position. The portal vein skeleton is calculated utilising methods from Bertrand and Malandain (1994); Malandain et al. (1993), and is corrected by pruning vessel segments that do not conform with a set of predetermined properties.

Graph based approaches to segment hepatic vessels guided by anatomical knowledge to search for the most likely vessel graphs were discussed in Omholt-Jensen (2002); Eidheim et al. (2004a,b). These are mainly 2D approaches, and a more promising method was presented in Eidheim (2005) where the procedures work on the whole CT volume in three dimensions.

The results need to be improved, however, mainly due to the lack of a good 3D skeletonisation algorithm, which typically forms the basis of a graph based method for vessel segmentation. Skeleton algorithms are then used to find the centres of elongated objects to locate vessel branches and centres. Although there exist several sound 2D skeleton algorithms, there has not previously been derived a general skeleton algorithm that works satisfactory in higher dimensions. The motivation for this work was therefore to derive such a 3D skeleton algorithm.

Relevant to this work, a mesh generation method for visualisation of a finished vessel graph, in particular branching structures, was derived in collaboration with Jo Skjermo (Skjermo and Eidheim, 2005).

3.1 Previous work

The derivation of 3D skeletons have been the subject of many studies in the field of image analysis, and a short overview was given in section 2.3.4. Many articles propose methods to find and delete simple points, that is voxels in a volume that can be removed without affecting the topology of the volume. An additional challenge is to select an appropriate order in which simple points are to be removed.

A classification of simple points, that is voxels that can be removed without altering the homotopy of an object, was given in Malandain and Bertrand (1992). This simple feature vector classifier has been used in many other articles such as in Palagyi et al. (2001) to create object skeletons in medical applications. During testing of the simple point definition, it was discovered that it had fundamental flaws under certain conditions. Figure 3.1 shows a voxel setup where the voxel in question is marked as not-simple. The reason is that the number of connected background voxels becomes higher than 1 since only the 18-neighbourhood is tested on background voxels. It also became clear that the deletion order given in Palagyi et al. (2001) is not

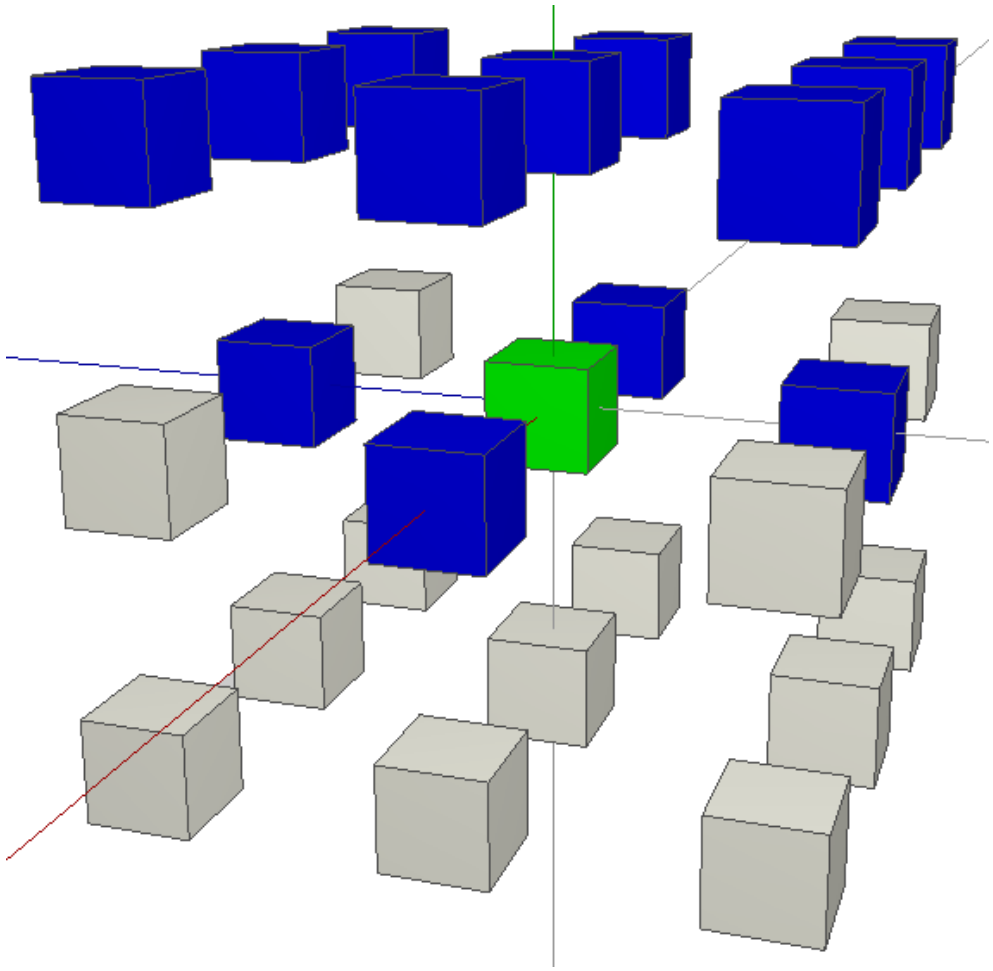


Figure 3.1: Example voxel setup shown with coloured foreground voxels, where the algorithm from Malandain and Bertrand (1992) fails to remove the centre voxel (shown in green).

sufficient to create a compact skeleton. An example of this is shown in figure 3.2.

Our goal was then to produce a more minimal 3D skeleton with as few branches as possible, much like the result from the 2D skeleton algorithm given in Guo and Hall (1989). Additionally, the algorithm should not be parameter controlled such as is typical in the distance transform based skeleton methods.

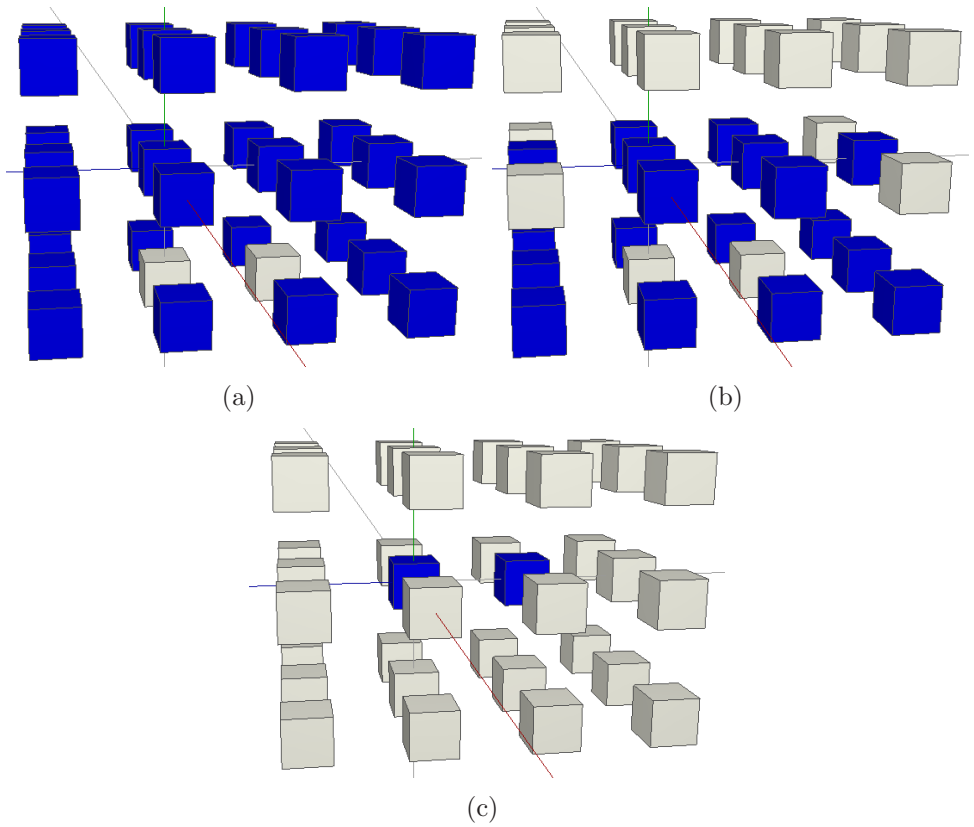


Figure 3.2: This figure shows that the deletion order given in Palagyi et al. (2001) is not sufficient to produce a sufficiently compact skeleton. a) Original object to be skeletonised. b) Non-compact skeleton produced by the algorithm given in Palagyi et al. (2001). c) A more compact skeleton (produced by the algorithm given in subsection 3.2).

3.2 Method

We begin by defining two terms that are essential in the algorithm, namely *simple points* and *end points*. The following definitions hold for any connectivity and dimensionality of the dataset.

Simple points are voxels that are crucial for retaining the topology of objects in a volume. Part of the new algorithm is a new way of identifying these simple points. The new definition is general and can be used with any number of volume dimensions: Count the number of connected components of the foreground and background before and after removing the voxel in question, if the numbers of connected components do not change, the voxel is a simple point. Additionally, we do not want to remove a voxel if it has

only opposing neighbours (see (3.3)). Thus, in 3D, a foreground point $x \in X$ is a simple point if and only if:

$$NC_6[X \cap (N_6(x) - \{x\})] = NC_6[X \cap N_6(x)] \quad (3.1)$$

$$NC_{26}[\bar{X} \cap (N_{26}(x) - \{x\})] = NC_{26}[\bar{X} \cap N_{26}(x)] \quad (3.2)$$

$$\begin{aligned} &|[(X \cap N_{(-1,0,0)}(x))XOR(X \cap N_{(1,0,0)}(x))] \\ &\bigcup[(X \cap N_{(0,-1,0)}(x))XOR(X \cap N_{(0,1,0)}(x))] \\ &\bigcup[(X \cap N_{(0,0,-1)}(x))XOR(X \cap N_{(0,0,1)}(x))]| > 0 \end{aligned} \quad (3.3)$$

where $NC_N(Y)$ returns the number of connected components in a collection of points Y given a neighbourhood N, $N_6(x)$ is the 6-neighbourhood of x, and N_{26} corresponds to the background neighbourhood given a 6-neighbourhood for the foreground. A 6-neighbourhood was used for the foreground because skeletons of higher neighbourhoods can easily be derived from lower neighbourhood skeletons. $N_{(\Delta a,\Delta b,\Delta c)}(x)$ is the neighbouring voxel at the relative position (a, b, c) to x, and $|Y|$ corresponds to the cardinality (number of elements) of the set Y.

A simple one-pass algorithm for finding the connected components is as follows (Matlab, 2005):

1. Scan all image voxels, assigning preliminary labels to nonzero voxels and recording label equivalences in a union-find table.

2. Resolve the equivalence classes using the union-find algorithm (Sedgewick, 1998).

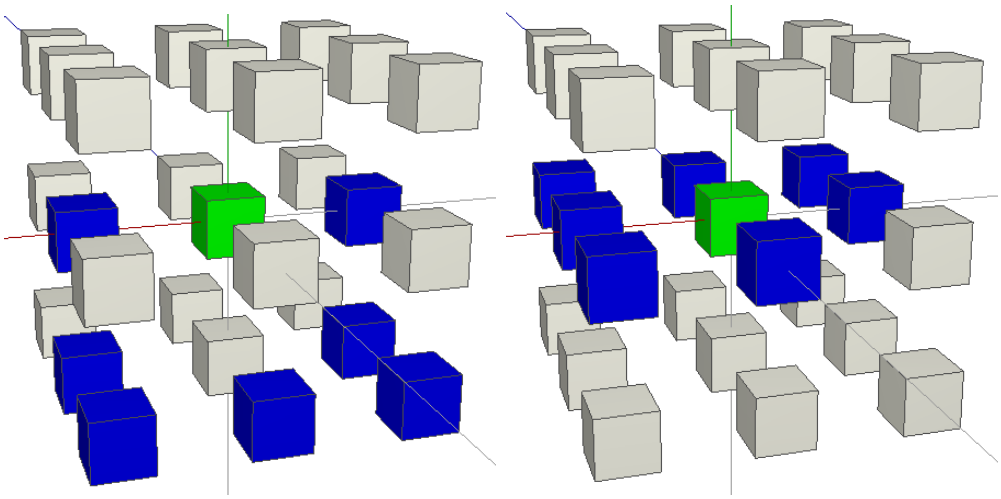
3. Relabel the voxels based on the resolved equivalence classes.

The first equation (3.1) is needed in order to keep the number of objects in the volume constant. To avoid the creation of new holes in the objects equation 3.2 must be fulfilled as well. Finally, equation 3.3 was added not to create obvious holes or separations that both equation (3.1) and (3.2) fail to identify. Figure 3.3 shows examples of such voxel setups where the centre voxel should be kept.

The definition of an end point is more intuitive. A voxel is an end point if it has only one neighbouring foreground voxel, and if this neighbouring voxel has only one more neighbour in addition to the original voxel. We choose this extended end point definition in order to keep the number of branches to a minimum. The definition of an end point $x \in X$ is thus as follows:

$$|X \cap (N_6(x) - \{x\})| = 1 \quad (3.4)$$

$$|X \cap N_{E\leq 2}(x)| = 3 \quad (3.5)$$



(a) (b)

Figure 3.3: Some voxels should be kept even though equations (3.1) and (3.2) are fulfilled. a) and b) are two example voxel setups that should be kept and equation (3.3) helps to retain.

where $N_{E\leq 2}(x)$ is the neighbours of x within the Euclidean distance ≤ 2.

Additionally, we define a *border voxel* as a voxel that has one foreground neighbour and one background neighbour in a given direction (i.e. back and forth in the x-direction, y-direction or z-direction in 3D).

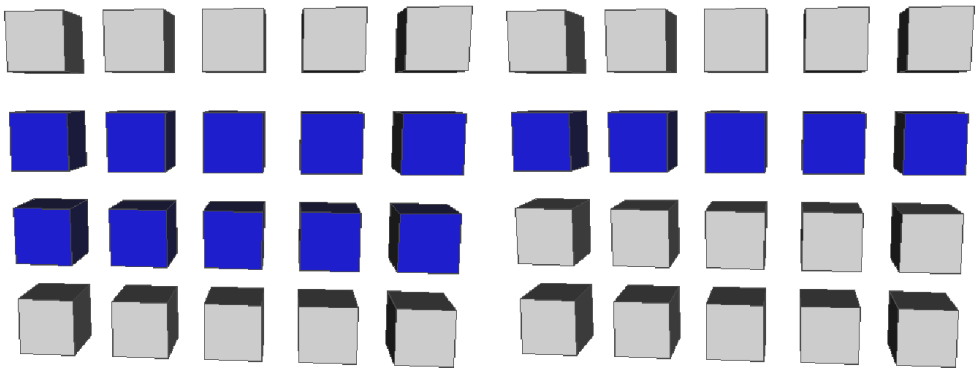
In certain setups such as is shown in figure 3.4, all voxels are marked for deletion during one iteration. In order to keep the structure in such cases, we define a *keep voxel* as follows: if the following voxel setup is found in the current directional thinning orientation (e.g. east): 0110 where the leftmost 1 is marked for removal during the east thinning orientation and the rightmost 1 is marked for removal during the west thinning orientation, remove the deletion mark for the leftmost 1.

Now that we have defined simple points, end points, border voxels and keep voxels, we can outline the skeleton algorithm as follows:

1. All border voxels, simple or not, are ordered in a deletion list first by the number of neighbours and then by a directional thinning order (for instance: up, down, north, south, west, east). Voxels that have fewest neighbours will be tested for removal first.

2. End points are identified.

3. *Keep voxels* are searched for successively in all the directional thinning orientations, and removed from the deletion list as they are found.



(a) (b)

Figure 3.4: This figure shows an example voxel setup where all voxels are marked for deletion. a) The initial voxel setup. b) The result of our algorithm, where voxels are kept if neighbouring voxels in the opposite directional thinning order are also marked for deletion.

4. Each voxel in the list is iteratively removed if it is a simple point and not an end point.

5. Repeat 4 until stability.

6. Repeat 1-5 until stability.

3.3 Results

An example result compared with the skeleton method in Palagyi et al. (2001) is shown in figure 3.2. As the figure shows, the simple point definition given in Malandain and Bertrand (1992) coupled with the deletion order presented in Palagyi et al. (2001) do not produce a sufficiently compact skeleton. It was also shown that this simple point definition fails in certain voxel setups (see figure 3.1). Furthermore, figure 3.5 shows that our method produces an even more compact skeleton than the widely used skeleton algorithm given in Guo and Hall (1989). It is apparent that the proposed method is not as dependent on the original boundary voxels as previous simple point algorithms are. Additionally, the new skeleton algorithm is not heavily controlled by parameters such as is typical in the distance based skeleton methods.

In 2D, however, the difference between our proposed algorithm and the algorithm given in Guo and Hall (1989) is virtually non-existent. As shown in figure 3.6, only a few pixels differ, which demonstrates that the proposed algorithm performs similarly to the most popularly used compact 2D skeleton

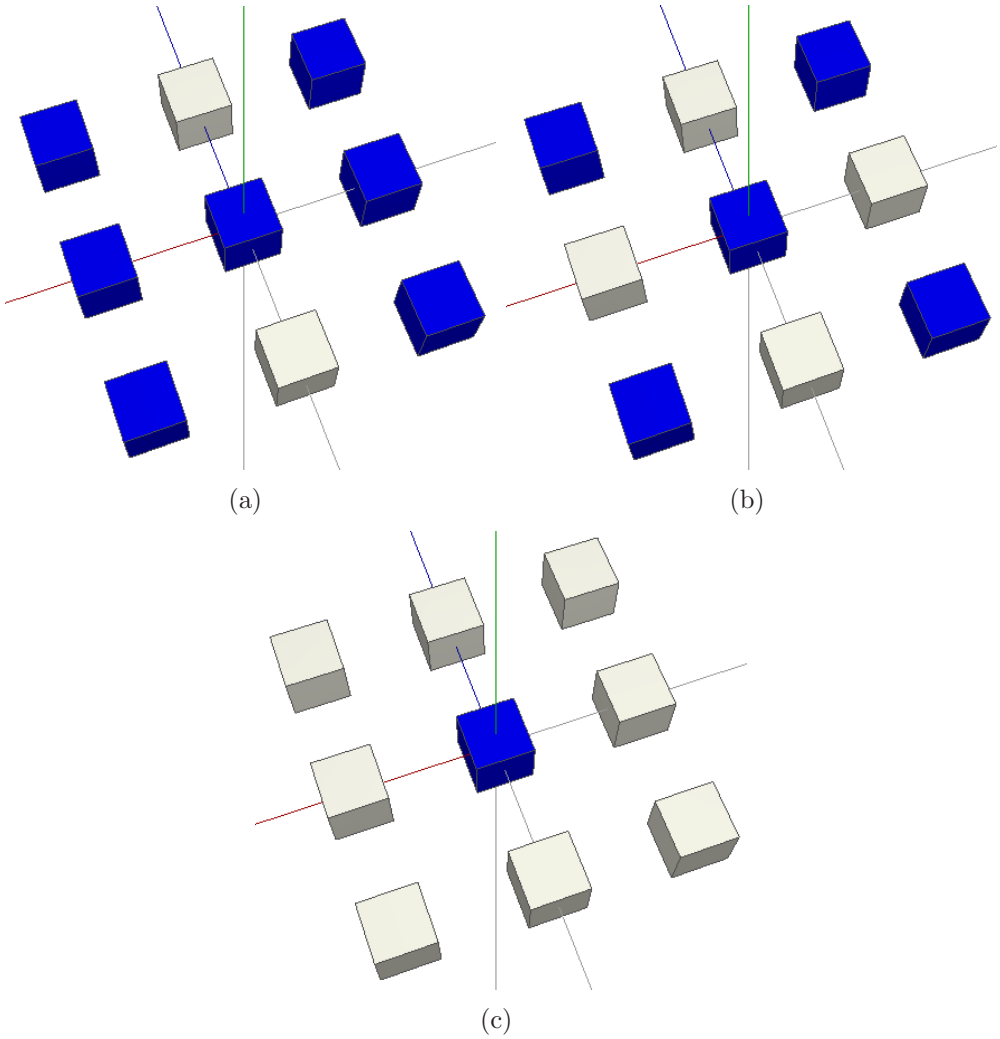
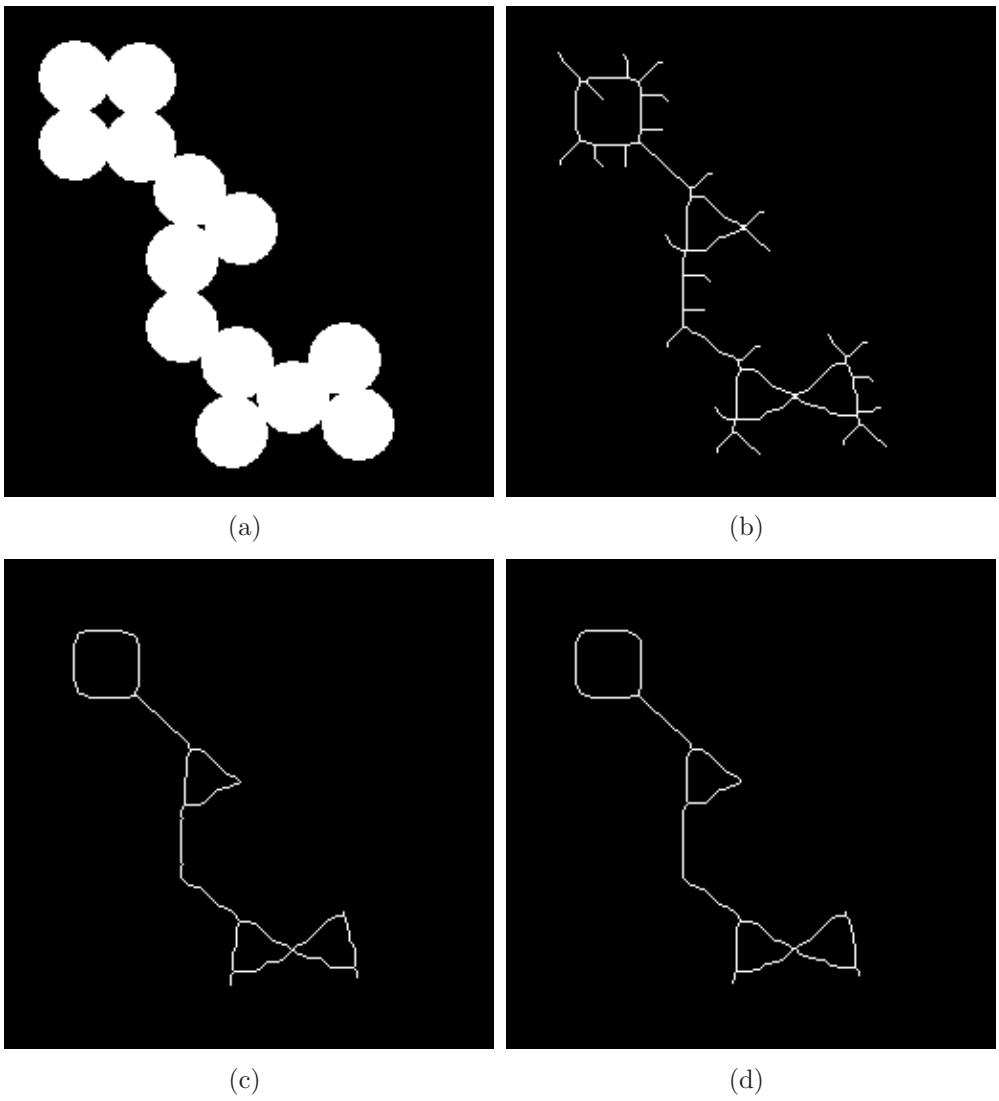


Figure 3.5: This figure shows that the proposed method produces an even more minimal skeleton than the widely used 2D skeleton method given in Guo and Hall (1989). a) Object to be skeletonised. b) Skeleton produced by Guo and Hall (1989). c) Skeleton produced by the proposed method in section 3.2.

algorithm. The result from our proposed algorithm is made 8-connected, by simply removing corner pixels from the 4-connected skeleton result.

Figure 3.7 and 3.8 show an example result from applying the proposed skeleton algorithm to a hepatic vessel segmentation presented in Eidheim (2005). As a comparison, figure 3.9 presents the result from applying the skeleton algorithm proposed in Palagyi et al. (2001). The latter result has multiple erroneous branches and even vessel loops that were not present in



(a) (b)

(c) (d)

Figure 3.6: The proposed skeleton procedure compared to previous 2D skeleton algorithm. a) 2D image to be skeletonised. b) Skeleton produced through morphological thinning (Soille, 2003; Gonzalez and Woods, 2008). c) Skeleton produced using the algorithm from Guo and Hall (1989). d) Skeleton result from applying the proposed skeleton method. For comparison, the result in d) is made 8-connected from the 4-connected result.

the original object structures.

Another comparison of the proposed algorithm and the algorithm given in Palagyi et al. (2001) is given in figure 3.10 and 3.11. The algorithms are here applied on a segmentation of hepatic vessels of a second CT scan.

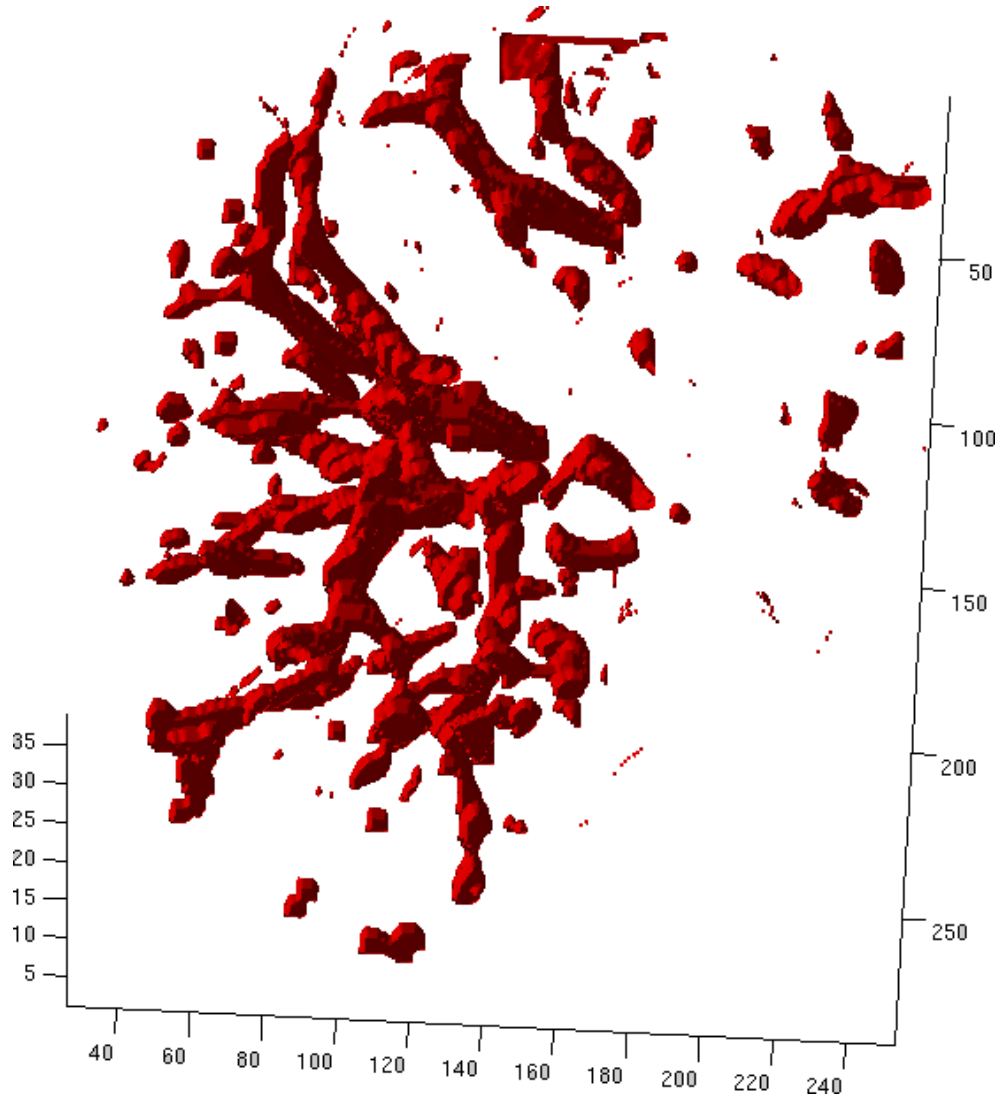


Figure 3.7: Segmented vessels that is to be skeletonised using the proposed skeleton method. See figure 3.8.

3.4 Discussion

The motivation behind the graph based approach was to derive the most likely hepatic vessel setup even though a CT or MR scan was of low resolution or distorted by noise. Compared to previous work, our method has the potential to produce more likely vessel graphs with appropriate parameters. Setting the parameters is although often difficult making the method unreliable in critical applications. This also applies to previous methods to

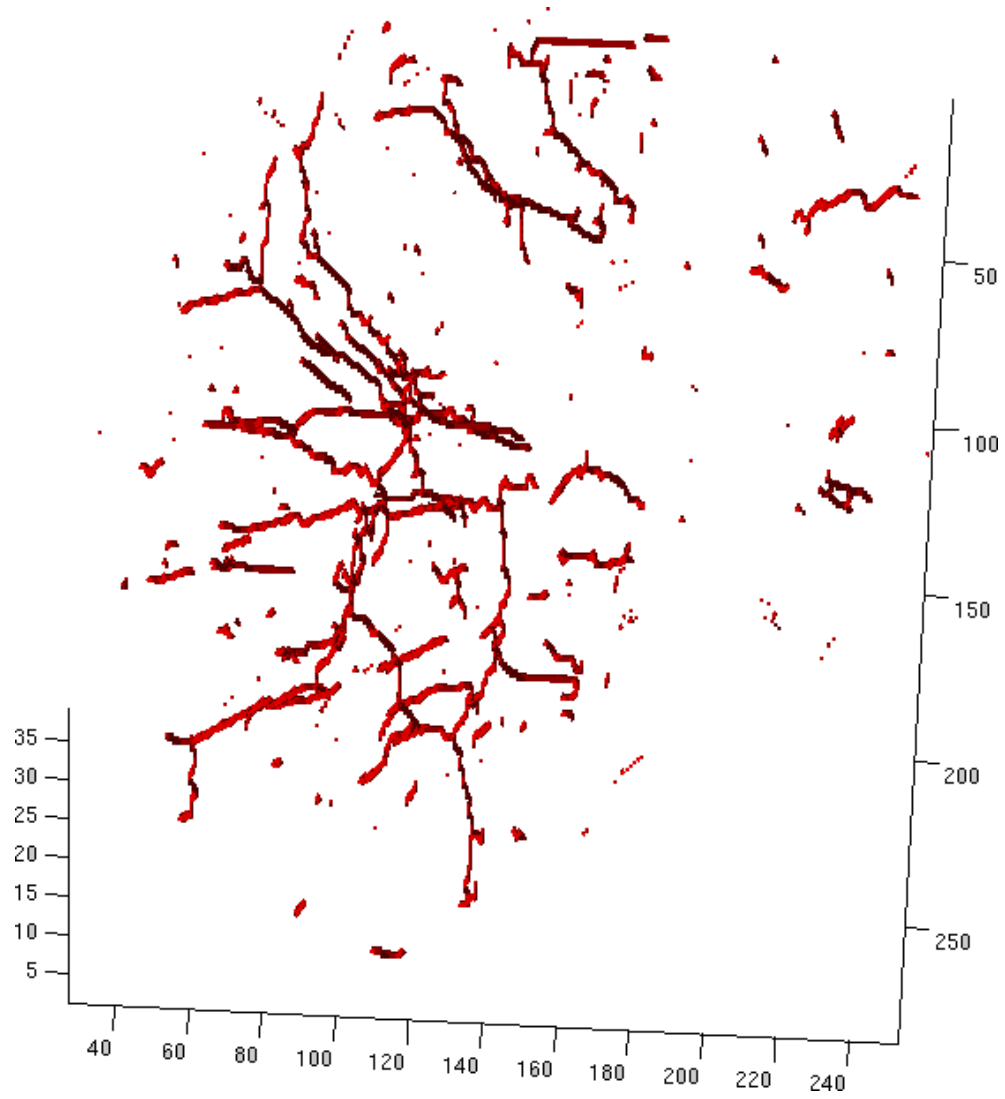


Figure 3.8: Skeleton result from the proposed skeleton algorithm applied on the vessel segmentation shown in 3.7.

a great extent. As the image modalities become more advanced, however, the resolution of the CT or MR scans improves and more direct methods for vessel extraction can be applied.

The work on the graph based method did, however, result in a new iterative and robust 3D skeleton algorithm that represents a noticeable contribution to the image processing community. In contrast with distance based skeleton algorithms, it is not controlled by any user defined parameters and the algorithm can be parallelised and run multi-threaded. The initial results

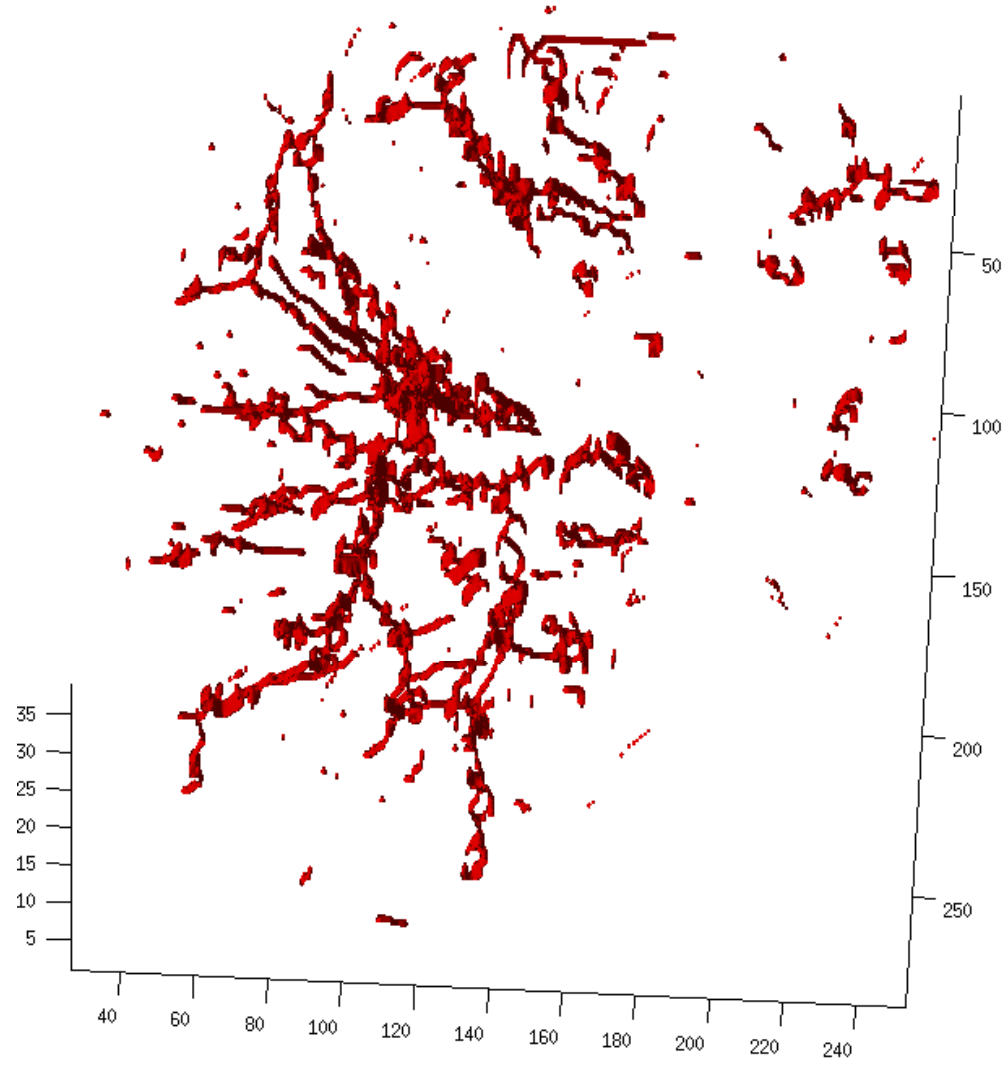


Figure 3.9: Skeleton result from the previously proposed skeleton algorithm (Palagyi et al., 2001) applied on the vessel segmentation shown in 3.7. As shown, this algorithm produces skeleton with multiple erroneous branches and loops.

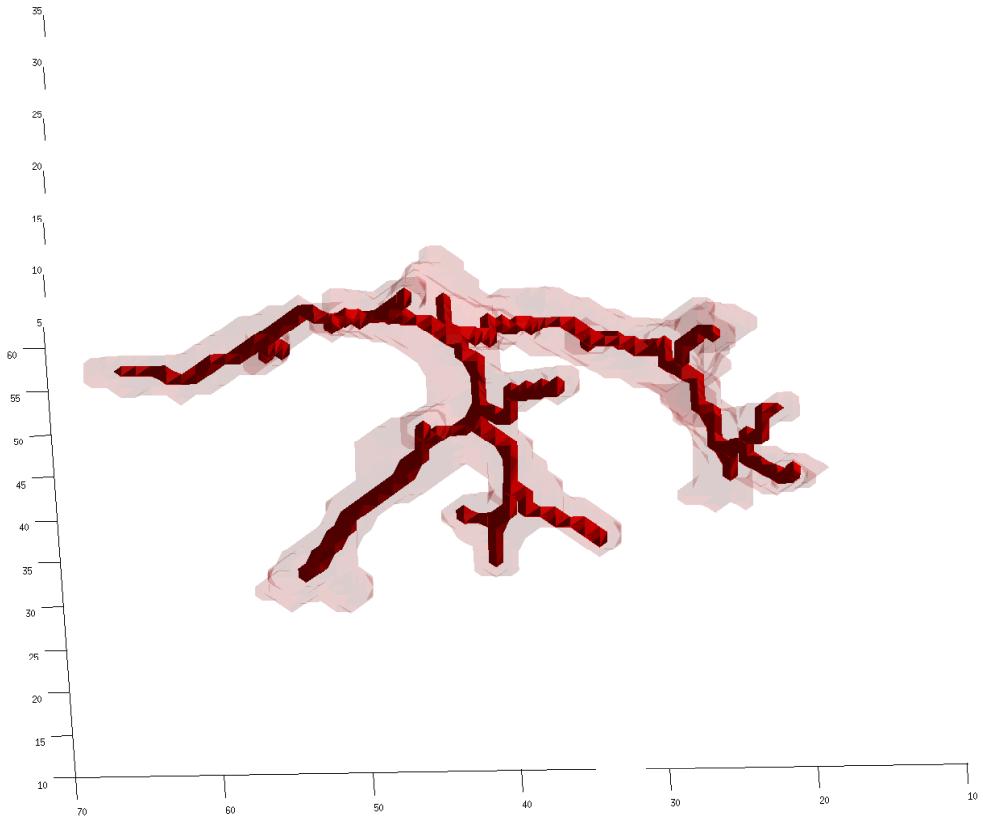


Figure 3.10: Skeleton result from the proposed skeleton algorithm applied to a second segmentation of hepatic vessels.

demonstrates that the 3D skeleton algorithm outperforms popular previous iterative methods in creating a compact representation of 3D objects. It is a general algorithm that can be used on 2D data as well, and even though it has not been studied extensively it is possible that the method can be extended to 4 or higher dimensions as well.

A 2D result from the proposed skeleton algorithm was compared to a result from the most well known compact skeleton representation given in Guo and Hall (1989). Both methods produced similar results, however, the method given in Guo and Hall (1989) has not been defined in 3 dimensions such as the presented skeleton algorithm is.

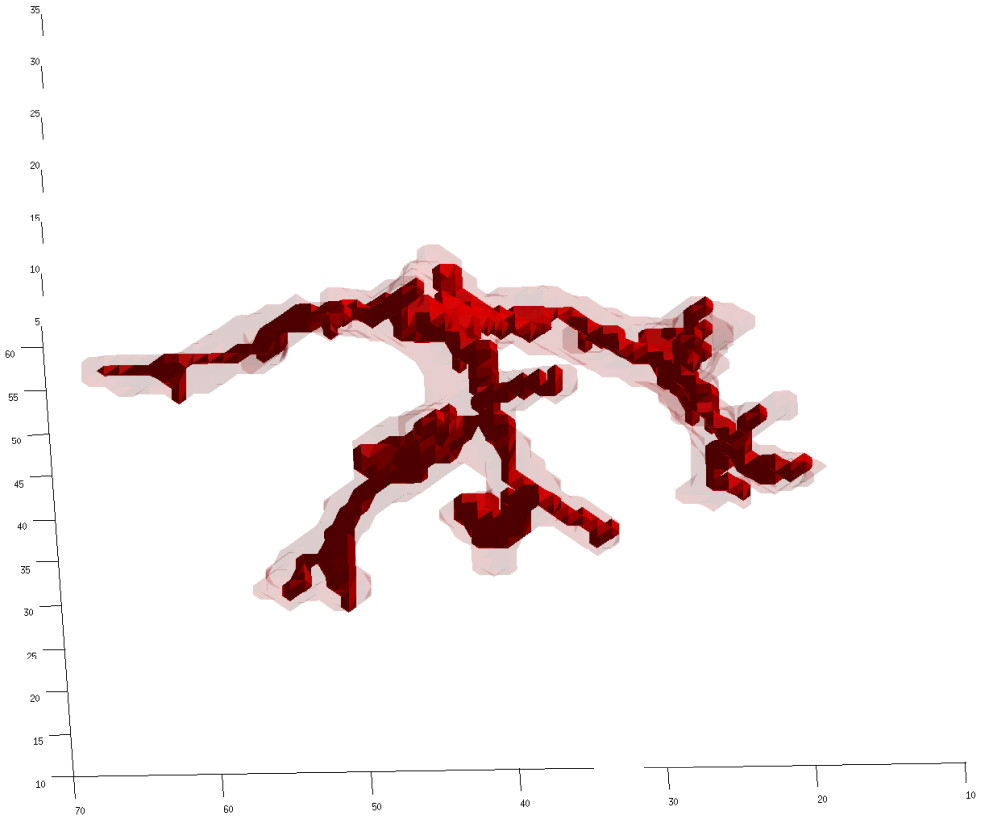


Figure 3.11: Skeleton result from the previously proposed skeleton algorithm (Palagyi et al., 2001) applied on a second hepatic vessel segmentation.

Chapter 4

Texture based segmentation of the liver

Prior to finding hepatic vessels and possible liver tumours, a sound segmentation of the liver is needed. Furthermore, visualisation of the liver and its internal structures gives a more complete overview for radiologists and surgeons. Segmentation of the liver is also important for doing an anatomical segmentation of the liver and to plan the parts of the liver that can be surgically removed.

Segmentation of the liver is a challenging task, and no general method has been presented before that gives reliable results. The main problem is that there are small texture changes and no significant boundaries exist between the liver and much of the surrounding tissue. Additionally, the liver can vary greatly in shape and location from person to person, and it is not a simple task to utilise statistical models to segment new datasets. Abnormal liver shapes are also difficult to include in a statistical model, but to generate a reliable and general method such cases must be successfully segmented as well.

Various methods implemented for liver segmentation were mentioned in chapter 2. In Soler et al. (2001); Gao et al. (1996); Heuch (2003); Liu et al. (2005) for instance, a snake model (Kass et al., 1988) was used to delineate the liver boundary using edge information. The edges in these papers were enhanced by two similar methods; anisotropic diffusion (Perona and Malik, 1990) and gradient vector flow (Xu and Prince, 1998, 2000). Another example of liver segmentation based on region boundary was described in Nakayama et al. (2006), where post processing techniques were added to refine the result using knowledge of the liver boundary. Similarly, in Seo et al. (2004), the liver was initially segmented through thresholding based on image histograms, and various techniques utilising anatomical knowledge were then

used to find the liver and improve the segmentation results. An interesting approach can also be found in Montagnat and Delingette (1997), where a reference model guided the deformable model to a large extent. These methods, however, are highly vulnerable to noise and regions where there are no significant edges between the liver and surrounding tissue. In Pham et al. (2007), however, texture is used as the basis of the liver segmentation. The texture is represented as statistical features from the regions' co-occurrence matrices, and regions of similar classification are merged. Even though the results from this method are promising, the features and classification algorithm do not separate the regions adequately and further development of the procedure is needed. Texture based segmentation also has the advantage over contour and shape based methods in that the interior of an object is segmented as well.

In this chapter, a new method for automatic segmentation of the liver will be presented. The method is based on an extended representation of texture in medical volumes. Using this representation, a classifier is first trained from a homogeneous starting region or seed point. This classifier is used next on surrounding regions, and similar regions are used to further update the classifier. An important advantage from using the chosen representation and classification method is that the classifier also unlearns tissues that are far away from the voxel in question, and thus the classifier avoids the risk of becoming too general in its function to classify new tissue. The segmentation method has the potential to be guided by physical and statistical constraints avoiding the most unrealistic results.

4.1 Method

The method is initiated from a homogeneous texture region or seed points, and new voxels are added to the segmentation result in a region growing scheme. The classifier is trained initially from the first voxels, and as new voxels are added to the segmentation result they may also be added as training samples to the classifier. The procedure will be explained in 3D, but the majority of the examples and illustrations are shown in 2D.

4.1.1 Description

The description of texture should be invariant to translation and rotation. The latter property is important to, for instance, generalise the texture in vessels having different orientations. In addition, the final description should take into consideration small possible texture change in an organ. The de-

0	2	1
0	3	3
1	1	2

(a)

	0	1	2	3
0	1	1	1	1
1		1	2	2
2			0	2
3				1

(b)

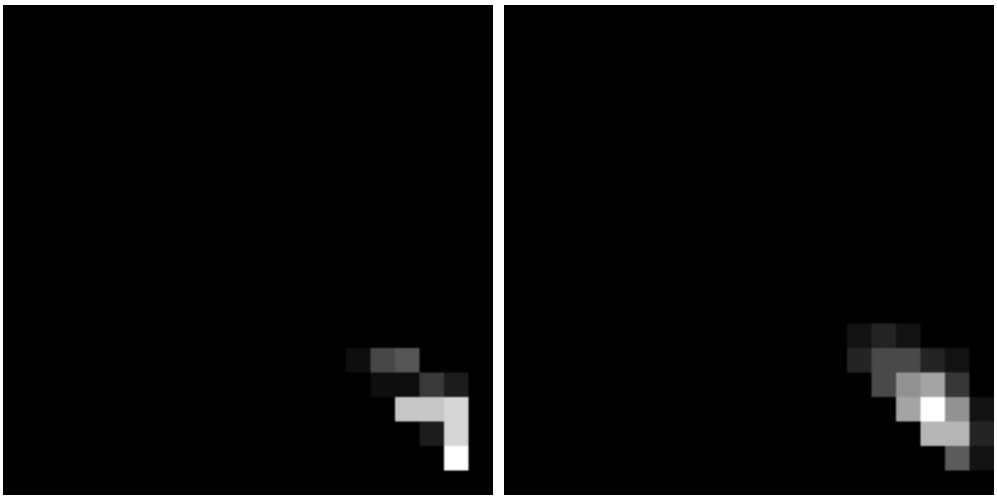
Figure 4.1: An example second-order representation of a image is shown in this figure. The representation is invariant to rotation and translation. a) A constructed image example. b) Resulting representation of a).

scription should not, however, be more compact due to the risk of loosing information that could help the classifier to distinguish two tissue types. This contradicts earlier attempts at using texture as the basis of CT segmentation. These methods typically extracted statistical descriptions from the texture representation, but in doing so important information that could assist the segmentation were lost. On the other hand, it leads to the disadvantage of working with large amounts of data. The runtime of the classification will consequently increase and more memory or disk storage is needed.

Julesz et al. (1973) states that textures with identical first- and second-order statistics cannot be pre-attentively discriminated by humans. Furthermore, Bevk and Kononenko (2002) summarise the work of Julesz and states that first-order statistics proves insufficient in distinguishing two textures if close-to-human perception is to be achieved. First-order statistics can be derived from second-order statistics, and we can therefore presume that we can base our description on second-order statistics alone. A common way to represent second-order statistics of a texture is by a co-occurrence matrix (Gonzalez and Woods, 2008; Sonka et al., 1999). First-order statistics are represented by a histogram.

The texture description we used was a slightly modified co-occurrence matrix where the textures were made invariant to rotation. Given a voxel position, (x_s, y_s, z_s), and the surrounding region (an 11 voxel wide cube), all neighbouring 6-connected voxel pairs were added to the co-occurrence matrix after sorting them by intensity value as illustrated in figure 4.1. This figure shows a 2D example, but the representation is a 2-dimensional matrix also when representing a 3D volume.

Since the number of samples in the co-occurrence matrix typically is low, the Parzen window method (Duda et al., 2001) was used to approximate a



(a) (b)

Figure 4.2: The Parzen window method was used to approximate the co-occurrence distributions since the number of samples is typically low. a) Resulting distribution prior to applying the Parzen window method. b) Resulting distribution after applying the Parzen window method.

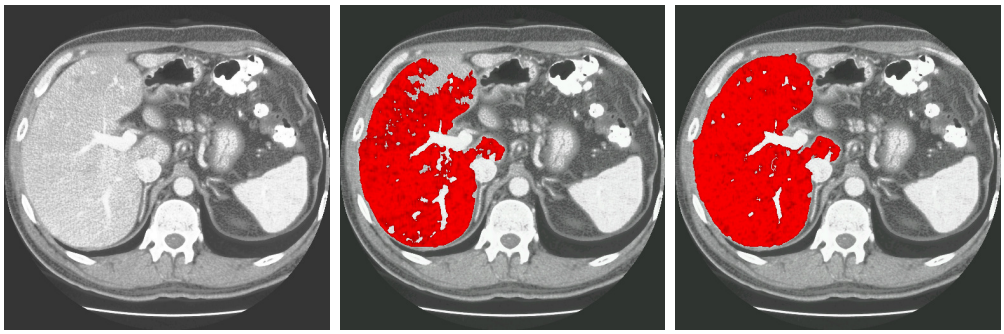
distribution from the N intensity samples:

$$C(x_s, y_s, z_s, x_i, y_i) = \sum_{n=1}^{N} \exp^{-(\frac{(x_i - x_n)^2 + (y_i - y_n)^2}{2\sigma^2})} \tag{4.1}$$

where (x_i, y_i) corresponds to the intensity coordinates of the co-occurrence matrix C. The standard deviation was set to $\sigma = h/\sqrt{N}$ (Duda et al., 2001), where h was typically set to 1/5 of the intensity range. The parameter h may, however, be adjusted more optimally with respect to the segmentation result according to the given task.

Memory usage presented a challenge and in order to keep the amount of data to a minimum, we grouped the intensity values into 20 bins after approximating the co-occurrence distribution. It is also possible to reduce the size of the co-occurrence matrix even further by exploiting the fact that some parts of the matrix is not used. Two example co-occurrence distributions, with and without the Parzen window approximation, is shown in figure 4.2.

Reducing the number of intensity values may worsen the quality of the results, and a good balance between memory usage, processing speed and quality of the segmentation must be carefully evaluated. A possibility is to have a nonlinear transform of the intensity values such that the intensities are optimally divided between the bins. This requires knowledge of the images to



(a) (b) (c)

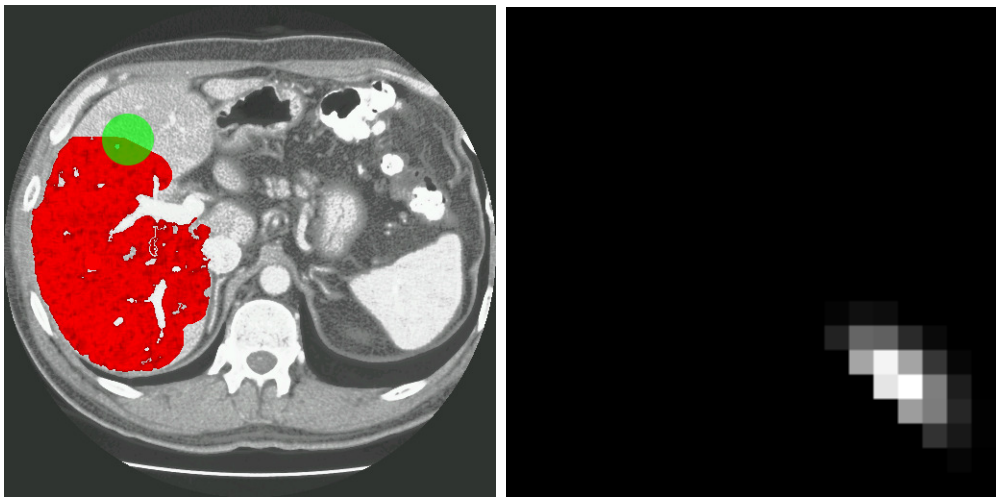
Figure 4.3: Example that shows the importance of the spatial dimensions in the feature space. a) Original CT slice where the liver is to be segmented. b) Segmentation of a) based on second-order statistics alone. c) Segmentation of the same CT slice and parameters as in b), but with the spatial dimensions included in the feature space.

be processed, and we did not add this requirement to the processing presented in this chapter.

The final feature space consists of five dimensions when segmenting a 3D dataset: three spatial dimensions and the two dimensions of the co-occurrence matrix. The three spatial dimensions were added so that the classifier can take into account small texture change in the spatial domain. The classifier will easily be too general or too specialised if these dimensions were disregarded. The spatial dimensions can have a lower resolution than that of the dataset to save memory and to speed up the procedure. Figure 4.3 shows an example segmentation of a 2D CT slice to illustrate the importance of adding the spatial dimensions to the feature space. The segmentation result in 4.3 (b) does not include the whole liver because there is a slight change in the liver texture towards the chest, but by including the spatial dimensions in the feature domain, spatial texture changes can be considered and the whole liver can be more accurately segmented as shown in the segmentation result in 4.3 (c).

4.1.2 Classification

To classify a voxel, the voxel's co-occurrence distribution was compared to an accumulated co-occurrence classifier distribution p through a normalised dot product (cosine angle) measure. If the product was above a predetermined threshold, the new voxel would be classified as the same class as the voxels that were compared against. The classifier distribution p was derived



(a) (b)

Figure 4.4: a) An example region, shown in green, of radius 27 with its corresponding $p(x_s, y_s)$ presented in b). x_s and y_s are the centre of the region. Red pixels are already segmented liver tissue and their intensity represents $\dot{P}$.

from downscaled co-occurrence matrices of already classified voxels in the surrounding region:

$$p(x_s, y_s, z_s, x_c, y_c) = \sum_{\sqrt{(x-x_s)^2+(y-y_s)^2+(z-z_s)^2} \leq r} C(x, y, z, x_c, y_c) \tag{4.2}$$

where r determines how many distributions should be summed. In our tests r was initially equal to 25, but if insufficient training data was found within that radius, the radius was expanded until sufficient data was found. The accumulation in the spatial dimensions ensures that the classifier changes with the voxel coordinates rather than having the same distribution for the whole object to segment. In figure 4.4, an example region is shown with its corresponding p, where x_s and y_s are the centre of the green region.

After the distribution was constructed, new voxels could be classified by taking their corresponding co-occurrence matrices and calculating the normalised dot product (cosine angle) $\dot{P}$:

$$\dot{P}(x_s, y_s, z_s) = \frac{\sum_{i,j=1}^{M} p(x_s, y_s, z_s, i, j) C(x_s, y_s, z_s, i, j)}{\sqrt{\sum_{i,j=1}^{M} p(x_s, y_s, z_s, i, j)^2} \sqrt{\sum_{i,j=1}^{M} C(x_s, y_s, z_s, i, j)^2}} \tag{4.3}$$

where M is the size of the co-occurrence matrix C. If $\dot{P}$ was above a threshold

T, this new voxel (x_s, y_s, z_s) was classified as the same tissue and its co-occurrence samples were added to the distribution p.

The normalised dot product was used so that $\dot{P}$ always is 1 when the distributions are equal, and to ensure that $\dot{P}$ is between 0 and 1. The threshold T is in this way simpler to set and not extensively dependent on the texture in the dataset.

Qian et al. (2004) states that the normalised dot product measure and the Euclidean distance measure yields similar results in cases such as ours where the number of dimensions is high. The normalised dot product has an advantage therefore since the result will always be between 0 and 1 when all vector elements are either 0 or positive.

A step to make the liver segmentation even more robust is to add an additional threshold T_p that $\dot{P}$ has to exceed in order to add a new co-occurrence matrix to the classifier p. Especially borders along the object to be segmented can eventually influence the classifier significantly, and thus by adding a second threshold this influence is greatly reduced.

4.1.3 Region growing

There are two ways to start a segmentation of an object using the previously described method. First, a user can select a seed point and voxels can be tested and added to the classifier through a region growing scheme. In the case you want the algorithm to operate fully automatically, homogeneous regions can be found by calculating co-occurrence matrices of the whole image. Then, for each homogeneous region, a separate classifier is made and trained by the voxels in the region and their co-occurrence matrices. New voxels are again added in a region growing scheme. An advantage with the latter method is that surrounding tissue can be used to classify uncertain border voxels between two tissue types. By having a distribution for both tissues, you can take advantage of, for instance, Bayesian risk in addition to the selected threshold value, and a less restrictive threshold can be used [1].

There are two common 2D region growing schemes, namely 4-neighbourhood and 8-neighbourhood growing. Both schemes are suboptimal with respect to circular growth and the boundary of the growing region has abrupt directional changes. Therefore, in order to classify new voxels always based on the nearest neighbour voxels, a more circular region growing method was needed. We decided to use the iterative method outlined in Coiras et al. (1998), where corners are suppressed at certain iterations and a near circular

[1] We did not, however, study Bayesian risk in combination of the proposed segmentation procedure.

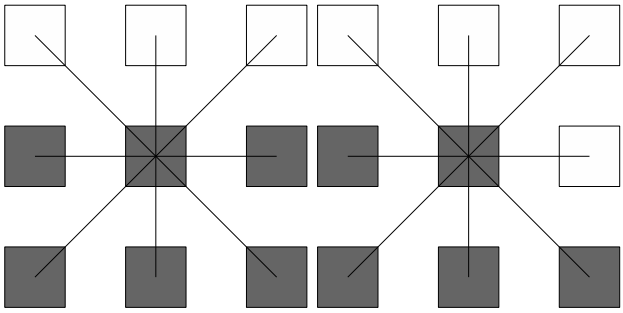


Figure 4.5: This figure illustrates the definition of a corner voxel in the iterative region growth. If $n-2$ (where n is the number of dimensions in the dataset) or fewer of the opposite voxels are true, the centre voxel is defined as a corner voxel. a) The centre voxel is not a corner voxel. b) The centre voxel is a corner voxel.

region growing is achieved through 4-neighbourhood and 8-neighbourhood growth. This method was extended to 3D, using 6-neighbourhood and 26-neighbourhood growth. The only difference with respect to the method in Coiras et al. (1998), is the definition of corner voxels that needed to be generalised for 3D processing. The new definition is as follows in n-dimensions: if there exist $n-2$ or fewer opposite true voxels (considering the neighbouring voxels, that is the 26-neighbourhood in 3D) with respect to the centre voxel, the centre voxel is defined as a corner voxel (see figure 4.5).

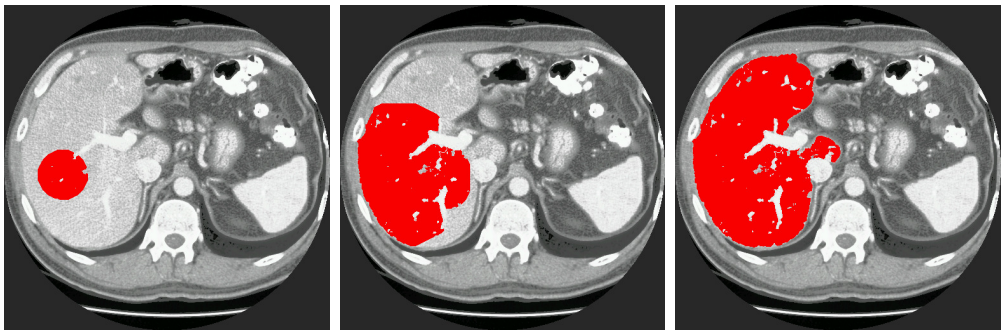
The stopping criteria of the region growing procedure is based on the texture classifier, that is the region growth at a point (x, y, z) is stopped if $\dot{P}(x, y, z)$ is below the threshold T.

A possibility to keep the segmentation result compact, the region growth could be restricted if the curvature of the local boundary overstepped a predetermined threshold. Thin faulty regions could then be disregarded and unnecessary computations avoided. We did not, however, study this in detail.

4.1.4 Method summary

We add the following method summary to give a better overview of the proposed procedure:

1. The segmentation is initiated from a homogeneous texture region or seed points, and the classifier distribution p is computed initially from these first voxels and their respective co-occurrence matrices C.



(a) (b) (c)

Figure 4.6: This figure shows an example runtime of the proposed algorithm. A seed point was given by the user in the centre of the first segmentation result a). b) and c) show the growth of the segmentation.

2. Voxels surrounding newly segmented voxels are found through a region growing scheme.

3. The voxels found in step 2 are included in the segmentation if their $\dot{P}$ exceed a threshold T.

4. The voxels found in step 2 and their respective co-occurrence distributions C are included in the classifier distribution p if their $\dot{P}$ exceed a threshold T_p.

5. Repeat steps 2-4 until stability.

4.2 Results

A sound segmentation of the liver was achieved without extensive parameter tuning. Figure 4.6 shows an example result of the proposed method applied on a single CT image. A seed point was given by the user in a homogeneous liver texture region. As shown in the figure, the segmentation distinguished the liver texture from the surrounding tissue even though the boundary between the tissue types is diffuse. The threshold T for this particular dataset was set to 0.65, but this threshold is highly domain- and case-dependent.

An example liver segmentation of a whole CT scan is shown in figure 4.7. The result was not smoothed or post-processed before presentation in order to present the original result of the algorithm. Even though the brightness and contrast of the CT slices change slightly, an accurate segmentation is achieved. This was the first complete result that was produced through

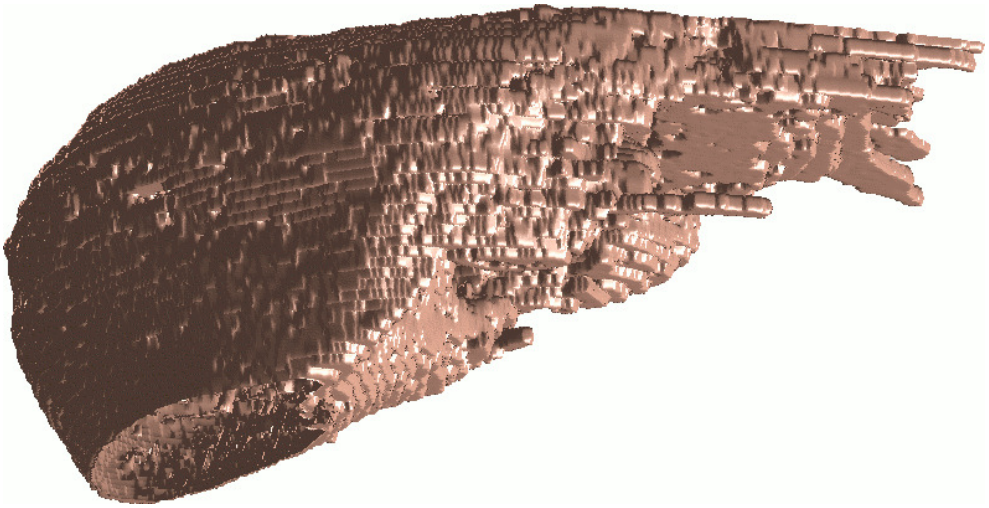
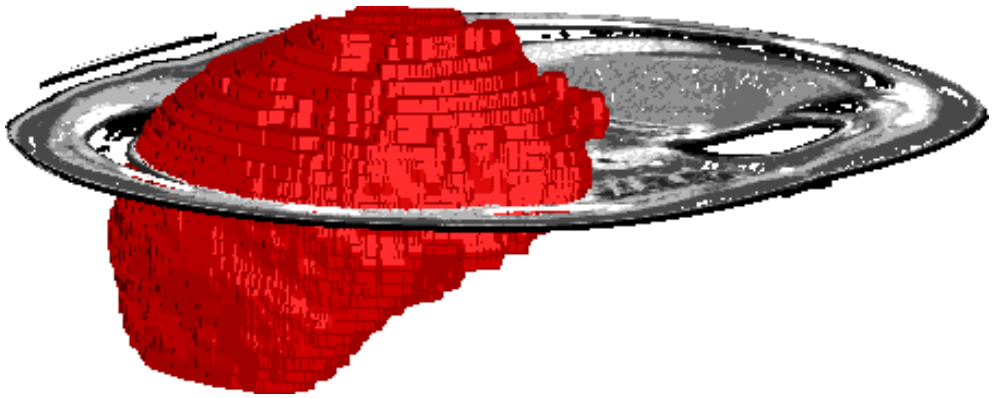


Figure 4.7: A complete 3D liver segmentation from a CT scan is presented in this figure.



xs

Figure 4.8: A complete 3D liver segmentation from a second CT scan is presented in this figure.

the algorithm. In this early result, the volume was segmented slice by slice in 2D, however, all of the succeeding 3D examples are results from a 3D implementation of the procedure.

Two more liver segmentations of different CT scans can be seen in figure 4.8 and 4.9. One threshold $T = 0.75$ was used and the result was post-processed through morphological opening and closing (Soille, 2003) instead of fine-tuning T for each scan. The purpose of these two figures is to show that good results can be achieved even though the threshold is not optimal by using simple post-processing of the results.

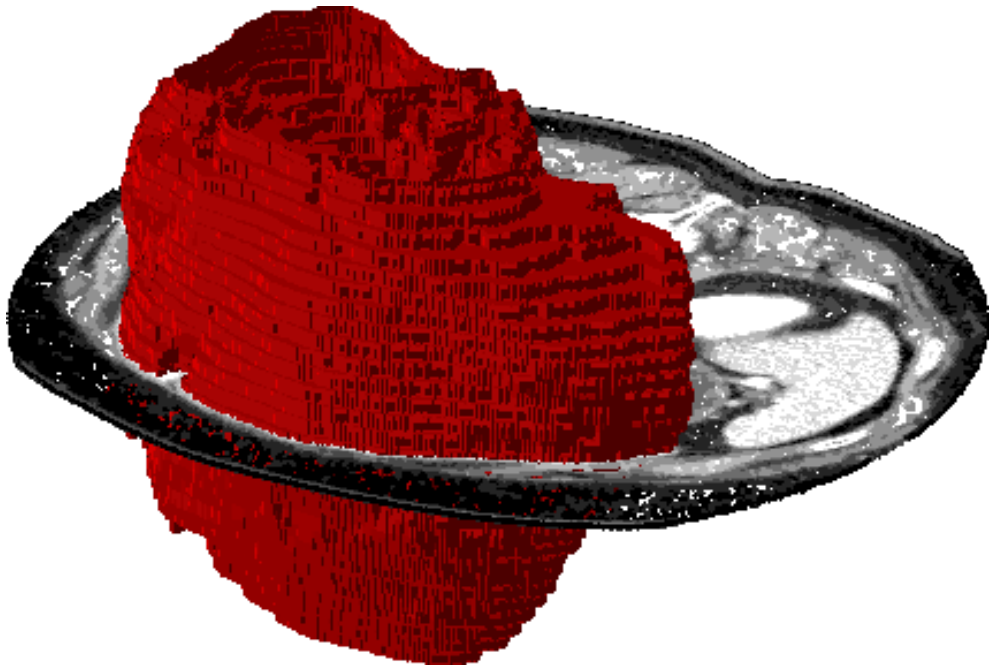


Figure 4.9: A complete 3D liver segmentation from a third CT scan is presented in this figure.

Figure 4.10, 4.11 and 4.12 show three additional liver segmentations from a fourth, fifth and sixth CT scan. The slices in these two CT scans were varying significantly in intensity and contrast, and additional processing was needed to overcome this problem. The slices were first histogram equalised (Gonzalez and Woods, 2008) individually before starting the segmentation. After the first segmentation was finished, additional segmentations were performed where the prior result was used as a starting point for the following segmentation. The prior result was first morphologically eroded with a structure element of equal size as the co-occurrence window, and all remaining voxels were used as seed points for the next segmentation. This process was repeated until stability.

Figure 4.13 shows an example result where seed points were selected automatically from homogeneous texture regions. A better boundary between two tissue types can generally be achieved by selecting the most probable tissue classification with respect to the tissue classifiers. At the end, two regions were merged if the ratio between the cardinality (number of elements) of the intersection and the union exceeded 0.2.

Figure 4.14 shows example segmentation results using T_p with different thresholds T, where T_p was derived from the formula $T_p = T + (1 - T)/2$.

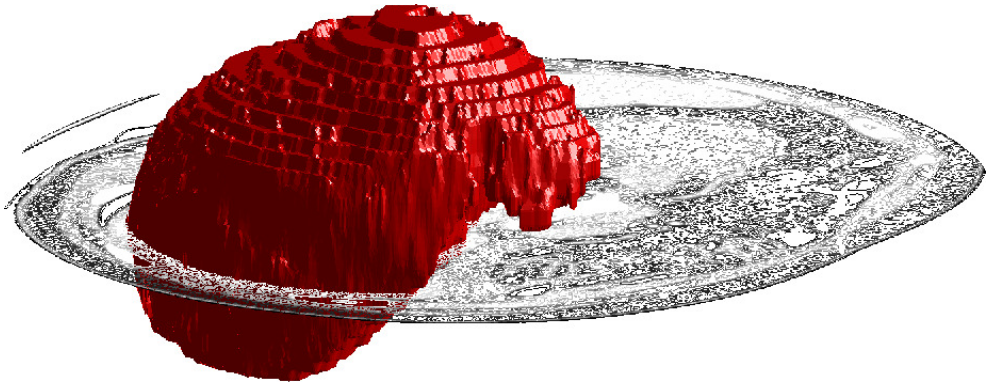


Figure 4.10: A complete 3D liver segmentation from a fourth CT scan is presented in this figure.

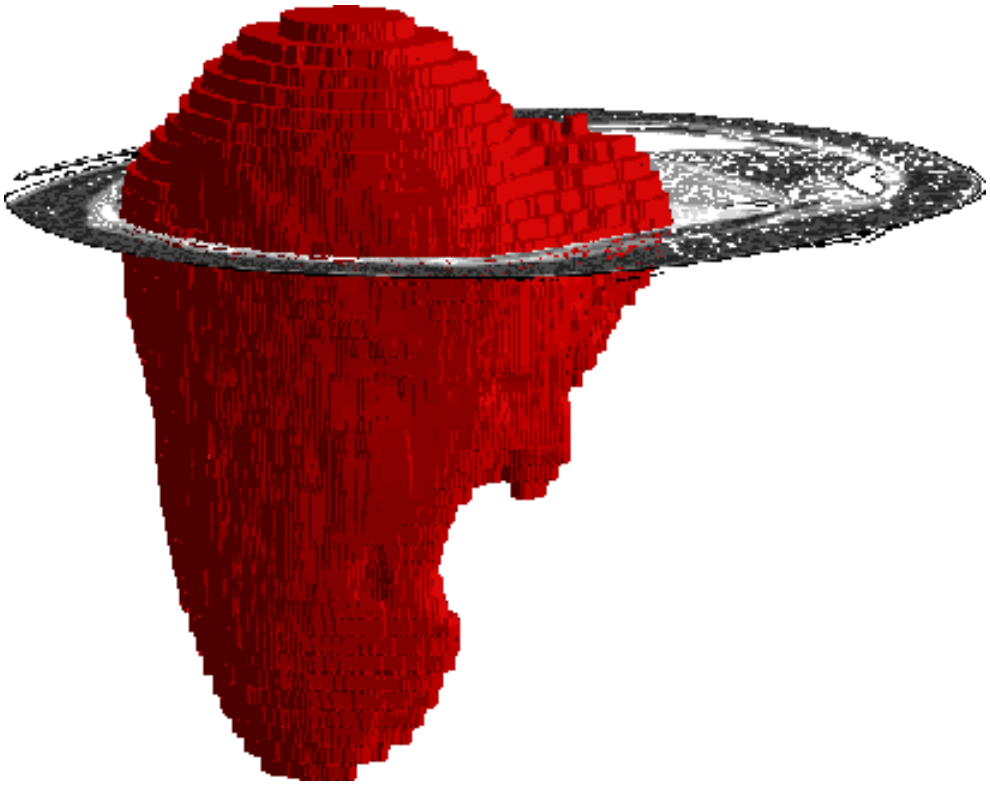


Figure 4.11: A complete 3D liver segmentation from a fifth CT scan is presented in this figure.

Segmentations 4.14 (d), (e) and (f) are sound with respect to this particular CT slice, that is to say $\frac{3}{20}$ of all parameter choices of T result in a high quality segmentation, and even $\frac{9}{20}$ cases result in a decent segmentation.

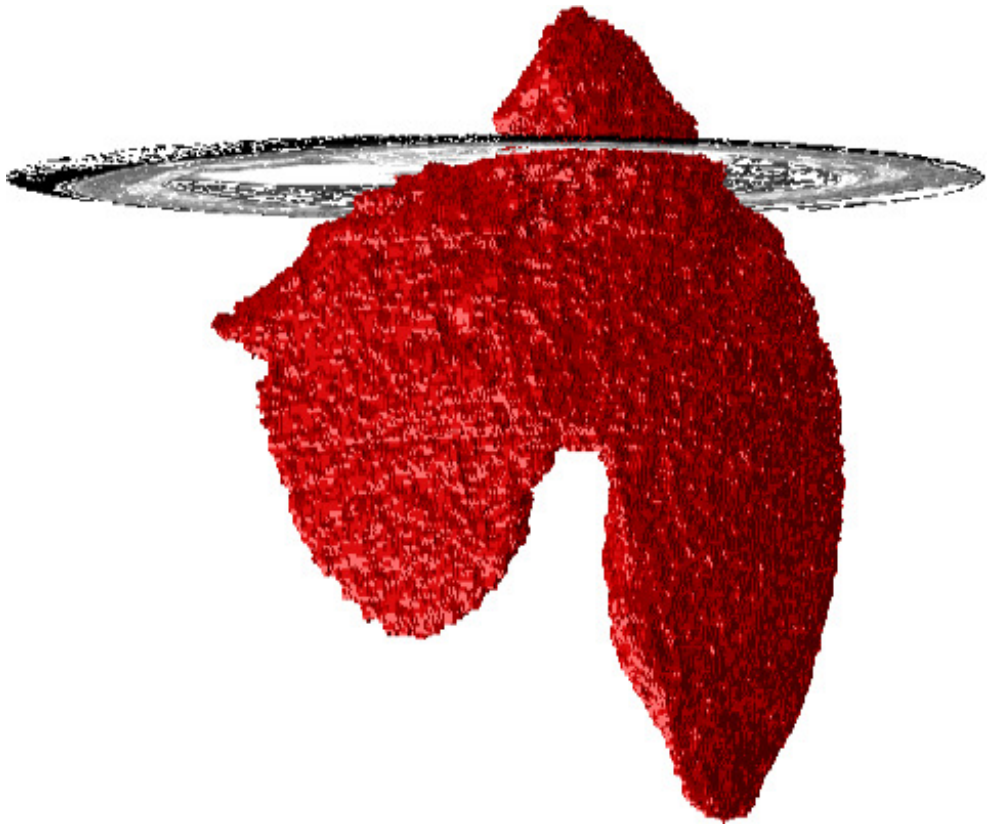
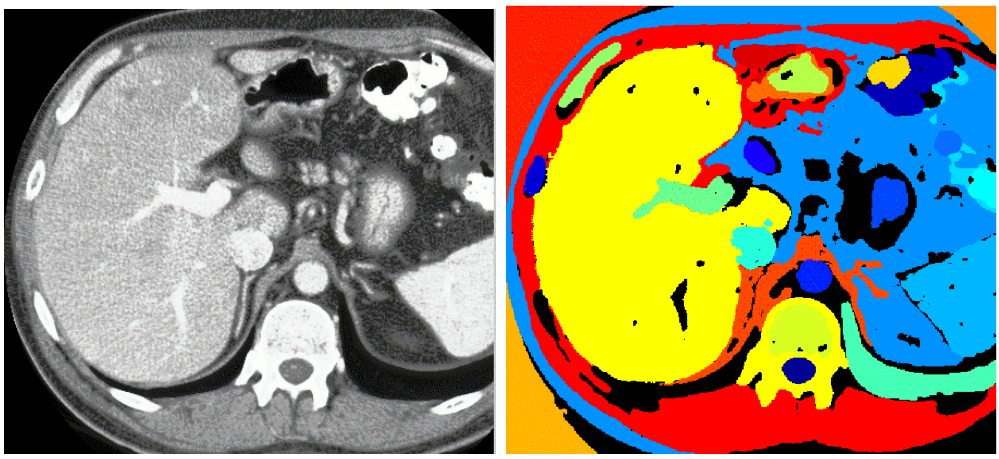


Figure 4.12: A complete 3D liver segmentation from a sixth CT scan is presented in this figure.

4.3 Discussion

Results show that the presented method is very robust and reliable. Regions were separated even though the boundary between them was diffuse, and the whole liver was segmented with the same parameters while the contrast and texture varied between the CT slices. We have not been able to find another segmentation procedure that can segment the liver as reliably as our proposed method. Commonly, organ boundaries form the basis of these previous techniques, whereas the proposed method takes advantage of the whole volume of the liver.

The texture was represented by a co-occurrence matrix that was invariant to texture rotation. A distribution was next built for already known tissue, and new tissue was compared to the already classified tissue through the normalised dot product measure. The advantage of using this measure over, for instance, the Euclidean distance measure is that it is normalised between



(a) (b)

Figure 4.13: This figure shows the result of a fully automatic segmentation of a CT image using the proposed algorithm. Seed points were set at homogeneous texture regions automatically, and highly similar regions were finally merged.

0 and 1 and the threshold T that defines if a region should be included in the end result is easier to set, even if the segmentation tasks are different.

The strength of the algorithm is its ability to adapt to slight texture variations and at the same time not be too general to separate distinct tissue types. The method is not dependent on numerous parameters that are difficult to set, but rather a single flexible parameter that decides whether a voxel is member of the given region. Additionally, known physical properties such as surface curvature can easily be added to limit extreme cases of inconsistent regions.

Since the proposed method searches for similar texture regions, the method will most likely not work well on objects with a high degree of texture variance. Although, if an object cannot be segmented by our method due to being comprised of partitions that are currently indistinguishable, a better resolution from future improved medical modalities might present the opportunity to successfully segment such objects as well.

Even though the results are of high quality compared to what has been done before, it is unlikely that we can base critical medical software on this method alone. Segmentation based on texture gives us a sound starting point, however, for further improvements based on, for instance, statistical data or physical models. An example implementation of texture based segmentation guided by statistical models is described in chapter 6.

The proposed method is reliant on only one or two parameters, depending on if the user wants the second parameter T_p to be automatically derived or not. Nevertheless, it is hard to accomplish a sound segmentation of an entire CT or MR scan with one set of parameters. The reason is that the texture can vary in intensity and contrast between the slices to a high degree. Possible solutions to this challenge are further normalisation of the slices, shape based models restricting the end result, and post-processing of the end result.

The runtime of the algorithm is another possible disadvantage. Before optimisation it took approximately 70 minutes to segment a liver in a 512x512 image (see figure 4.14 f)) on a Intel Pentium 4 2.8GHz computer using Matlab. Since several voxels are considered at each iteration, we could implement the algorithm multi-threaded and run the classification in parallel. On a modern two dual-core 2.4GHz processor computer, using optimised multi-threaded programming in C, we achieved a runtime of 18 seconds on the same method that was run in Matlab. With the increasing number of cores in a single processor combined with the possibility to run more processors on a single motherboard, the runtime will be improved further in the near future, even on inexpensive hardware.

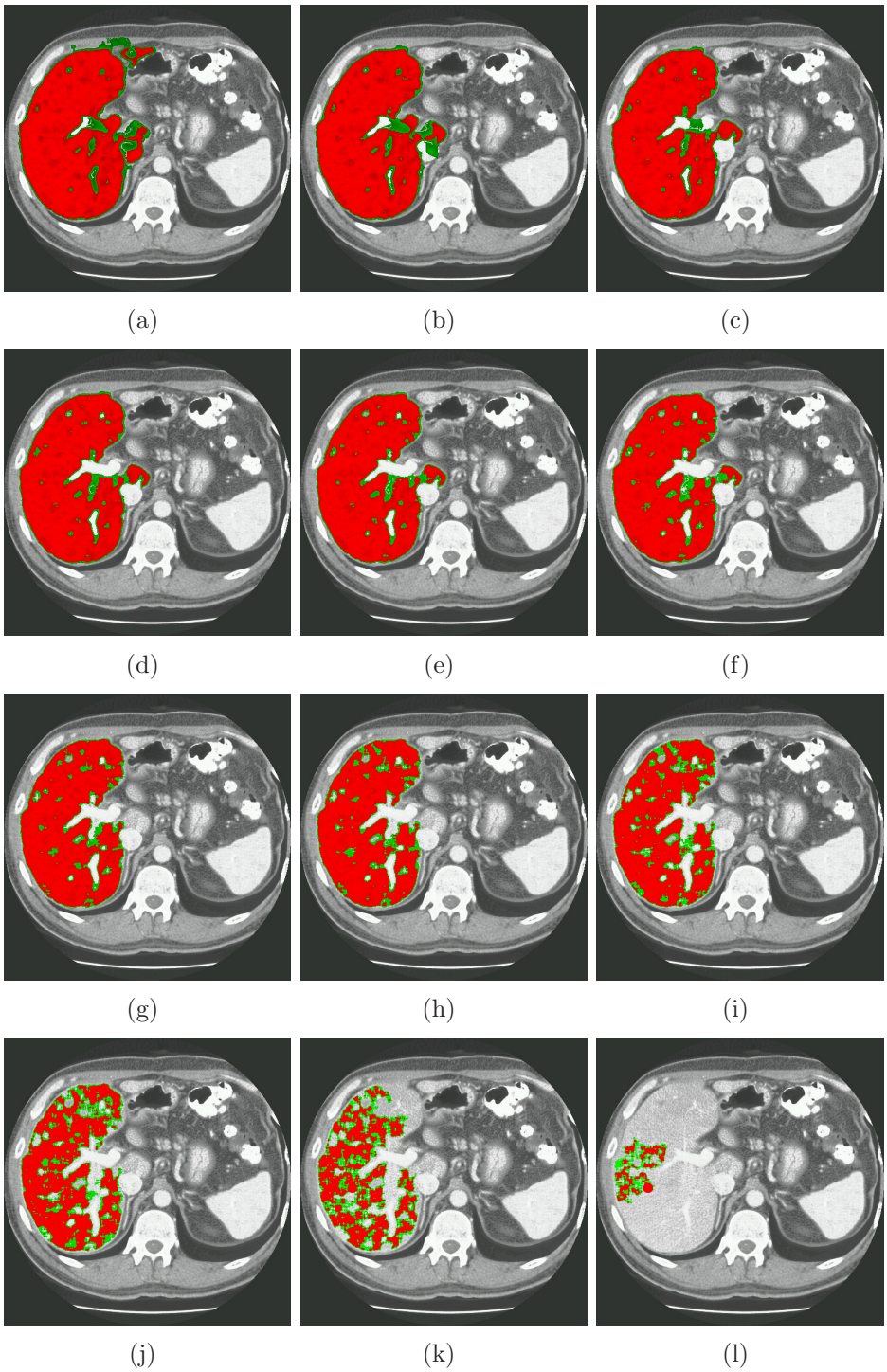


Figure 4.14: Segmentation results of a CT slice using varying threshold T. a) $T = 0.4$. b) $T = 0.45$. c) $T = 0.5$. d) $T = 0.55$. e) $T = 0.6$. f) $T = 0.65$. g) $T = 0.7$. h) $T = 0.75$. i) $T = 0.8$. j) $T = 0.85$. k) $T = 0.9$. l) $T = 0.95$. Red regions are regions that result in a $\dot{P}$ above both T and T_p, while green regions have $\dot{P}$ between T and T_p.

Chapter 5

Applications of texture based segmentation

In this chapter, we will apply the texture based segmentation method outlined in chapter 4 on hepatic vessels, liver tumours and kidney segmentation from CT scans.

Segmentation of hepatic vessels has previously been examined in chapter 3. As discussed, the given graph based vessel segmentation method has several weaknesses and numerous parameters that highly influence the result.

Once the liver is segmented, the vessels and tumours are excluded and we wish to use this result as the starting point in locating the hepatic vessels and possible tumours. In CT scans, hepatic vessels have a higher intensity than the liver, and such regions could be used as training sets for the vessel segmentation. An alternative is to let the user select a seed point on one of the larger vessels manually. Tumour texture is darker than the liver tissue, and the shape of tumours have previously been represented by a simple function since the general shape is known (Soler et al., 2001). Tumours may, however, have non-typical shapes and we wish to identify such tumours as well.

Finally, we will use texture based segmentation applied on kidney segmentation from CT scans. The difference between the kidney texture and surrounding tissue is more distinct than in the case of liver segmentation, making the kidney segmentation likely to be more robust.

5.1 Method

In the following subsections we will describe the procedures we developed to segment the hepatic vessels and locating possible tumours from the texture based segmentation of the liver. Kidney segmentation will also be presented

in the final subsection.

5.1.1 Hepatic vessel and tumour segmentation

The method for hepatic vessel segmentation is similar to what was described in chapter 4. However, some changes were made due to the number of border voxels, which is substantial in vessel segmentation compared to segmenting a larger more homogeneous organ. It is important that these border regions not influence the vessel classifier in a significant manner. We chose also to base the segmentation largely on the previous segmentation results of the liver.

A seed point was chosen, and the vessels were segmented using a low threshold $T = 0.3$ although a high threshold $T_p = 0.95$ such that border regions would not be added to the vessel classifier and thus influence the end segmentation result. After segmenting the vessels, a comparison between the liver segmentation and the hepatic vessel segmentation was performed. Here, a voxel was classified as hepatic vessel tissue if its co-occurrence distribution was more similar to the surrounding vessel distribution than the liver distribution.

The vessel segmentation result was finally masked with a morphologically closed (Soille, 2003) liver segmentation to present solely the vessels within the liver. A sound segmentation of the liver is thus required where the parameters are set appropriately. The vessel segmentation algorithm, however, does not need optimal parameter choices as long as all the vessels are found within the liver. This was the reason why the parameter T was set relatively low in the vessel segmentation compared to what was used in the liver segmentation procedure.

Once the liver and hepatic vessels were segmented, segmentation of possible tumours was performed using simple morphological operators on the remaining empty regions in the liver. The holes were morphologically closed with a small circular structure element (the radius was 4 voxels in our tests) and the remaining holes were compared to the average intensity of the surrounding liver. Segments that had noticeably lower average intensity than the liver were classified as possible tumours.

Results

The final segmentation result of the liver, hepatic vessels and a possible tumour can be seen in figure 5.1. Figures 5.2 and 5.3 show the hepatic vessel and tumour segmentation results in 2D. An additional complete hepatic vessel segmentation is presented from two angles in figure 5.4 and 5.5. Finally,

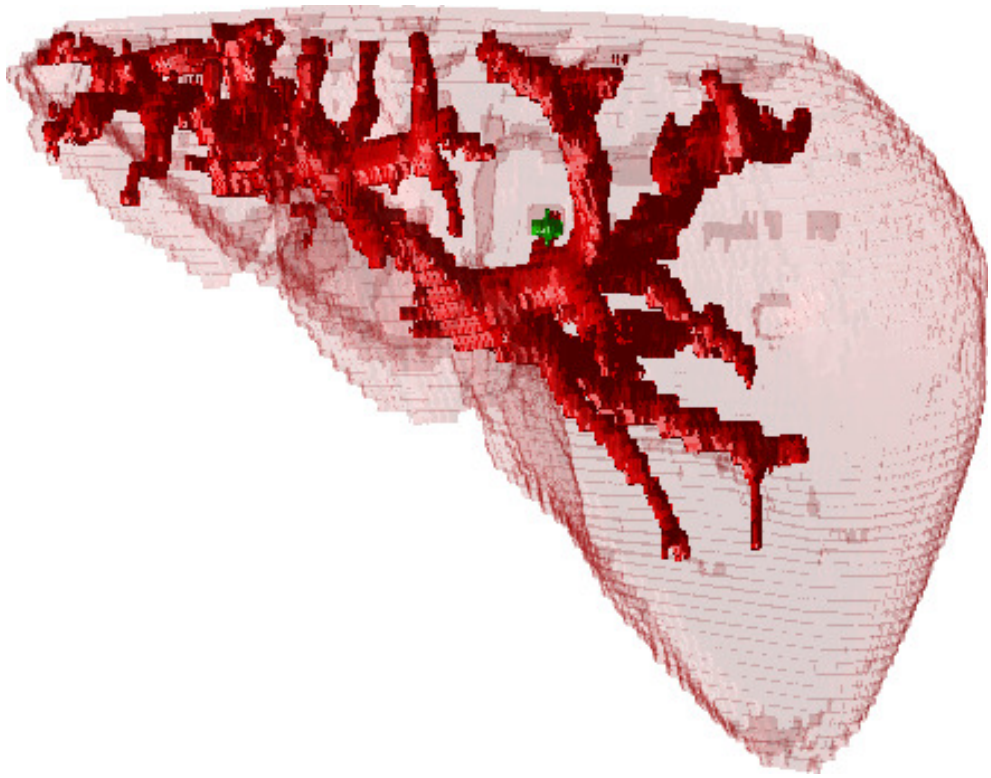


Figure 5.1: Complete segmentation result including hepatic vessels and a tumour. The tumour can be shown in green, and the liver is made opaque so that the interior is visible.

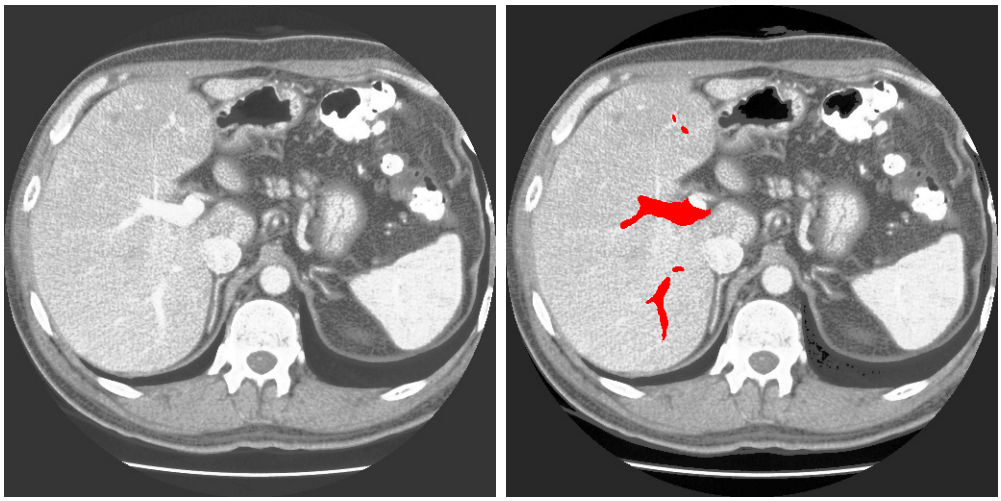
figure 5.1.1 shows a third liver segmentation with two tumours.

5.1.2 Kidney segmentation

The kidney has a compact shape and its interior has a homogeneous texture much like the liver. We used therefore the same approach as in the liver segmentation described in chapter 4 to segment the kidney.

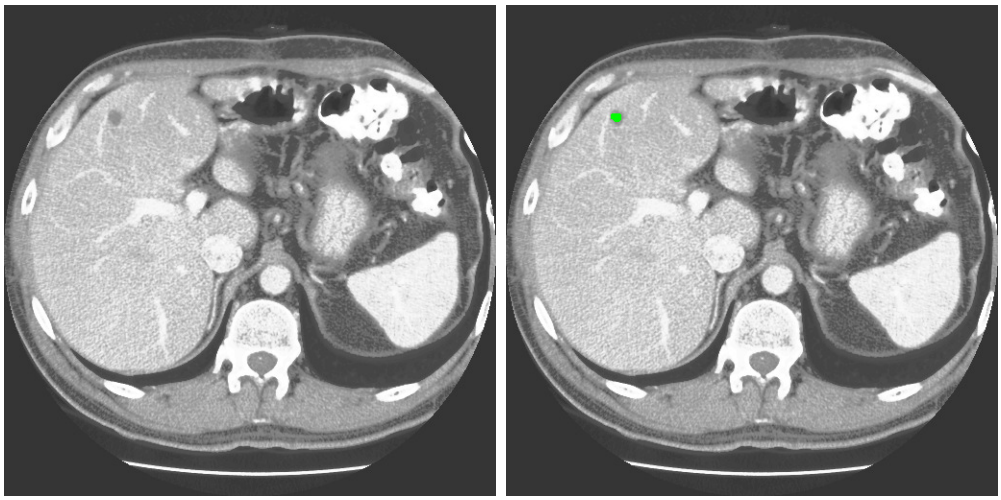
Results

Figure 5.7 shows an example segmentation of the left kidney. The same threshold $T = 0.65$ was used as in the liver segmentation. However, segmentation of the kidney is even more robust than segmentation of the liver and a wider range of T will yield acceptable results. Segmentation of both kidneys from a second CT scan is presented in figure 5.8.



(a) (b)

Figure 5.2: This figure shows an example vessel segmentation through the proposed method applied on a single CT slice. A seed point were used to train the initial classifier, which next was used to locate the remaining hepatic vessels.



(a) (b)

Figure 5.3: Liver and hepatic vessel segmentations were here used to detect the tumour (in green) through morphological operators.

5.2 Discussion

Compared to the graph based approach for hepatic vessel delineation described in chapter 3, the proposed texture based method has significantly

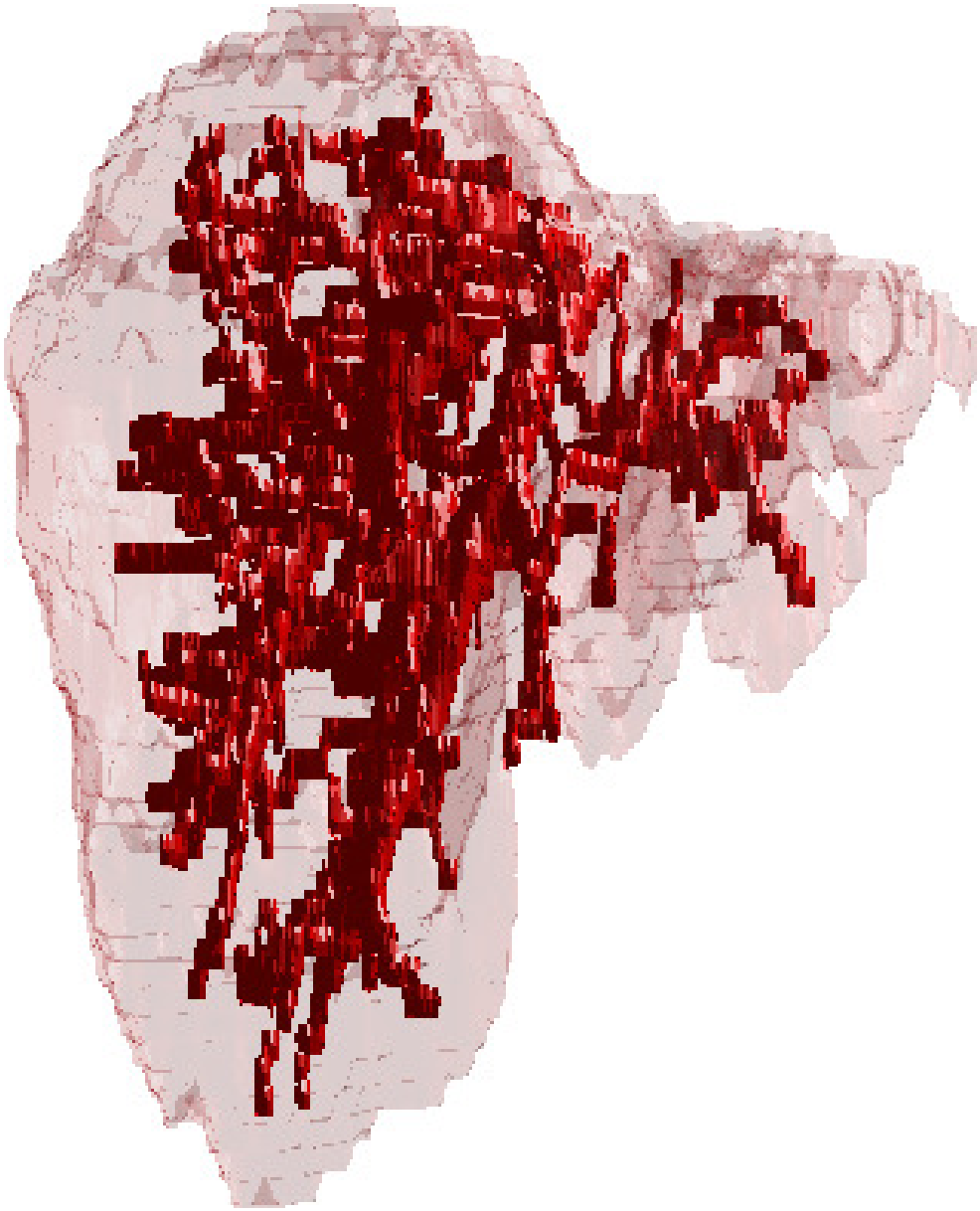


Figure 5.4: Hepatic vessel segmentation from a second CT scan.

fewer parameters and performs the segmentation directly on the CT or MR scan in a *region growing scheme*. This results in a more sound segmentation that is not highly dependent on parameter choices, and the method is general with respect to improved resolutions of the medical volumes. With adequate resolutions the need for improving the vessel segmentation through anatomical knowledge decreases and more direct approaches are sufficient.

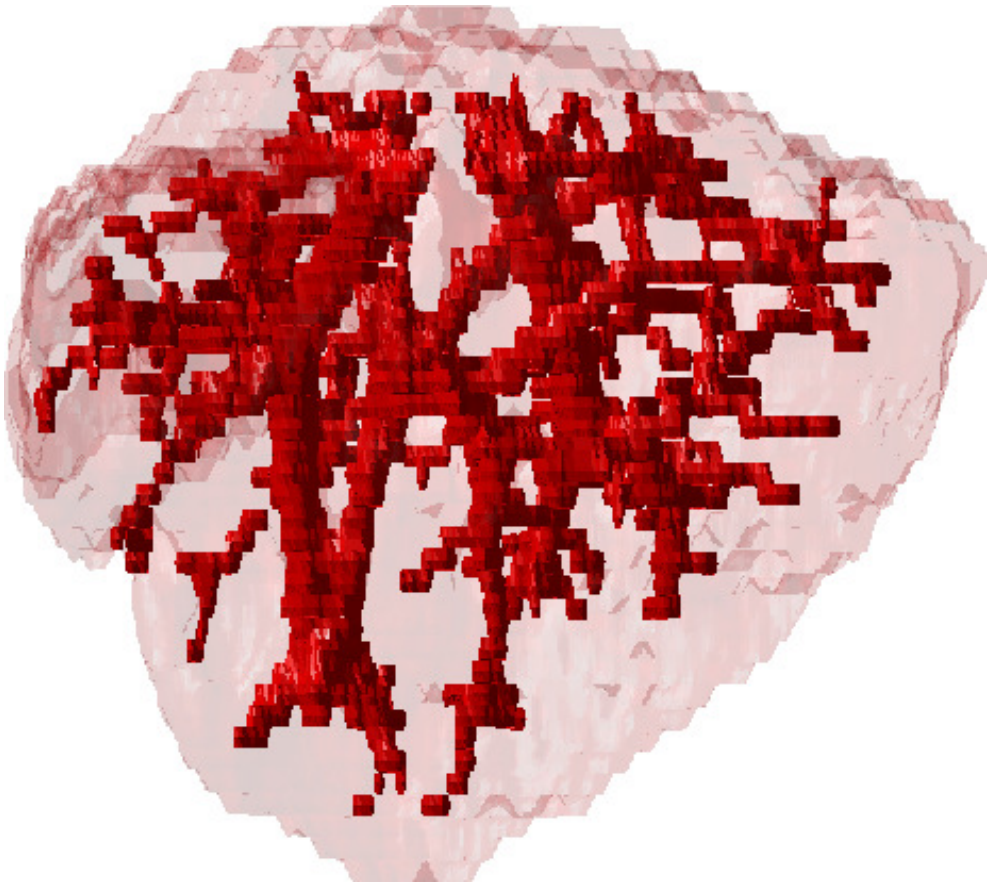


Figure 5.5: Hepatic vessel segmentation from a second CT scan shown from a different angle.

In Soler et al. (2001), the largest hepatic vessels were located through traditional region growing. The proposed method, however, is potentially less vulnerable to noise and changing texture of the vessels. Additionally, second-order statistics describe texture in more detail than using intensities directly. The threshold T in the vessel segmentation is highly flexible since the result is compared to the liver segmentation at the end.

This also applies to segmentation of possible tumours in the liver. We did not implement a shape criteria for the tumours since they can have abnormal shapes when, for instance, positioned close to the liver walls. Most liver tumours can be identified by comparing the tissue average contrast to that of the liver, however, it is difficult to create a general tumour classification algorithm based on texture alone.

Segmentation of the kidneys was also accomplished through the texture based segmentation method described in chapter 4. The difference between

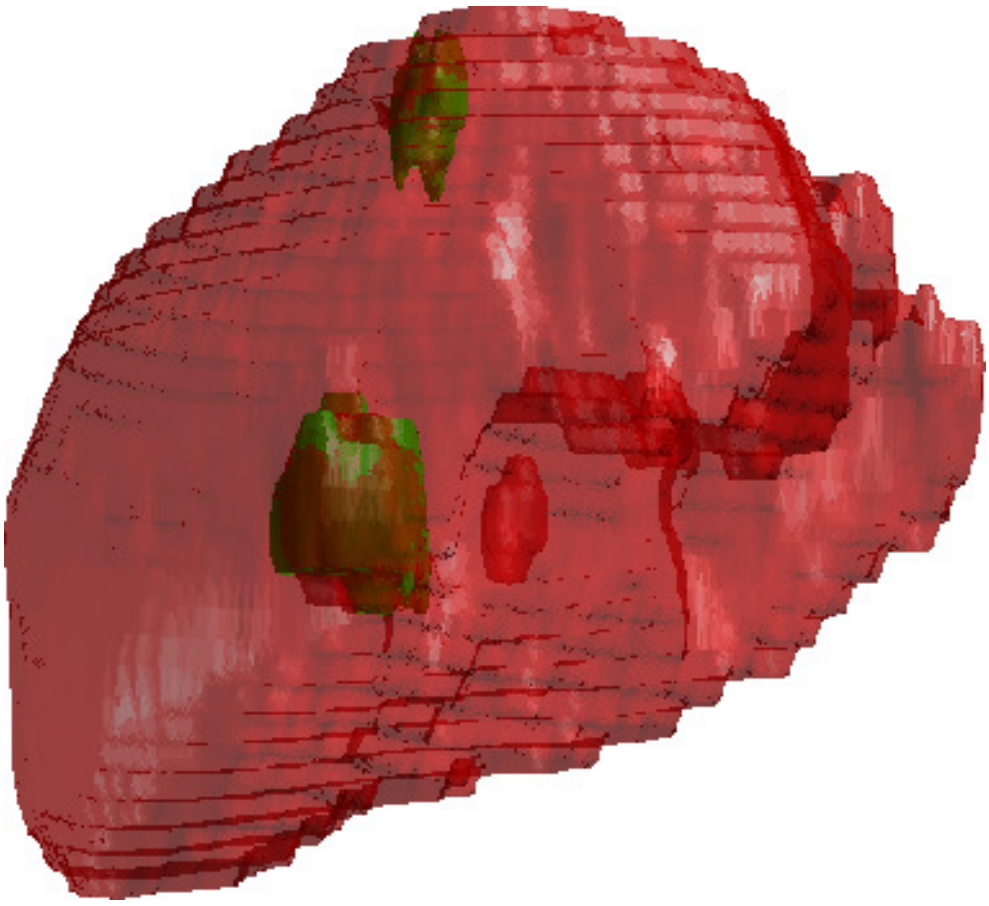


Figure 5.6:
Tumour segmentation of a third CT scan. The tumours are shown in green.

kidney segmentation and the previously described segmentation tasks was that normalisation of intensities and contrast of the CT scan were not available directly through, for instance, a previously segmented liver. Still, the spatial dimensions of the feature space ensures that the classifier adapts to varying kidney texture. The borders between the kidney walls and surrounding tissue were also significant, making the segmentation more robust than liver segmentation.

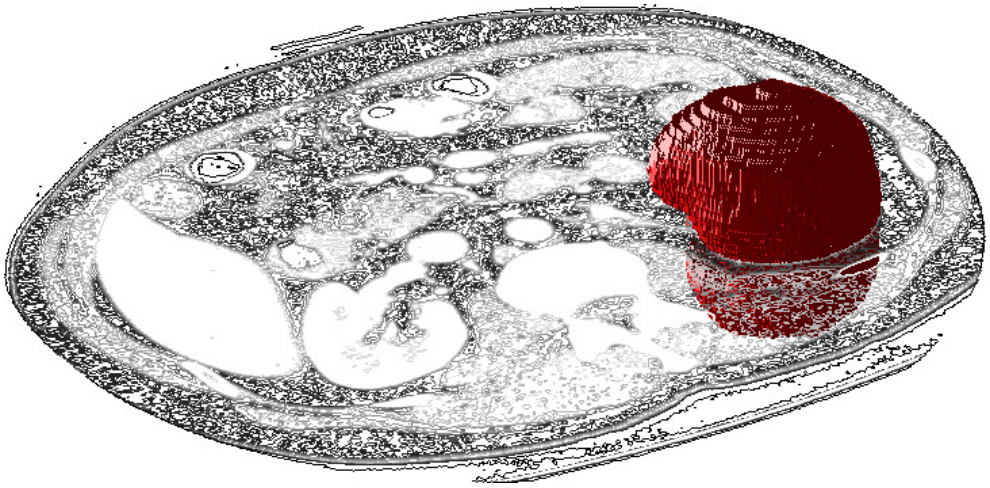


Figure 5.7: An example segmentation of the left kidney.

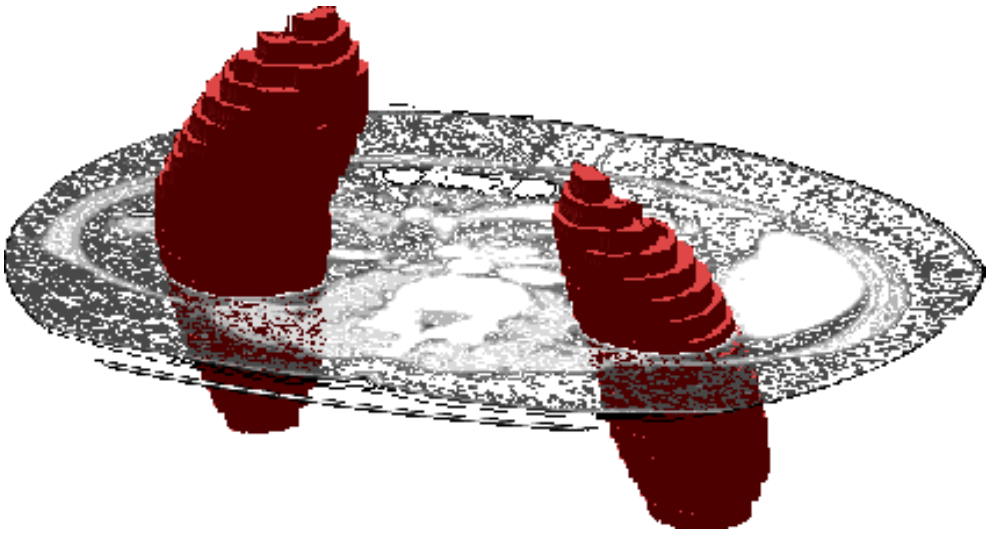


Figure 5.8: Segmentation result of both kidneys from a second CT scan.

Chapter 6

A new general representation of complex shapes to automate texture based segmentation

To be able to classify already segmented objects is important in especially two specific cases. First, if only one known object is segmented, verification of the object may be needed where you compare the result to statistical variations of the object in question. Second, if an unknown number of objects have been segmented by an automated segmentation method, classification is needed to label the various objects or groups of objects.

Focusing on segmentation of the liver, we need to verify that the result is viable according to anatomical variations of the liver. If the result varies significantly from the statistical model, parameters may be adjusted and other improvements may be performed. The challenge, however, is to make a meaningful representation of the liver shape that is as simple as possible but at the same time can be used to distinguish the various liver shapes.

Many previous articles use simple shape characteristics to describe shapes. An example can be found in Soler et al. (2001) where elongation and compactness of objects were used to classify lesions. Such descriptions may work well with simple shapes, but complex shapes need a more complete representation to separate them adequately. A promising way to represent more advanced organ shapes was used in Kelemen et al. (1998); Kelemen and Székely (1999); Székely and Gerig (2000), where corresponding points were evenly spread on the surface of the organs. This made it possible to represent shapes as points in a hyper-dimensional space[1] and thus in a simple manner

[1]Example: 3 3-dimensional points $[p1, p2, p3]$ become a 9-dimensional point if you represent them in vector form $[p1_x, p1_y, p1_z, p2_x, p2_y, p2_z, p3_x, p3_y, p3_z]$.

derive new models representing the statistical variance of the organs. Finding these corresponding points may be straightforward on similar objects, but as the objects differ more it becomes increasingly more challenging. This problem is overcome in the representation presented in Tsagaan et al. (2002). Instead of using points to define the surfaces, NURBS surfaces were used that are defined by a set of control points. Even though this representation may solve the major difficulty with the surface points representation, it is not general enough to represent complex objects such as organs with an interior, and the represented shape needs to be simple to limit the number of control points. In addition, it may not always be a simple task to derive the NURBS surface from a segmented object even when disregarding the interior.

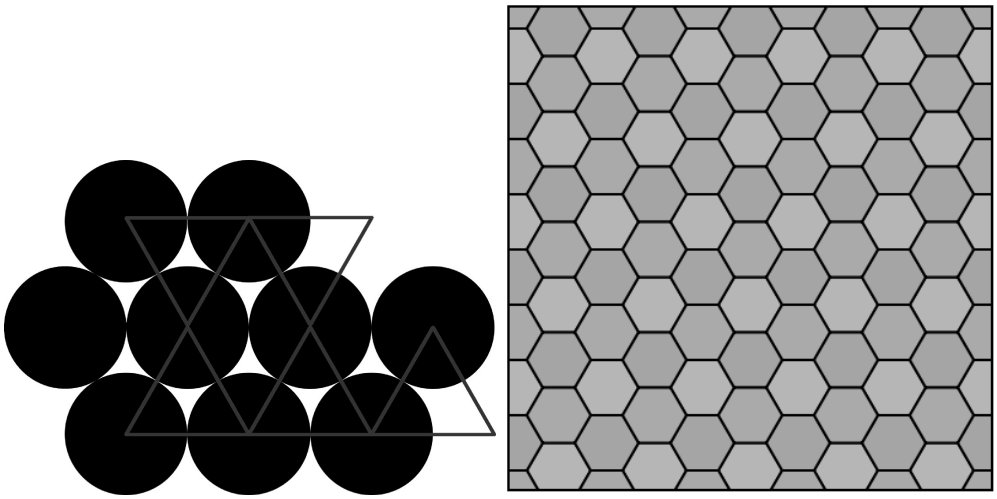
We have sought a representation that is general and can be used to represent any complex shape or groups of shapes. The representation needs to be simple to be derived from any given object and be able to produce new shapes. Additionally, the representation must scale well with any object resolution and be represented as a point in a hyper-dimensional space such that well known classifiers can be used to cluster and classify the represented objects.

The main motivation behind this representation is to incorporate statistical knowledge to the texture based segmentation algorithm from chapter 4 such that it operates fully automatically and the most appropriate threshold T is found. Segmentation results should be compared to a statistical template base, and the best fit result returned as the final segmentation result.

6.1 Method

We will present a new representation of one or more segmented objects that is simple and general for objects up to 8 dimensions. The representation can be interpreted as a point in hyper-dimensional space, and previously derived classification methods can be used to cluster and classify the represented objects. It will also be shown how the representation can be used to generate new objects to create a smaller template base representing the variation of a larger dataset.

In 2D, the representation can be seen as lowering the resolution of the area that contains the objects to be represented. The area is subdivided into equally sized and shaped partitions that contain an intensity value representing the intensities of the voxels that are included in each of the separate partitions. We can then represent the objects as a point where the number of partitions is equal to the number of dimensions of the hyperspace. The location of the point is decided by the intensity value of the partitions.



(a) (b)

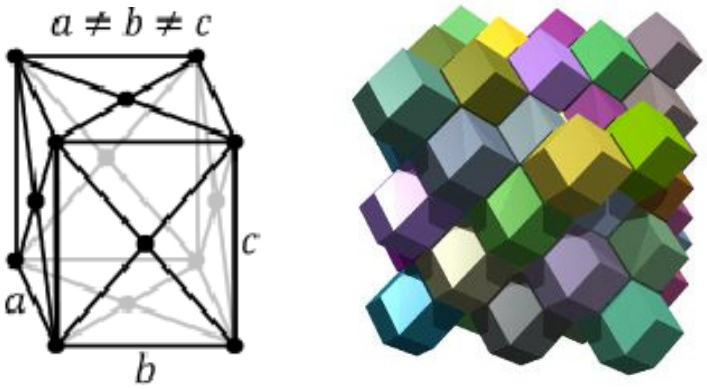
Figure 6.1: The 2D partition shape (b) is derived from the Voronoi diagram of the densest packed circle setup shown in a).

The average intensity value of the partition is used directly as the intensity value of the partitions, or in other words the location of the representative point in hyperspace.

A normalisation step is typically needed before computing this representation as it performs no normalisations in itself. An example would be to centre the objects with respect to the mass centre and rotate the principal axes of the objects to a predefined angle. These normalisation steps may, however, vary according to the problem at hand.

6.1.1 Partition shape

If one wants to compute the average of a region in an image and only the region centre and size are given, a circular region is the optimal choice since the result is then invariant to rotation. However, one cannot pack several circles into a closed region without having overlapping circles. The tesselation or the partition shape of a given dimensional space can be derived from a Voronoi diagram of the densest packed circle, sphere or hyper-sphere setup and its centre (Mathworld, 2007). In 2D, this tesselation is derived from a regular triangular lattice, and yields a hexagon tesselation (Wikipedia, 2007) as shown in figure 6.1. Similarily, in 3D, the tesselation is derived from a face-centred cubic lattice and the result is a rhombic dodecahedron partition shape (Wikipedia, 2007) as shown in figure 6.2.



(a) (b)

Figure 6.2: The 3D partition shape (b) is derived from the Voronoi diagram of the densest packed sphere setup shown in a). The black dots at the cube corners and face centres correspond to the sphere centres in a).

Compared to a square partition shape, the proposed partitions are more invariant to rotation, and the maximum radius of each partition is smaller. Furthermore, these two advantages become increasingly apparent as the number of dimensions increases.

If the number of dimensions exceeds 8, the most compact way to pack identical hyper-spheres is unknown. This would result in a suboptimal partition grid where the partitions are irregular.

An important advantage of using these compact partition shapes is that the distance to the centre of neighbouring partitions is always equal. This makes the representation as accurate as possible given a partition size, and it results in an optimal matching between two objects as well as an improved reconstruction from representation domain to spatial domain.

A visualisation of the partition shapes in 2D and the corresponding centre intensity values are shown in figure 6.3.

6.1.2 Validation

From the proposed representation, we can easily validate new segmentations with respect to previously verified segmentations. This can for instance be achieved by using the Parzen window method to estimate a segmentation distribution. Probabilities can then be derived directly from the distribution giving a good estimate of the correctness of new segmentations. Other classifiers can be used as well that, depending on the implementation, do not require the memory usage or the processing power of the Parzen win-

(a) (b)

Figure 6.3: This figure shows the partition shapes in 2D and the intensity value of the partition centres in b) computed from a). The intensity value equals the average intensity of a partition.

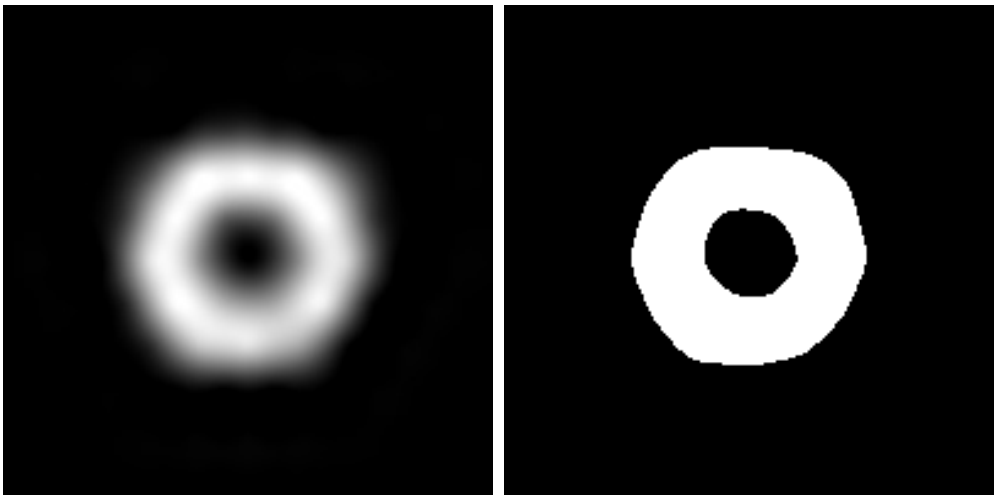
dow method. Examples are artificial neural networks and support vector machines.

6.1.3 Object reconstruction

It may be of interest to derive a few templates representing the statistical variations of previously segmented objects. These templates could for instance be used to initialise a segmentation of a new dataset such as described in Székely and Gerig (2000).

To derive such a statistical shape from the previously outlined representation, we suggest a three step procedure. First, a point is derived from the representation domain using given statistical descriptors, and the point is transformed to the spatial domain. Second, use the spline interpolation technique given in Sandwell (1987) to derive the voxel intensities throughout the spatial domain. Finally, assuming intensity values between 0 and 1, the result is thresholded and voxel values larger or equal to 0.5 are set to $true$[2]. In this way, given an adequate partition size, any complex shape can be both represented and reconstructed with an acceptable level of detail. See figure 6.4 for an example interpolation and result of the description shown in figure

[2]The original volume we want to reconstruct is binary and we thus set the voxels from the interpolation result that are closer to 1 than 0 to $true$.



(a) (b)

Figure 6.4: An example reconstruction from the shape in figure 6.3. The spline interpolation algorithm from Sandwell (1987) was used and the result was thresholded such that pixels with values ≥ 0.5 were kept.

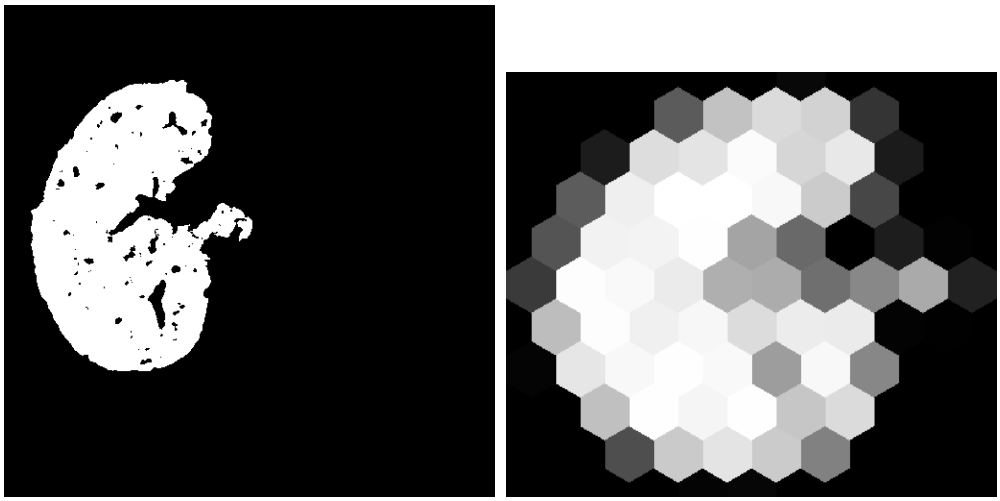
6.3.

6.2 Results

We applied the proposed representation on the liver segmentation results found in figure 4.14, and compared them to a template consisting of a previous segmentation of the same CT slice. The statistical template can be viewed in figure 6.5, where a partition grid of size 10×9 was used, and the segmentation results were resized to fit this grid. The representation was thus invariant to the segmentation size and location in the CT scan. The normalised dot product was used to compare the segmentation results in 4.14 to this template. Arguments in favour of the chosen measurement were presented in section 4.1.2.

Figure 6.6 shows the resulting graph of the normalised dot products and the highest values of T represents the best segmentation results given the statistical template. The segmentations with best fit are the results where the threshold T is set to 0.55, 0.6 and 0.65. These three segmentation results gives a sound segmentation of the liver from this particular CT slice as shown in figure 4.14.

As a second example, we used the representation in 3D ($6 \times 4 \times 6$ partitions) to find differences in grey matter of brain MRI of 14 year old subjects.



(a) (b)

Figure 6.5: The statistical template used to find the best threshold T in the texture based liver segmentation. a) Original segmentation. b) Representation of a) using the proposed representation. The representation was invariant to segmentation size and location in the CT slice.

54 segmentations from normal subjects were first rerepresented in the given model as a statistical model of grey brain matter. 51 prematurely born subjects were next compared to the normal subjects through the normalised dot product to identify variations from the statistical model (see figure 6.7). All the normal subjects were used in the comparison and the highest resulting dot product was used to characterise the statistical variation. We also compared the normal subjects to each other in the same manner (see figure 6.8). The grey matter segmentations were normalised with respect to rotation, scale and translation prior to applying the proposed representation method.

As the figures show, the resulting comparisons are generally better for the normal group than the premature group. This makes it plausible to believe that the grey matter in prematurely born subjects is different from normal subjects even at the age of fourteen.

6.3 Discussion

The proposed representation can describe any complex shape given an adequately small partition size and can describe shapes of up to 8 dimensions. Moreover, since the representation can be theoretically viewed as a point in hyperspace, the representation can be utilised by most previously derived

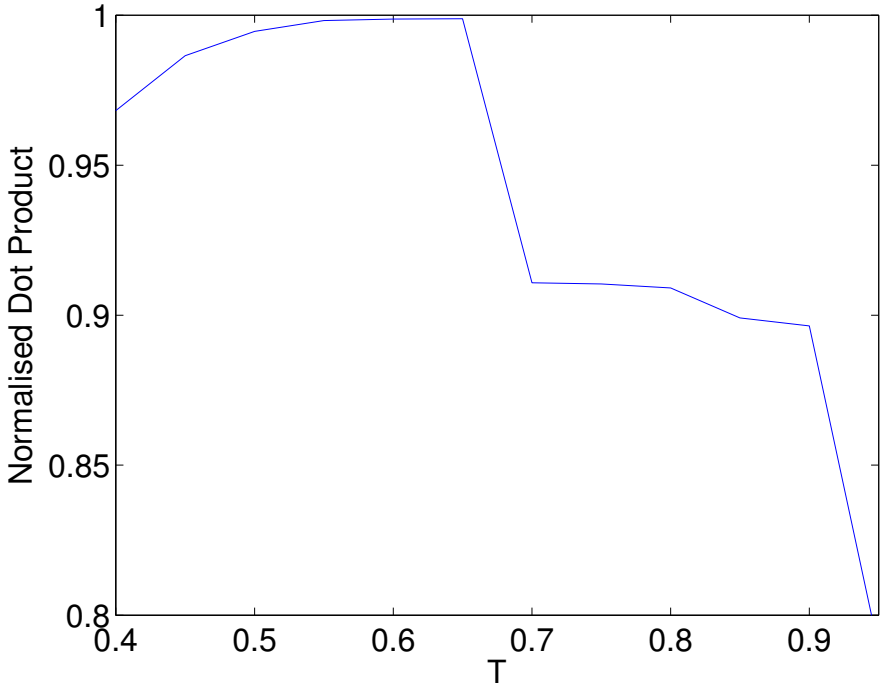


Figure 6.6: The resulting normalised dot product in the comparison between liver segmentation results with different thresholds T and a previous segmentation result of the same CT slice. The proposed representation was used to compare the various results.

classification methods making this representation flexible and available to many image processing tasks.

Compared to simple downscaling of an image using a rectangular grid, this representation is more compact, especially when regarding higher dimensional data. The distance to the closest neighbours is always the same, and since the partitions are more compact, fewer partitions can be used to represent the given objects at an adequate detail level.

Figure 6.4 provides an example in which a good reconstruction is made even when the number of dimensions is low. Generally speaking, the number of dimensions can be higher using the proposed representation than representations previously used to, for instance, represent human organs from CT and MR scans. To get the robustness needed in medical imaging, however, this significant number of features is required to distinguish organ shapes adequately. Even though the number of dimensions can pose a problem, the test data to describe an organ seldom exceed a few hundred datasets, and

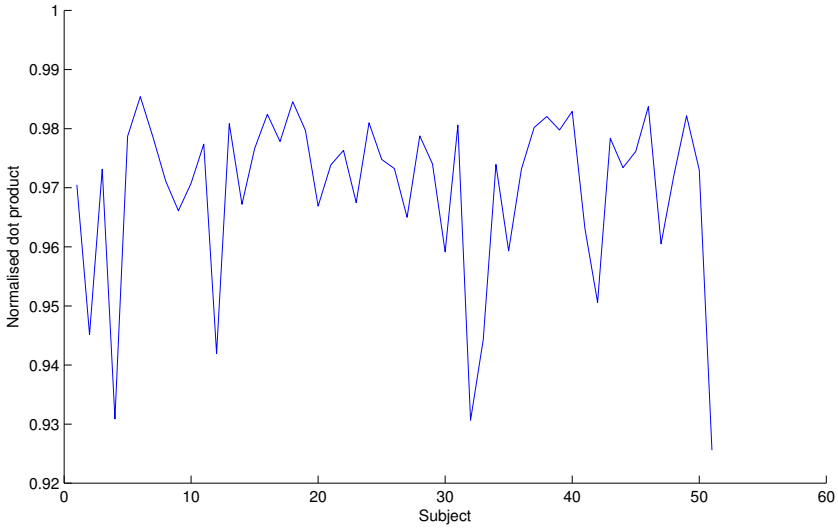


Figure 6.7: The result of comparing the grey matter in 3D from brain MRI of prematurely born subjects to normal subjects.

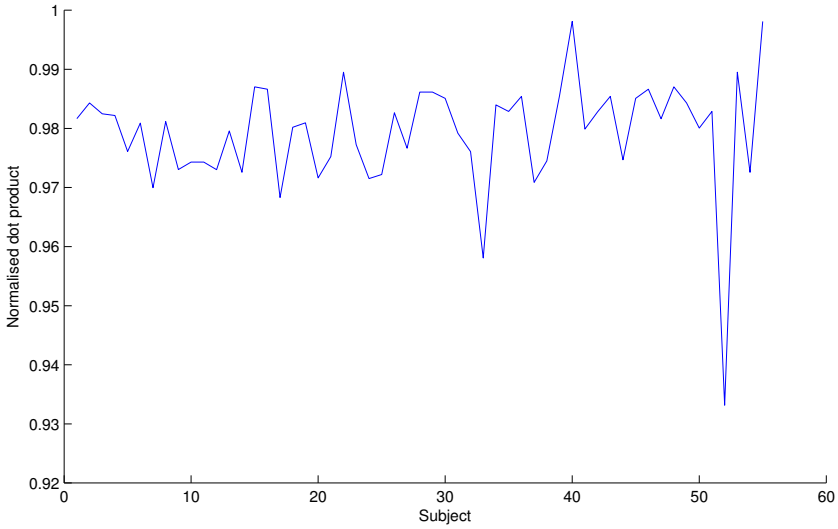


Figure 6.8: The result of comparing the grey matter in 3D from brain MRI between the normal subjects.

the processing time will then not be exceedingly affected by the number of dimensions. Additionally, there will most likely be a few empty partitions

throughout the dataset that can be disregarded in the processing.

A possible strength of the proposed representation is the potential to sort the training data by partitions. For instance, it is possible to sort out the training sets that are likely to have no relevance to a new set by comparing only a few central partitions. Another possibility is to incorporate the relevant datasets during a segmentation of an organ. In the case of segmentation through region growing, one could base the statistical influence by training sets that share common properties with the subsequent segmentation result.

The representation can also be used to store higher dimensional representations that later can be projected onto a lower dimensionality for matching that can be performed directly in the representation domain. An example usage could be matching a liver slice from a single CT or MR image and finding a good estimate of the location of the slice with respect to the liver shape. Another example, although outside of our problem domain, is classification by matching objects from an image to 3D templates and search for the best fit.

An example use of this representation was applied to the texture based segmentation method outlined in chapter 4, where the threshold T was automatically selected by comparing the results to a statistical template base. A 3D example was also given where segmentations of grey matter from brain MRI of prematurely born subjects were compared to normal subjects in order to find differences between the two groups.

Chapter 7

Parallel image processing using GPU

Another challenge arises during for instance ultrasound analysis where the user is presented with a video stream and analysis of the stream is performed concurrently. Here, image analysis tasks must be run in real-time and the delay between the input devices and the output devices must be minimal.

Since image analysis techniques generally are resource demanding, a cluster of computers is typically needed to perform all the operations necessary in a medical application in real-time. This solution is often impractical, however, recent advances in GPU (Graphics Processing Unit) performance on inexpensive graphic cards has made it possible to solve very complex image processing tasks efficiently.

We have looked at two ultrasound applications. The first application consists of extracting the left ventricle walls for ischemia diagnosis. By locating the ventricle walls during heart cycles, analysis of the heart movement can be performed. In the second application, we automatically detect lesions in the liver. The purpose here is to more accurately determine the positions of previously detected lesions. These lesions are typically found prior to surgery through CT or MR scans.

This work was published in Eidheim et al. (2005).

7.1 Previous work

Programmable GPUs became available in 2000 to let developers get more control over the rendering pipeline on graphics hardware. The main motivation was increased performance, and the technology has been pushed forward by the gaming industry.

Modern GPUs are highly optimised for *stream* computations as described in Venkatasubramanian (2003), and using the GPU as a *stream co-processor* is gaining popularity as it often outperforms the CPU for such tasks. In addition, todays GPUs also include *spatial parallelism* in that each pixel on the screen can be seen as a stream processor. This results in a high degree of parallelism when used appropriately.

According to Strzodka (2004), GPUs may increase the speed of the algorithms previously run on the CPU by more than 10 times. Furthermore, the development of GPUs does not follow Moore's law, but advances even more rapidly than the development of CPUs.

Several advanced image processing methods have been implemented using GPGPU (General-Purpose computation on GPUs) for speed improvements. Some examples can be seen in Strzodka (2004), including segmentation by the level-set method, the Hough Transform, and motion estimation by eigenvector analysis of spatio-temporal tensors. Another example is segmenting the brain surface from an MRI data set by the level-set method Lefohn et al. (2003).

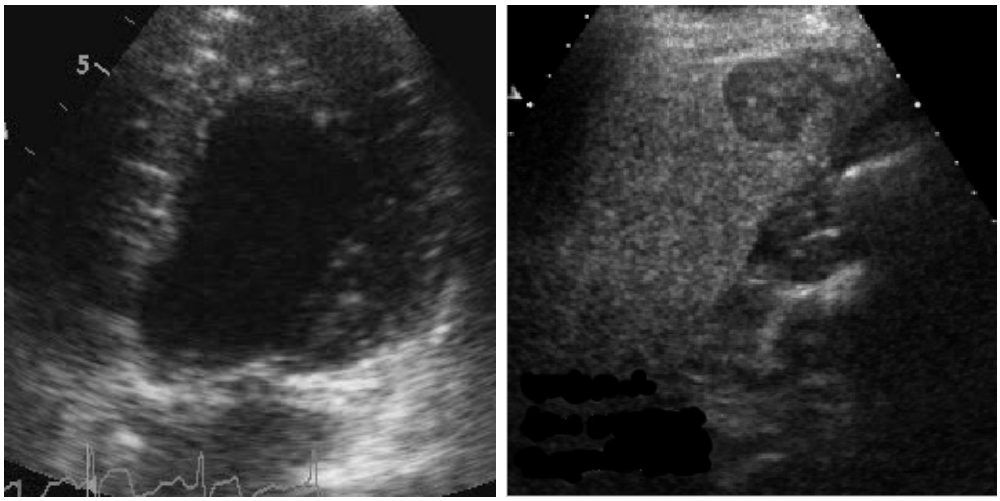
7.2 Methods

All operations described in this section were implemented on the GPU. The image processing tasks were implemented on the fragment shader using Nvidia's Cg programming language in combination with C++ and OpenGL. The graphics card used was a GeForce 6600 GT on a 2.8 GHz Intel Pentium 4 computer.

The two most computationally expensive stages in medical image analysis are frequently preprocessing and segmentation. The procedures are typically iterative, and calculations are commonly executed for each pixel in the image. Fortunately, one procedure is often repeatedly executed on each pixel for each iteration. This matches the stream computation model and spatial parallelism in GPGPU.

As seen in figure 7.1, the borders of the desired regions may be challenging to find. In the first image, showing a left ventricle, the motivation is to locate the ventricle walls in order to calculate the ventricle volume for ischemia diagnosis. Similarly, in the second example, the goal is to segment the tumour in the liver.

During the preprocessing step in both applications, grey level mathematical morphology Soille (2003) was used in to reconstruct the fuzzy borders. Since the regions to be segmented were darker than the surrounding tissue, we first used grey level morphological dilation in order to make the borders



(a) (b)

Figure 7.1: This figure shows two examples of ultrasound images to be processed and segmented. The border of the target regions may be fuzzy and undefined. a) Ultrasound image of a left heart ventricle. b) Ultrasound image showing a liver tumour.

physically larger. After this, grey level morphological erosion was computed to reduce the borders to original size, although, as a result some border elements are now connected. The described processing steps also corresponds to what is commonly called a morphological closing Soille (2003).

The implementation of grey level morphological operators on the GPU is similar to the implementation of convolution on the GPU as described in Rost (2004). Morphological operators are based on morphological erosion and dilation, which is simply local minima and maxima filters over specific domains respectively. However, in order to run these operators effectively on the GPU, one has to take advantage of the built-in min and max functions of the shading language. This is due to the fact that branching, e.g. if-branches, are very expensive operations on the GPU.

After having implemented the two basic morphological operators, namely erosion and dilation, a wide range of morphological methods can be run easily. A few examples are thinning, skeletonising, thickening, geodesic erosion, geodesic dilation, reconstruction, and the distance transform Soille (2003). Still, morphological reconstruction can be more efficiently implemented on the CPU using the algorithm described in Vincent (1993).

After the borders of the desired regions have been emphasised, the next step is to apply a segmentation algorithm. Our method of choice was the

active contour model described in Kass et al. (1988). This model is popularly called the snake model, and is basically a set of connected nodes that are moved according to internal snake forces as well as forces external to the snake arising from the image itself or from specific user constraints. The internal forces are commonly proportional to the first and second spatial derivative, representing tension and rigidity of the snake respectively. Image and external forces are typically forces set to guide the snake towards desired regions of an image. An example of image forces is the image gradient, while balloon forces Cohen (1991) is an example of external forces.

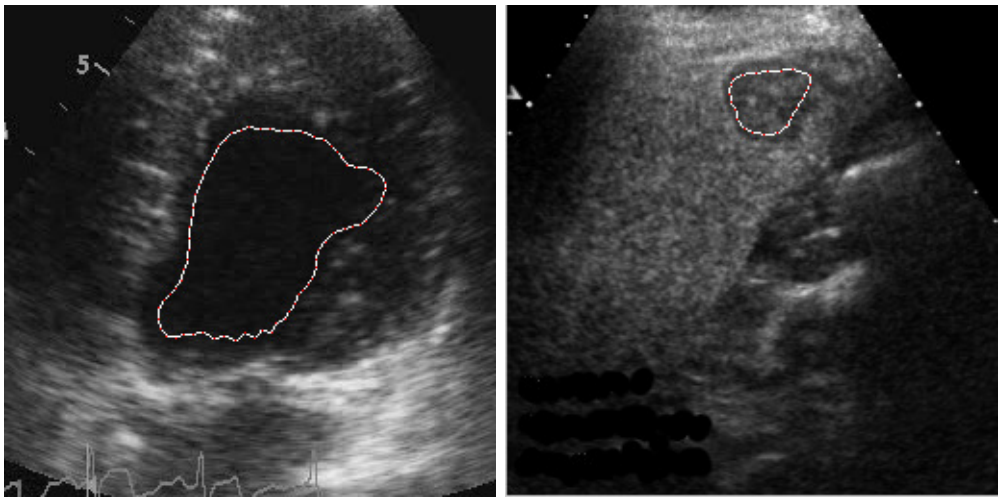
To attract the snake towards the left ventricle walls or the walls of the tumour, we need to compute forces from the image that will achieve this. Our solution was to calculate the gradient vector flow Xu and Prince (1998, 2000) of the ultrasound images. This method attracts the snake towards edges with increased capture range, and attracts the snake into concavities. The process of calculating the gradient vector flow, however, is time-consuming and resource demanding. This is due to the need of an algorithm that has to be run for each pixel iteratively. This led to an implementation on the GPU, which greatly reduced the processing time of the procedure.

The general position of the target regions of both applications is assumed known, either from centring the ultrasound device on the left ventricle, or from previous segmentations of the liver tumours from CT or MR scans. Thereby, we used this general position, assumed to be inside of the desired region, to create a small initial snake. By the means of balloon forces Cohen (1991) we then expanded the snake until stability, and the snake's final position was used to segment the region.

7.3 Results

The most time-consuming procedures of our methods, are the preprocessing steps. By implementing these procedures on the GPU, we received a running time reduction of 34 times when processing 512x512 images compared to the CPU. The runtime of the whole procedure is 54 ms (frame rate of 18.5 fps), and thus the processing steps outlined in the previous section can be run in real-time using the GPU. Final segmentation results can be seen in figure 7.2.

Figure 7.3 shows the runtime of GPU programs compared to CPU programs using our setup. If the number of elements are lower than approximately 24^2, it is more efficient to run the procedure on the CPU. In our applications, the number of nodes in our snake model did not exceed this number, and we therefore chose to run the snake model on the CPU instead



(a) (b)

Figure 7.2: The final segmentation results can be seen in this figure. a) Outlined left ventricle wall. b) Segmented liver tumour.

of the GPU.

7.4 Discussion

The segmentation results demonstrated in this chapter are of poor quality and dependent on parameters that are difficult to set appropriately. However, the motivation was to implement the algorithms on inexpensive hardware and demonstrate example results. A significant speed increase was shown compared to serial processing of the same algorithms.

Calculations on the GPU are not as exact as calculations on the CPU due to inherent difficulties with data types larger than 32-bit. In our applications, this drawback resulted in only minor artifacts, through we argue that this drawback will not significantly influence the end result.

A possible processing bottleneck is the AGP bus when moving data from CPU to GPU memory. The AGP bus is asymmetric, meaning data is sent faster to the graphics card than back. A solution to this problem is to some extent offered by the new PCI express cards that are full duplex and send data to the graphics card at twice the rate of 8x AGP cards. A data stream of up to approximately 4 Gbps is attainable on this bus presently.

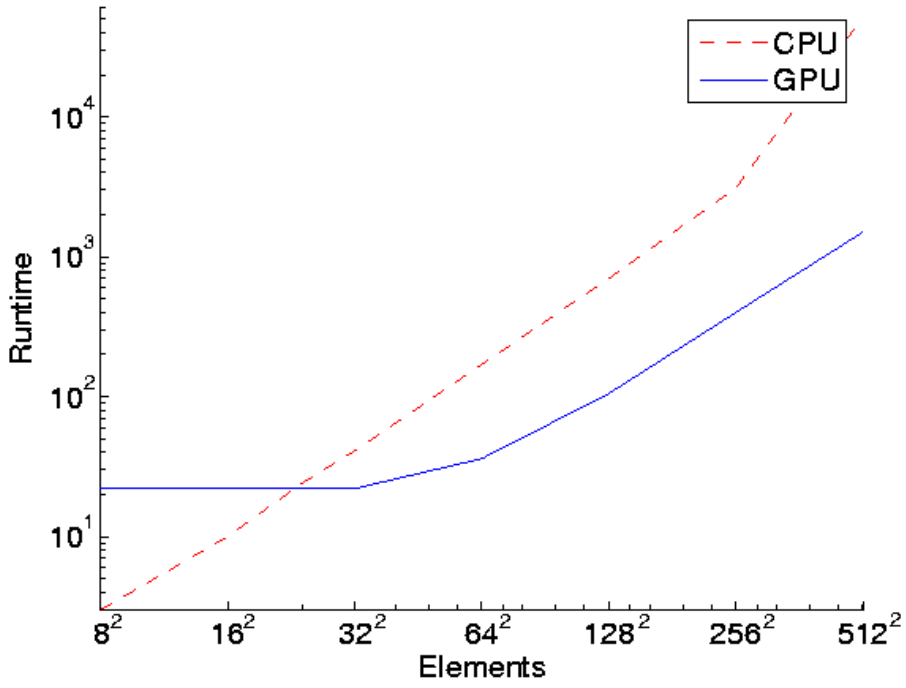


Figure 7.3: This figure shows a comparison of procedures running on the GPU and the CPU. The number of iterations run on each test were 2400 (100 iterations 24 times a second). If the number of elements exceeds 24^2, the processing time of the GPU is less than that of the CPU. For instance, the processing time of an image of size 512x512 is 34 times faster on the GPU than the CPU on our test platform.

Chapter 8

Conclusion

After studying a large number of previous articles on segmentation of organs from CT and MR scans, we conclude that presently there exists no robust method to solve the segmentation tasks focused on in this thesis. A common factor with all the previously proposed methods is that they are dependent on several inflexible parameters that need to be fine-tuned for each particular segmentation task.

Parallel hardware is becoming common and inexpensive. This makes it possible to implement costly procedures to a higher scale than before. We based our work therefore on methods working on less simplified feature sets than previously published procedures.

The contributions of this thesis will be discussed in turn in the following sections.

8.1 Texture based segmentation

We have derived a new texture based segmentation method and demonstrated its use, in particular, on liver segmentation from CT scans. Results show that it is a robust method that is not greatly influenced by parameter choices. The new segmentation method was implemented in parallel using multiple CPU cores, and the processing speed is acceptable even though the number of computations is significant.

Based on the liver segmentation, segmentation of the hepatic vessels and liver tumours were presented. The direct hepatic vessel approach is more promising than previous work due to the low number of parameters needed to be set appropriately.

The proposed texture based segmentation algorithm was also applied to kidney segmentation with promising results. The kidney has more distinct

texture compared to the surrounding tissue and can therefore be more easily segmented than, for instance, the liver.

Compared to earlier popular attempts to separate tissues in CT and MR scans, the results from the proposed method show a significant improvement. We have not found any previously derived segmentation algorithm that is able produce similar results. As opposed to most earlier methods, the proposed procedure uses only a single parameter that is resilient and, in comparison, easily derived. Furthermore, the procedure is not explicitly based on the boundary of the organs, but the whole scan volume is exploited. On one example CT slice, $\frac{3}{20}$ of all possible parameter choices of T yielded sound liver segmentations. Moreover, a demonstration was made where this parameter was set automatically through comparison of the possible results to a statistical template.

Further improvement to the procedure is needed in order to apply it to medical applications. We do not therefore present a complete system to segment organs from CT or MR scans, but the method is a step further to separate different tissues automatically. Especially statistical templates can improve the segmentation, and make it even more robust and eventually ready for clinical use.

Furthermore, it is hard to accomplish a sound segmentation of an entire CT or MR scan with one set of parameters. The reason is that the texture can vary in intensity and contrast between the slices to a high degree. Possible solutions to this challenge are further normalisation of the slices, shape-based models restricting the end result, and post-processing of the end result. Another promising solution was presented, however, where several texture-based segmentations were performed subsequently. After each segmentation, the result was eroded with a structure element equal to the co-occurrence window, and the remaining voxels were added as seed points for the following segmentation process.

8.2 Iterative skeleton representation of 3D objects

A new iterative skeleton method was derived to create a more compact representation of objects in 3 dimensions. Compared to previous iterative methods we studied, a more compact representation was computed without known artifacts. Iterative procedures are not dependent on parameter choices such as procedures based on the distance transform, and now that a sound iterative skeleton method has been derived the proposed solution is a better choice for

computing 3D object skeletons.

A sound skeleton of the hepatic vessels can be used to perform automatic measurements and anatomical segmentation of the liver. Due to the lack of a working 3D skeleton algorithm, this has not been previously possible without extensive post-processing. The 3D skeleton algorithm has many more uses, and this method is a substantial contribution to the image processing community.

Previous 3D skeleton algorithms produced artifacts when processing certain voxel setups. Further testing must be executed on our skeleton algorithm to ensure that similar artifacts are not produced.

The proposed 3D skeleton method is general and works for 2D objects as well as 3D objects. It is likely that one can extend the method to objects above 3 dimensions, however, further research is needed in order to prove this.

8.3 Representation of complex shapes

A new general representation of objects up to 8 dimensions was derived to describe segmentation results from the texture based segmentation algorithm. In comparison with many previous representations, the proposed method can represent shapes with interior features such as holes. The representation is simple and general, and can form the basis of an even more compact representation where the interior is preserved. Moreover, the method has a sound mathematical foundation, ensuring compact representations of objects of higher dimensions.

A way to reconstruct objects from the representation was also derived, such as for instance reconstructions representing the statistical variance of a larger dataset. Since the representation can be viewed as a point in hyperspace, feature vector classifiers and clustering techniques can be applied directly.

An example use of this representation was demonstrated, where the threshold T in the texture based segmentation method was automatically chosen based on a previous segmentation result of the liver.

8.4 Minor contributions

A new and improved way to produce polygon meshes of branching structures was derived in collaboration with Jo Skjermo (Skjermo and Eidheim, 2005). We used this meshing method to visualise vessels in the graph-based

approach for hepatic vessel segmentation (Eidheim, 2005). In this way, the segmentation results can be visualised using common visualisation hardware that is based on polygon models. Slightly improved branching structures were produced using this procedure compared to previous algorithms.

The performance of the CPU and GPU was compared with respect to real-time segmentation of ultrasound images (Eidheim et al., 2005). Several image processing operators was implemented on the GPU with a substantial improvement to the processing speed. The texture-based segmentation method was also implemented multi-threaded on the CPU and a significant improvement of the processing time was shown.

Multiple cores on modern computers are getting more and more common. Since they are more flexible than the GPUs on graphic cards, it seems that CPUs represents the cheapest, most available and adaptable solution in general for parallel processing in the future. Nevertheless, the choice of parallel technology is highly dependent on the problem at hand and available resources.

A graph based approach for hepatic vessel segmentation was implemented and tested (Eidheim et al., 2004a,b; Eidheim, 2005). Results show, however, that more direct methods, such as through the texture based method, is not as dependent on parameter choices as the graph based approach is.

8.5 Discussion of research questions

We will here attempt to answer the given research questions on the basis of the previous discussion and conclusions.

- Is it possible to achieve a sound and robust segmentation of the liver, hepatic vessels, liver tumours and the kidney with previously derived image processing techniques?

  Based on the literature that has been reviewed, there is no current method that is sufficiently sound and robust for medical use. This also applies to the proposed solutions in this thesis. The anatomical variations make it difficult to derive automatic procedures with guaranteed success. Focusing on liver segmentation as an example, the liver can contain numerous and large tumours, and previous interventions may greatly alter the texture and shape of the liver. Even though humans can interpret the medical volumes well, it is challenging to create a complete mathematical method that takes into consideration anatomical knowledge to a sufficient degree.

- How can we best segment the liver, hepatic vessels, possible tumours and the kidney from CT and MR scans?

  The proposed methods for liver, hepatic vessels and kidney segmentation represents a significant improvement compared to previously published material. On the other hand, tumour segmentation was solely presented as another example use of our texture based segmentation algorithm, and previous shape-based methods may produce better results. Generally speaking, an incorporation of texture and shape based segmentation together with statistical models would most likely produce the best results for the given segmentation tasks.

- What statistical models can guide and verify the accuracy of the proposed segmentation techniques?

  The choice of representation depends highly on the problem at hand. Given a liver segmentation, the representation given in chapter 6 would also include interior features that a surface representation would disregard. Contrarily, a skeleton representation could be the logical choice for representing hepatic vessels, and an exterior shape model might be most suitable for tumour representation.

8.6 Future work

The following tasks correlated to this thesis are applicable for future research:

- Further improvement and testing of the texture based segmentation algorithm is needed. The proposed method can be applied to a number of related segmentation tasks.

- A large statistical base must be produced to improve the automatic segmentation of the liver, liver tumours and the kidney. The proposed representation of objects with an interior can be applied to create a template base representing the statistical variations of the organs.

- The segmentation of hepatic vessels can be improved through further development of the texture based segmentation algorithm and by applying anatomical knowledge using our 3D skeleton approach.

- Automatic measurements can be applied to the hepatic vessels results through the proposed skeleton method.

- Texture based segmentation can be guided by both statistical templates and physical constraints. We did not study this in detail.

- It is difficult to prove that our skeleton representation is most compact for all objects. A larger quantitative test should be made to try to find cases where the proposed skeleton procedure may fail. Earlier methods had special cases where the procedures yielded suboptimal results, but it is unknown if our method also fails in certain voxel setups.
- It is likely that one can extend the 3D skeleton method to objects having more than 3 dimensions. Further research is needed to prove this.

Bibliography

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: foundational issues, methodological variations, and system approaches. *AI Commun.*, 7(1):39–59.

Aykac, D., Hoffman, E. A., McLennan, G., and Reinhardt, J. M. (2003). Segmentation and analysis of the human airway tree from three-dimensional x-ray ct images. *IEEE Transactions on Medical Imaging*, 22(8):940–950.

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton, NJ.

Bertrand, G. and Malandain, G. (1994). A new characterization of three-dimensional simple points. *Pattern Recognition Letters*, 15(3):196–175.

Bevk, M. and Kononenko, I. (2002). A statistical approach to texture description of medical image: A preliminary study. In *Proceedings of the 15th IEEE Symposium on Computer-Based Medical Systems (CBMS 2002)*.

Blum, H. (1967). A Transformation for Extracting New Descriptors of Shape. In Wathen-Dunn, W., editor, *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, Cambridge.

Borgefors, G., Nystrom, I., and Baja, G. (1999). Computing skeletons in three dimensions. *Pattern Recognition*, 32:1225–1236.

Chaudhuri, S., Chatterjee, S., Katz, N., Nelson, M., and Goldbaum, M. (1989). Detection of blood vessels in retinal images using two-dimensional matched filters. *IEEE Transactions on Medical Imageing*, 8(3):263–269.

Choi, H. S., Haynor, D. R., and Kim, Y. (1991). Partial volume tissue classification of multichannel magnetic resonance images - a mixel model. *IEEE Transactions on Medical Imaging*, 10(3):395–407.

Chung, D. H. and Sapiro, G. (2000). Segmenting skin lesions with partial-differential-equations-based image processing algorithms. *IEEE Transactions on Medical Imaging*, 19(7):763–767.

Cohen, L. D. (1991). On active contour models and balloons. *CVGIP: Image Understand.*, 53:211–218.

Coiras, E., Santamaria, J., and Miravet, C. (1998). Hexadecagonal region growing. *Pattern Recogn. Lett.*, 19(12):1111–1117.

Comaniciu, D. and Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603 – 619.

Cootes, T. F., Edwards, G. J., and Taylor, C. J. (1998). Active appearance models. *Lecture Notes in Computer Science*, 1407:484–498.

Duda, R. O., Hart, P. E., and Stork, D. G. (2001). *Pattern Classification*. John Wiley & Sons.

Eidheim, O. C. (2005). Reconstruction of hepatic vessels from ct scans. Master's thesis, Norwegian University of Science and Technology.

Eidheim, O. C., Aurdal, L., Omholt-Jensen, T., Mala, T., and Edwin, B. (2004a). Interconnecting segmented hepatic vessels in adjacent ct slices. *NOBIM*, pages 91–97.

Eidheim, O. C., Aurdal, L., Omholt-Jensen, T., Mala, T., and Edwin, B. (2004b). Segmentation of liver vessels as seen in mr and ct images. *Computer Assisted Radiology and Surgery*, pages 201–206.

Eidheim, O. C., Skjermo, J., and Aurdal, L. (2005). Real-time analysis of ultrasound images using gpu. *Computer Assisted Radiology and Surgery*.

Fernández, G., Bischof, H., and Beichel, R. (2003). Nonlinear filters on 3d ct imaging - bilateral filter and mean shift filter. *Computer Vision - CVWW'03*.

Fok, Y.-L., Chan, J. C. K., and Chin, R. T. (1996). Automated analysis of nerve-cell images using active contour models. *IEEE Transactions on Medical Imaging*, 15(3):353–368.

Frangi, A., Rueckert, D., Schnabel, J., and Niessen, W. (2002). Automatic construction of multiple-object three-dimensional statistical shape models: application to cardiac modeling. *IEEE Transactions on Medical Imaging*, 21(9):1151–1166.

Frigon, N. L. and Mathews, D. (1997). *Practical Guide to Experimental Design*. Wiley.

Gao, L., Heath, D. G., Kuszyk, B. S., and Fishman, E. K. (1996). Automatic liver segmentation technique for three-dimensional visualization of ct data. *Radiology*, 201:359–364.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distribution, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAM1-6, No. 6:721–741.

Glombitza, G., Lamadé, W., Demiris, A. M., Göpfert, M.-R., Mayer, A., Bahner, M. L., Meinzer, H.-P., Richter, G., Lehnert, T., and Herfarth, C. (1999). Virtual planning of liver resections: image processing, visualization and volumetric evaluation. *International Journal of Medical Informatics*, 53:225–237.

Gomberg, B., Saha, P., Song, H. K., Hwang, S., and Wehrli, F. (2000). Topological analysis of trabecular bone mr images. *IEEE Transactions on Medical Imaging*, 19(3):166–174.

Gonzalez, R. C. and Woods, R. E. (2008). *Digital Image Processing*. Prentice Hall, third edition.

Guo, Z. and Hall, R. W. (1989). Parallel thinning with two-subiteration algorithms. *Communications of the ACM*, 32(3):359–373.

Haris, K., Efstratiadis, S. N., Maglaveras, N., Pappas, C., Gourassas, J., and Louridas, G. (1999). Model-based morphological segmentation and labeling of coronary angiograms. *IEEE Transactions on Medical Imaging*, 18(10):1003–1015.

Hart, C. (1998). *Doing a Literature Review: Releasing the social science research imagination*. Sage.

Heuch, H. (2003). Segmentation of the liver from mr and ct images. Master's thesis, Norwegian University of Science and Technology.

Hevner, S. March, J. P. and Ram, S. (2004). Design science research in information systems. *Management Information Systems Quarterly*, 28(1):75–105.

Hokland, J. (2002). Introduksjon til bayesiansk bildeanalyse. *Kompendium i fag SIF8068 - Statistisk Bildeanalyse og Læring*, pages 1–9.

Hough, P. V. C. (1959). Machine analysis of bubble chamber pictures. *International Conference on High Energy Accelerators and Instrumentation, CERN*.

Inaoka, N., Suzuki, H., and Fekuda, M. (1992). Hepatic blood vessel recognition using anatomical knowledge. In Loew, M. H., editor, *Medical Imaging VI: image processing*, pages 509–513. SPIE.

Intel (2007). Intel. http://www.intel.com.

Jiang, H. and Alperin, N. (2004). A new automatic skeletonization algorithm for 3d vascular volumes. In *Engineering in Medicine and Biology Society, 2004. IEMBS '04. 26th Annual International Conference of the IEEE*, volume 1, pages 1565–1568.

Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer-Verlag New York Inc., second edition.

Julesz, B., Gilbert, E. N., Shepp, L. A., and Frisch, H. L. (1973). Inability of humans to discriminate between visual textures that agree in second-order-statistics. *Perception*, 2:391–405.

Kapur, J. N., Sahoo, P. K., and Wong, A. K. C. (1985). A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing*, 29:273–285.

Kass, M., Witkin, A., and Terzoploulos, D. (1988). Snakes: Active contour models. *International Journal of Computer Vision*, pages 321–331.

Kelemen, A. and Székely, G. (1999). Elastic model-based segmentation of 3-d neuroradiological data sets. *IEEE Transactions on Medical Imaging*, 18(10):828–839.

Kelemen, A., Székely, G., and Gerig, G. (1998). Three-dimensional model-based segmentation of brain mri. In *Proceedings of the Workshop on Biomedical Image Analysis*, pages 4–13.

Kitchenham, B. (2004). Procedures for performing systematic reviews. Technical report, Keele University and NICTA.

Krivanek, A. and Sonka, M. (1998). Ovarian ultrasound image analysis: follicle segmentation. *IEEE Transactions on Medical Imaging*, 17(6):935–944.

Lam, L., Lee, S.-W., and Suen, C. (1992). Thinning methodologies-a comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9):869–885.

Lefohn, A. E., Kniss, J. M., Hansen, C. D., and Whitaker, R. T. (2003). Interactive deformation and visualization of level set surfaces using graphics hardware. *IEEE Visualization 2003*.

Liu, F., Zhao, B., and Kijewski, P. K. (2005). Liver segmentation for ct images using gvf snake. *Medical Physics*, 32(12):3699–3706.

Lobreget, S., Verbeek, P., and Groen, F. (1980). Three-dimensional skeletonization: principle and algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2:75–77.

Lobregt, S. and Viergever, M. A. (1995). A discrete dynamic contour model. *IEEE Transactions on Medical Imaging*, 14(1):12–24.

M. Frucci, P. P. and di Baja, G. S. (2008). *Case-Based Reasoning on Images and Signals*, chapter Case-Based Reasoning for Image Segmentation by Watershed Transformation, pages 319–353. Springer Berlin / Heidelberg.

Ma, C. and Sonka, M. (1996). A fully parallel 3d thinning algorithm and its applications. *Computer Vision and Image Understanding*, 64:420–433.

Malandain, G. and Bertrand, G. (1992). Fast characterization of 3d simple points. In *11th IEEE International Conference on Pattern Recognition*, pages 232–235.

Malandain, G., Bertrand, G., and Ayache, N. (1993). Topological segmentation of discrete surfaces. *International Journal of Computer Vision*, 10(2):183–197.

Malladi, R. and Sethian, J. A. (1996). A unified approach to noise removal, image enhancement, and shape recovery. *IEEE Transactions on Image Processing*, 5(11):1554–1568.

Malladi, R., Sethian, J. A., and Vemuri, B. C. (1995). Shape modeling with front propagation: A level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175.

Martinez-Perez, M., Hughes, A., Stanton, A., Thom, S., Bharath, A., and Parker, K. (1999). Segmentation of retinal blood vessels based on the second directional derivative and region growing. *Proceedings of the International Conference on Image Processing*, 2:173–176.

Mathworld (2007). Mathworld. `http://mathworld.wolfram.com/`.

Matlab (2005). *Matlab documentation*. The MathWorks, Inc.

McInerney, T. and Terzopoulos, D. (2000). Deformable models. In Bankman, I., editor, *Handbook of Medical Imaging*, pages 127–145. Academic Press.

Montagnat, J. and Delingette, H. (1997). Volumetric medical images segmentation using shape constrained deformable models. In *CVRMed*, pages 13–22.

Nakayama, Y., Li, Q., Katsuragawa, S., Ikeda, R., Hiai, Y., Awai, K., Kusunoki, S., Yamashita, Y., Okajima, H., Inomata, Y., and Doi, K. (2006). Automated Hepatic Volumetry for Living Related Liver Transplantation At Multisection CT. *Radiology*, 240(3):743–748.

Niblack, C. W., Gibbons, P. B., and Capson, D. W. (1992). Generating skeletons and centerlines from the distance transform. *CVGIP: Graph. Models Image Process.*, 54(5):420–437.

Nyström, I. and Smedby, Ö. (2001). Skeletonization of volumetric vascular images – distance information utilized for visualization. *Journal of Combinatorial Optimization*, 5(1):27–41. Special Issue on Optimization Problems in Medical Applications.

Omholt-Jensen, T. (2002). Segmentation of the hepatic vessels as seen in mr or ct images. Master's thesis, Norwegian University of Science and Technology.

Palágyi, K. and Kuba, A. (1999). A parallel 3d 12-subiteration thinning algorithm. *Graph. Models Image Process.*, 61(4):199–221.

Palagyi, K., Sorantin, E. Balogh, E., and Kuba, A. (2001). A sequential 3d thinning algorithm and its medical applications. In Insana, M. and Leahy, R., editors, *IPMI 2001, LNCS 2082*, pages 409–415. Springer-Verlag Berlin Heidelberg.

Papoulis, A. (1991). *Probability, Random Variables, and Stochastic Proceses*. McGraw-Hill, New York, 3rd edition.

Perner, P. (1999). An architecture for a cbr image segmentation system. In *ICCBR '99: Proceedings of the Third International Conference on Case-Based Reasoning and Development*, pages 525–534, London, UK. Springer-Verlag.

Perona, P. and Malik, J. (1990). Scalar-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639.

Pham, M., Susomboon, R., Disney, T., Raicu, D., and Furst, J. (2007). A comparison of texture models for automatic liver segmentation. In *Medical Imaging 2007: Image Processing.*, volume 6512 of *Lecture Notes in Computer Science*.

Qian, G., Sural, S., Gu, Y., and Pramanik, S. (2004). Similarity between euclidean and cosine angle distance for nearest neighbor queries. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 1232–1237, New York, NY, USA. ACM.

Ray, N., Acton, S. T., and Ley, K. (2002). Tracking leukocytes in vivo with shape and size constrained active contours. *IEEE Transactions on Medical Imaging*, 21(10):1222–1235.

Ripley, B. D. (1987). *Stochastic Simulation*. John Wiley & Sons.

Rost, R. J. (2004). *The OpenGL Shading Language*. Addison-Wesley Professional.

Russell, S. and Norvig, P. (2002). *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition.

Saha, P., Chauduri, B., and Majumder, D. (1997). A new shape preserving parallel thinning algorithm for 3d digital images. *Pattern Recognition*, 30:1939–1955.

Sandwell, D. T. (1987). Biharmonic spline interpolation of GEOS-3 and SEASAT altimeter data. *Geophysical Research Letters*, 12(2):139–142.

Sato, M., Bitter, I., Bender, M. A., Kaufman, A. E., and Nakajima, M. (2000). Teasar: Tree-structure extraction algorithm for accurate and robust skeletons. *pg*, 00:281.

Sclaroff, S. and Liu, L. (2001). Deformable shape detection and description via model-based region grouping. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(5):475–489.

Sedgewick, R. (1998). *Algorithms in C*, pages 11–20. Addison-Wesley, 3rd edition.

Seo, K., Ludeman, L. C., Park, S., and Park, J. (2004). Efficient liver segmentation based on the spine. In *Advances in Information Systems*, volume 3261 of *Lecture Notes in Computer Science*, pages 400–409. Springer Berlind/Heidelberg.

Sethian, J. A. (1996). *Level Set Methods and Fast Marching Methods: Evolving interfaces in Computatinal Geometry, Fluid Mechanics, Computer Vision and Materials Science*. Cambridge University Press.

Sethian, J. A. (1997). Tracking interfaces with level sets. *American Scientist*.

Sethian, J. A. (2004a). Evolving interface approach to shape recovery. `http://math.berkeley.edu/~sethian/Movies/Movieartery.html`.

Sethian, J. A. (2004b). Noise removal from images. `http://math.berkeley.edu/~sethian/Movies/Movienoiseremoval.html`.

Shawe-Taylor, J. and Cristianini, N. (2000). *Support Vector Machines and other kernel-based learning methods*. Cambridge University Press.

Skjermo, J. and Eidheim, O. C. (2005). Polygon mesh generation of branching structures. *14th Scandinavian Conference on Image Analysis*.

Soille, P. (2003). *Morphological Image Analysis*. Springer-Verlag.

Soler, L., Delingette, H., Malandain, G., Montagnat, J., Ayache, N., Koehl, C., Dourthe, O., Malassagne, B., Smith, M., Mutter, D., and Marescaux, J. (2001). Fully automatic anatomical, pathological, and functional segmentation from ct scans for hepatic surgery. *Computer Aided Surgery*, 6:131–142.

Sonka, M., Hlavac, V., and Boyle, R. (1999). *Image Processing, Analysis and Machine Vision*. PWS Publishing, second edition.

Strzodka, R. (2004). Gpgpu: General-purpose computing on graphics processors. *IEEE Visualization 2004*.

Székely, G. and Gerig, G. (2000). Model-based segmentation of radiological images. *Künstliche Intelligenz*, 3:18–23.

Thomas, J. G., Peters II, R. A., and Jeanty, P. (1991). Automatic segmentation of ultrasound images using morphological operators. *IEEE Transactions on Medical Imaging*, 10(2):180–186.

Tom, B., Efstratiadis, S., and Katsaggelos, A. (1994). Motion estimation of skeletonized angiographic images using elastic registration. *IEEE Transactions on Medical Imaging*, 13(3):450–460.

Tsagaan, B., Shimizu, A., Kobatake, H., and Miyakawa, K. (2002). An automated segmentation method of kidney using statistical information. In *MICCAI '02: Proceedings of the 5th International Conference on Medical Image Computing and Computer-Assisted Intervention-Part I*, pages 556–563, London, UK. Springer-Verlag.

Tuduki, Y., Murase, K., Izumida, M., Miki, H., Kikuchi, K., Murakami, K., and Ikezoe, J. (2000). Automated seeded region growing algorithm for extraction of cerebral blood vessels from magnetic resonance angiographic data. *Proceedings of the 22nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 3:1756–1759.

Vaishnavi, V. K. and Kuechler, W. J. (2007). *Design Science Research Methods and Patterns: Innovating Information and Communication Technology*. Auerbach.

Venkatasubramanian, S. (2003). The graphics card as a stream computer. *SIGMOD Workshop on Management and Processing of Data Streams*.

Vincent, L. (1993). Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms. *IEEE Transactions on Image Processing*, 2(2):176–201.

Vincent, L. and Soille, P. (1991). Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598.

Volkau, I., Zheng, W., Baimouratov, R., Aziz, A., and Nowinski, W. L. (2005). Geometric modeling of the human normal cerebral arterial system. *IEEE Transactions on Medical Imaging*, 24(4):529–539.

Wikipedia (2007). Wikipedia. http://www.wikipedia.org.

Winkler, G. (1995). *Image Analysis, Random Fields and Dynamic Monte Carlo Methods*. Springer-Verlag.

Xie, W., Thompson, R., and Perucchio, R. (2003). A topology-preserving parallel 3d thinning algorithm for extracting the curve skeleton. *Pattern Recognition*, 36:1529–1544.

Xu, C. and Prince, J. L. (1998). Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7(3):359–369.

Xu, C. and Prince, J. L. (2000). Gradient vector flow deformable models. In Bankman, I., editor, *Handbook of Medical Imaging*, pages 159–169. Academic Press.

Yim, P., Choyke, P., and Summers, R. (2000). Gray-scale skeletonization of small vessels in magnetic resonance angiography. *IEEE Transactions on Medical Imaging*, 19(6):568–576.

Zahlten, C., Jürgens, H., Evertsz, C. J. G., Leppek, R., Peitgen, H.-O., and Klose, K. J. (1995). Portal vein reconstruction based on topology. *European Journal of Radiology*, 19(2):96–100.

Zhou, Y. and Toga, A. W. (1999). Efficient skeletonization of volumetric objects. *IEEE Transactions on Visualization and Computer Graphics*, 5(3):196–209.

Appendix A

Interconnecting segmented hepatic vessels in adjacent CT slices

O. C. Eidheim[a], L. Aurdal[b], T. Omholt-Jensen[c], T. Mala[d], B. Edwin[d]
[a]Norwegian Institute of Science and Technology, Trondheim, Norway
[b]Norwegian Computing Centre, Oslo, Norway
[c]Agentus, Trondheim, Norway
[d]Interventional Centre, Rikshospitalet, Oslo, Norway

Published in NOBIM, 2004

Abstract. Deriving liver vessel structure from CT and MR scans manually is time consuming and error prone. An automatic procedure which could help the radiologist in her analysis is therefore needed. We use matched filters to emphasise blood vessels, and entropy based thresholding to segment the vessels complemented by segmentation with respect to local variances. Vessel interconnections are extracted and finally exported to a graph structure for further refinements. Results show that the methods presented are promising and can eventually be used clinically.

A.1 Introduction

The liver is extensively perfused and during liver surgery special care has to be taken to avoid bleedings. Prior to surgery, CT scans are examined carefully, and especially the position of large hepatic vessels must be determined. This process is currently done manually and is time consuming and error prone.

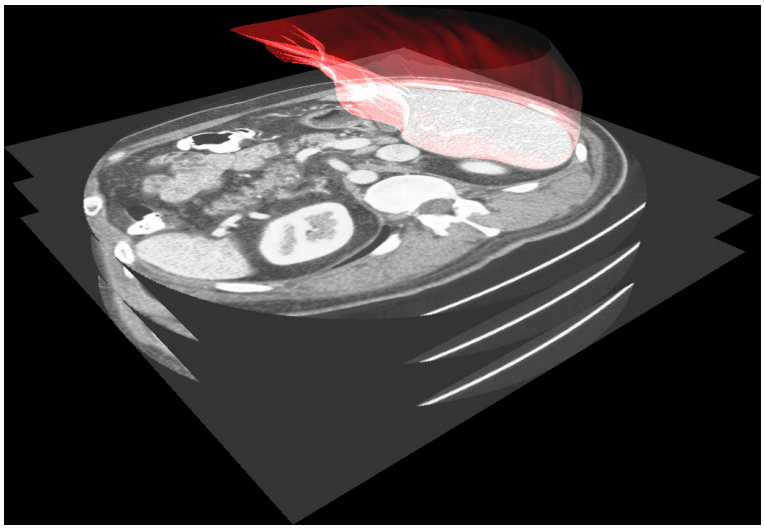


Figure A.1: An illustration of the datasets we are processing. The liver, outlined in red, is positioned together with a few CT slices.

We have sought ways to delineate these vessels from CT images automatically, and methods for deriving initial interconnections between segmented vessels in adjacent CT slices. A separate ongoing project deals with improving the initial interconnections through global search methods, but this will not be discussed here. For illustrative purposes, we have added figure A.1 to show the liver positioned together with a few CT slices.

This work is a continuation of the master thesis [1].

A.1.1 Previous methods

Several articles propose methods for delineating the liver vessels automatically [2, 3, 4, 5, 6]. Most promising, with respect to creating a 3D model of the liver vessel structure, are the methods by Zalthen et al. and Soler et al.

Zalthen et al. use a voxel based region-growing-algorithm to extract the portal vein, but the algorithm requires an initial seed point and is therefore not fully automatic. In the article by Soler et al. however, the portal trunk is located using its general anatomical position. The portal vein skeleton is calculated utilising methods of Malandain and Bertrand [7, 8], and is corrected by pruning vessel segments that do not confirm with a set of predetermined properties.

Both papers deal with delineation of the portal vein, where as we seek a complete vessel structure. Moreover, both concepts previously described

are based on local knowledge and do not regard the vessel structure as a whole. We seek methods which on the contrary do that, and that incorporate knowledge based decisions.

A.2 Methods

The proposed method consists of five main processing steps. A histogram equalisation [9] is first executed on each CT slice to normalise the contrast of each image. Second, Matched filtering [2] is used to emphasise the hepatic vessels. Entropy based thresholding [3] is used to segment the blood vessels in combination with knowledge based threshold selection [10] and several morphological operations [11, 9, 12]. Segmentation results are furthermore improved by analysis of local image variances. The vessel centres are then extracted mainly through further use of morphological operators, and modified with respect to vessels in adjacent CT slices. Finally, the vessel segments are interconnected based on the Euclidean distance to other segments in neighbouring CT images, as well as on the segments' sizes.

A.2.1 Preprocessing

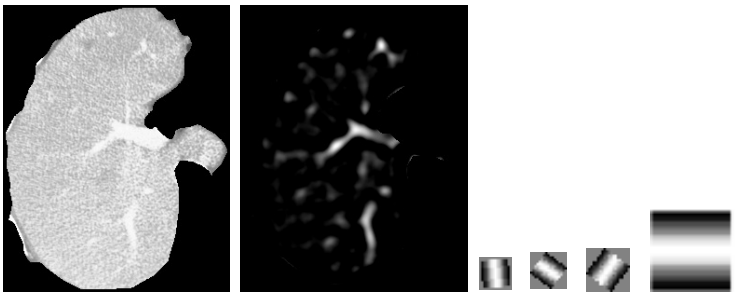
The preprocessing step starts by masking the liver. A prerequisite to our algorithm is therefore an already segmented liver. Liver segmentation is the subject of a separate study at Rikshospitalet, and we assume for the purpose of this article that such a mask exists.

CT scans may contain images with varying contrast, but in order to process the datasets automatically, the image intensities have to be normalised. For this purpose, we used histogram equalisation [9] applied on all the images in the CT sequence.

Matched filtering [2] using rotated and scaled Gaussian hill templates is used to emphasise blood vessels. Blood vessels seen in MR and CT images have an approximately Gaussian profile in 1D [2]. By convolving the images with rotated and scaled Gaussian hill templates and summing the resulting images, blood vessels having the same size as the templates are more clearly distinguished. See figure A.2.

A.2.2 Segmentation

The images are first segmented using entropy based thresholding [3] combined with a model based threshold selection [10]. The entropy of the images



(a) (b) (c)

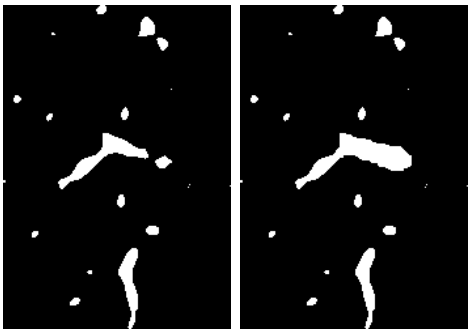
Figure A.2: Preprocessing: a) Masked liver from a CT image after histogram equalisation. b) After applying matched filtering, the vessels are emphasised and can be more clearly distinguished by a following segmentation algorithm. c) A sample of the rotated and scaled Gaussian hill templates used to emphasise the blood vessels.

resulting from each possible threshold level is calculated. Typically, the relationship between thresholds and entropy will have several local maxima. To choose among these we use a method proposed by Glombitza et al. where each locally optimal threshold value is tested in a model based selection algorithm. Figure A.3 a) shows an example segmentation.

During the preprocessing step however, matched filtering fails to emphasise large and irregular blood vessels, e.g. vessels cut by the liver mask. In order to achieve a sound segmentation of these vessels as well, we have developed an additional segmentation technique.

The intensity variance surrounding pixels within a hepatic vessel is lower than in the surrounding tissue. Based on this, we have formulated an algorithm that segments large vessels unconstrained by the profile of the vessels. The algorithm is performed in five basic steps:

1. Local minima are filled using morphological filling 11.
2. The variance for each pixel and its surrounding neighbours is calculated resulting in a variance image.
3. A $Marker$ is thresholded from the variance image using a threshold set close to zero variance.
4. A $Mask$ is likewise segmented, but by using a slightly higher threshold.
5. A morphological reconstruction based on dilation [11] is finally executed to reconstruct the identified liver vessels from the $Marker$ image.



(a) (b)

Figure A.3: a) A segmentation of the matched filtered image in figure A.2 using entropy based thresholding in combination with model based threshold selection. b) Segmentation result by adding segmentation with respect to local variance to a). The large vessel cut by the liver mask is now more correctly segmented.

The basic idea is to first find the areas with low local variance. These areas typically corresponds to pixels within large hepatic vessels. Then, to segment the identified vessels, the areas are enlarged with a slightly lower variance set as a constraint. The results are shown in figure A.3 b).

Both segmentation results, from entropy based thresholding and from the procedure just described, are eventually unified.

Morphological opening and closing [11] is used to remove noise and to reconnect segments separated during segmentation.

A.2.3 Vessel delineation

After the vessels are segmented, a method for interconnecting them between CT slices is needed. We chose to distinguish the vessel segments into two classes:

1. Vessels running perpendicular to a CT slice.

2. Vessels running at an oblique angle to the CT slice.

Each segment is classified by measuring the vessel's convex hull relatively to its actual area, and by comparing the segment's minor and major axes. Figure A.4 shows an example classification.

The next step is to identify the vessel coordinates. The centre of a vessel running perpendicular to the CT slice, that is a circular vessel segment,

Figure A.4: Each segment is classified according to its convex hull relative to its actual area, and the relationship between the segment's minor and major axes. Grey and white vessel segments are classified as vessels running perpendicular to the CT slice and vessels running at an oblique angle to the CT slice respectively.

is simply its mass centre. On the other hand, the calculation of the centres in vessels running at an oblique angle to the CT slice is slightly more complicated. First, the segment's skeleton is measured using morphological thinning [9, 11] iterated until stability. In order to receive the most basic skeleton while retaining the same Euler number [9], the endpoint segments are pruned [9, 11] from the skeleton. The skeleton is finally used to derive the vessel centres. See figure A.5 for an example.

During the vessel graph initialisation, the endpoints of the skeletons and the mass centres will be used as potential interconnection points. We will discuss this in greater detail in chapter A.2.4.

Large vessels running in parallel with the CT slices may be projected in more than one slice. Using the current vessel centres, visualisation erroneously show several parallel vessels. Another problem arises when a vessel running at an oblique angle to the CT slice, branches into a vessel running perpendicular to the slice as shown in figure A.6 a). In this case, there are no interconnection point that could interconnect these two segments. Corrections of the vessel centres and interconnection points are therefore required. First, these corrections consists of removing vessel centres based on the thickness of similar vessel segments lying in corresponding positions in adjacent CT slices. After this, potential interconnection points are added making likely interconnections possible. See figure A.7 for an example result.

Figure A.5: Vessel centres, in light gray, are derived either from the segment's skeleton or the mass centre depending on the classification of the segment. The endpoints of the skeletons and the mass centres will be treated as potential interconnection points in the graph initialisation phase.

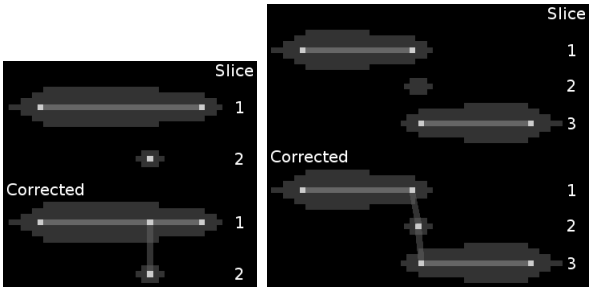
The algorithm is as follows:

1. Follow the skeleton of each segment in each CT slice.
2. Remove the centres whose distance to the background is smaller than that of nearby centres in adjacent CT slices. Review figure A.7.
3. Follow the remaining skeletons.
4. Add potential interconnection points nearby endpoints or mass centres if there exist no such points already. See figure A.6 a).
5. If endpoints have been removed in step 2 such that potential graph interconnections are made impossible, points are added to make the interconnections possible. See figure A.6 b).

The distance, mentioned in step 2, is measured using a Euclidean distance map of the segmented vessels. Since only the centre most distance is of interest, that is the distance corresponding to the size of the vessel, the distance map is morphologically dilated and masked as shown in figure A.8.

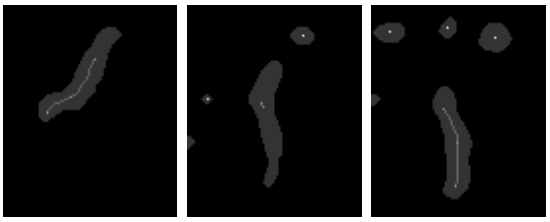
A.2.4 Initialisation of the vessel graph

In the last phase, vessel interconnection points resembling in size and with similar positions, are interconnected constituting the initial vessel graph.



(a) (b)

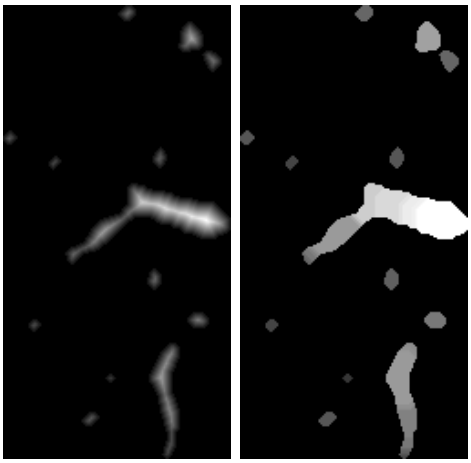
Figure A.6: Vessel centre corrections. White pixels correspond to interconnection points, and opaque gray lines are possible interconnections. a) Add potential interconnections points nearby endpoints or mass centres if no such points exist. b) Interconnection points are added if interconnections between two segments are made impossible.



(a) (b) (c)

Figure A.7: This figure shows the results after applying the centre corrections on three adjacent CT images. A vessel branches from a), and continues into b) and c).

The size of the interconnection points are measured using the previously described distance map shown in figure A.8 b). There is one major concern however, when building the initial vessel graph. The graph may contain loops that are unlikely with respect to vessel anatomy. We resolve these loops by first ordering the chosen interconnections by distance and size between the interconnection points. Finally, each interconnection is added individually if it does not contribute to a vessel loop.



(a) (b)

Figure A.8: a) An Euclidean distance map of the image in figure A.3 b). b) The distance map is morphologically dilated and masked with the original segmented vessels. This is done to ensure that the right distance, the one corresponding to the size of the vessel, is picked.

A.3 Results

The results show that the initial interconnections, using the previously described methods, yield a vessel graph close to the anatomical truth. In figure A.9, the portal vein originating from the vena cava, can be seen forking into the liver in several directions. 3 CT scans were used in testing, and the liver vessels were partially segmented and interconnected. Due to the complexity of the task at hand, a perfectly correct vessel graph is not achievable by our methods, but we present an improvement with respect to already published material.

The processing time of applying the entire procedure on one CT scan is approximately one hour.

A.4 Conclusion and discussion

In conclusion, we have developed a method that once the liver is segmented, automatically generates an initial vessel graph describing the vessel interconnections in the liver. Based on these interconnections, global search mechanisms can be applied to further improve the results using cost functions that favour likely vessel structures. The results can finally be visualised in 3D and

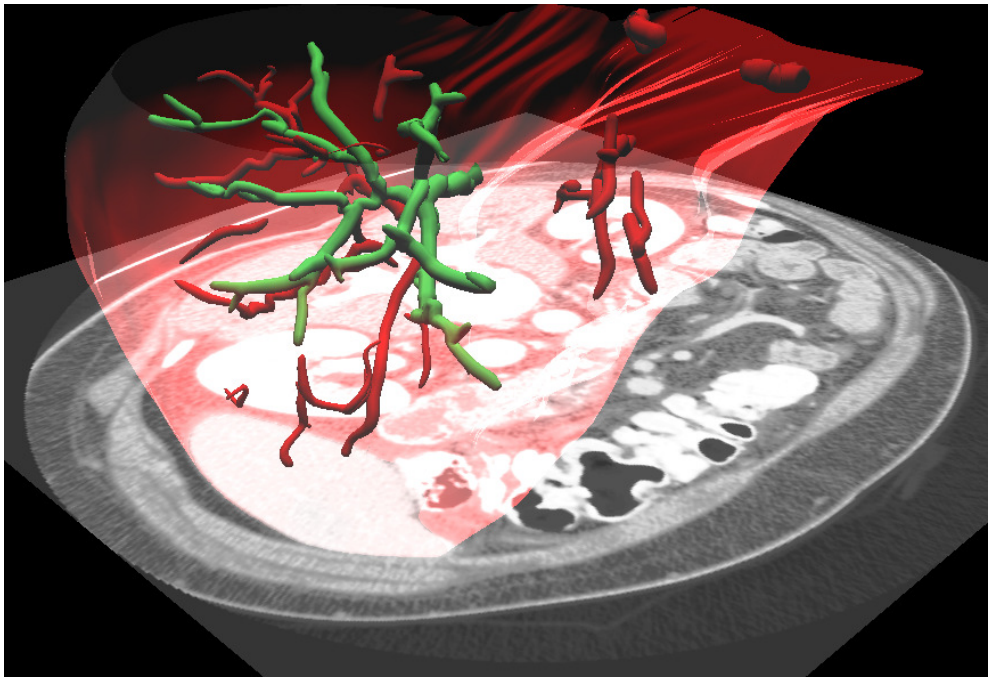


Figure A.9: A visualisation of an initial interconnected graph. Outline of the processed liver and a CT image is also visualised for evaluation. A fully interconnected vessel graph can be seen in green, corresponding to a part of the portal vein expanding into the liver.

provide valuable assistance for radiologists and surgeons during liver surgery planning.

Acknowledgements

This project was funded by Norwegian University of Science and Technology. Rikshospitalet, the Norwegian national hospital, provided us with the datasets used for testing in this article.

Bibliography

[1] Tormod Omholt-Jensen. Segmentation of the hepatic vessels as seen in mr or ct images. Master's thesis, Norwegian University of Science and Technology, June 2002.

[2] Subhasis Chaudhuri, Shankar Chatterjee, Norman Katz, Mark Nelson, and Michael Goldbaum. Detection of blood vessels in retinal images using two-dimensional matched filters. IEEE Transactions on Medical Imageing, 8(3):263-269, 1989.

[3] J. N. Kapur, P. K. Sahoo, and A. K. C. Wong. A new method for gray-level picture thresholding using the entropy of the histogram. Computer Vision, Graphics, and Image Processing, 29:273-285, 1985.

[4] N. Inaoka, H. Suzuki, and M. Fekuda. Hepatic blood vessel recognition using anatomical knowledge. In Murray H. Loew, editor, Medical Imaging VI: image processing, pages 509-513. SPIE, 1992.

[5] Cornelia Zahlten, H. Jürgens, C. J. G. Evertsz, R. Leppek, H.-O. Peitgen, and K. J. Klose. Portal vein reconstruction based on topology. European Journal of Radiology, 19(2):96-100, 1995.

[6] Luc Soler, Herve Delingette, Gregoire Malandain, Johan Montagnat, Nicholas Ayache, Christophe Koehl, Olivier Dourthe, Benoit Malassagne, Michell Smith, Didier Mutter, and Jacques Marescaux. Fully automatic anatomical, pathological, and functional segmentation from ct scans for hepatic surgery. Computer Aided Surgery, 6:131-142, 2001.

[7] G. Bertrand and G. Malandain. A new characterization of three-dimensional simple points. Pattern Recognition Letters, 15(3):196-175, 1994.

[8] G. Malandain, G. Bertrand, and N. Ayache. Topological segmentation of discrete surfaces. International Journal of Computer Vision, 10(2):183-197, 1993.

[9] Rafael C. Gonzalez and Richard E. Woods. Digital Image Processing. Prentice Hall, second edition, 2002.

[10] Gerald Glombitza, Wolfram Lamadè, Athanasios M. Demiris, Marc-Roger Göpfert, Achim Mayer, Malte L. Bahner, Hans-Peter Meinzer, Göte Richter, Thomas Lehnert, and Christian Herfarth. Virtual planning of liver resections: image processing, visualization and volumetric evaluation. International Journal of Medical Informatics, 53:225-237, 1999.

[11] Pierre Soille. Morphological Image Analysis. Springer-Verlag, 2003.

[12] Milan Sonka, Vaclav Hlavac, and Roger Boyle. Image Processing, Analysis and Machine Vision. PWS Publishing, second edition, 1999.

Appendix B

Segmentation of liver vessels as seen in MR and CT images

O. C. Eidheim[a],*, L. Aurdal[b], T. Omholt-Jensen[c], T. Mala[d], B. Edwin[d]
a Norwegian Institute of Science and Technology, Trondheim, Norway
b Norwegian Computing Centre, Oslo, Norway
c Agentus, Trondheim, Norway
d Interventional Centre, Rikshospitalet, Oslo, Norway

Published in Computer Assisted Radiology and Surgery, 2004

Abstract. Deriving liver vessel structure from CT and MR scans manually is time consuming and error prone. An automatic procedure that could help the radiologist in her analysis is therefore needed. We use matched filters to emphasise blood vessels, and entropy based thresholding to segment the vessels. Vessel interconnections are next extracted and exported to a graph structure. At the end, genetic algorithms are used to search globally for the most likely graph based on a set of fitness functions. Results show that the methods presented are promising and can eventually be used clinically.

Keywords: vessel; liver; segmentation; vizualisation

B.1 Introduction

The liver is a vital organ with vascular, metabolic, secretory, and excretory functions. It is extensively perfused and during liver surgery special care has to be taken in order to avoid bleedings. Prior to liver surgery, the patient will typically be examined using MR and/or CT scans, in particular the position of large hepatic vessels must be determined. The relative position of,

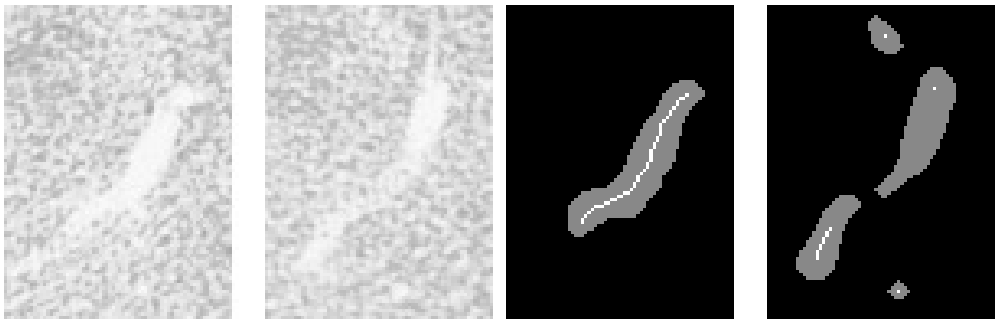
for instance, tumours to these vessels are of great importance when planning the procedure and in evaluating the operability of the patient. Surgeons and radiologists will typically base their evaluation on a visual inspection of the 2D slices produced by the different imaging modalities. It is difficult, however, to deduce a detailed liver vessel structure from such images. Surgeons at the Interventional Centre at Rikshsopitalet have found 3D renderings of the liver and its internal vessel structure to be a valuable aid in this complex evaluation phase. Currently, these renderings are based on a largely manual segmentation of the liver vessels. This procedure is time consuming and error prone, and we have sought a way to extract the liver vessel structure automatically from MR and CT scans.

B.1.1 Previous methods

Several articles propose methods to delineate the liver vessels automatically [1,2,3,4,5]. Most promising, with respect to creating a 3D model of the liver vessel structure, are the methods by Zalthen et al. and Soler et al. Zalthen et al. use a voxel based region-growing-algorithm to extract the portal vein, but the algorithm requires a manually set initial seed point and is therefore not fully automatic. In the article by Soler et al., the portal trunk is located using its general anatomical position. The portal vein skeleton is calculated utilising methods of Malandain and Bertrand [6,7], and is corrected by pruning vessel segments that do not confirm with a set of predetermined properties. Both papers deal with delineation of the portal vein, whereas we seek a complete vessel structure. Moreover, both concepts previously described are based on local knowledge and do not regard the vessel structure as a whole. We seek methods which on the contrary do that, and that are incorporated with knowledge-based decisions.

B.2 Methods

The proposed method consists of three main steps. First, a preprocessing step where the liver itself is isolated using a mask and where vessel candidates are detected in the different 2D slices of the MR or CT sequence. Second, an initialisation step is made where these candidates are interconnected in a graph structure based on simple proximity considerations. Finally, a main processing step is performed where the graph structure is refined according to global fitness criterion using genetic algorithms.



(a) (b)

Figure B.1: a) A vessel before processing, shown in two adjacent CT slices. b) The resulting segmentation (in grey) and derived vessel centres (in white) of a). The centres are modified with respect to adjacent vessel segments.

B.2.1 Preprocessing

The preprocessing step starts by masking the liver, and this obviously requires a segmentation of the liver. Liver segmentation is the subject of a separate study at Rikshospitalet, for the purpose of this article we will assume that such a mask exists. After masking the liver, a histogram equalisation [8] is performed in order to normalise the contrast of the images. MR and CT sequences may contain images with varying contrast. In order to obtain algorithms that perform autonomously this step is required. Finally, matched filtering [1] using rotated and scaled Gaussian hill templates is used to emphasise blood vessels. Blood vessels seen in MR and CT images have an approximately Gaussian profile in 1D [1]. By convolving the images with rotated and scaled Gaussian hill templates and summing the resulting images, blood vessels having the same size as the templates are more clearly distinguished.

B.2.2 Initialisation

During the initialisation step, candidate liver vessels are located in the 2D images of the MR or CT sequences. This is performed in two steps:

1. The images are initially thresholded using entropy based thresholding [2] combined with a model based threshold selection [9]. The entropy of the images resulting from each possible threshold level is calculated. Typically, the relationship between thresholds and entropy will have several local maxima. To choose among these we use a method proposed

by Glombitza et al. where each locally optimal threshold value is tested in a model based selection algorithm.

2. The thresholding results are refined using morphologic operators [8,10,11]. Performing a morphological opening and closing on the thresholded images removes spurious pixels and reconnects separated segments. Some of the vessel segments may in addition contain interior holes. In particular, this occurs in branching points where the vessels do not match any of the templates used during the matched filtering. These holes are filled by geodesic reconstruction by erosion [10].

The result of these two steps is a number of candidate vessel segments. These segments can correspond to either a vessel running vertically through the MR or CT slice, a vessel running at an oblique angle to the slice, or erroneous segments not corresponding to vessels. If the area of a segment's convex hull is close to its actual area, and the segment is circular, it is classified as a vessel running vertically through the slice. Otherwise, the segment is classified as a vessel running at an oblique angle to the slice. At this point, a graph representing the vessel interconnection is deduced from the classified vessel segments. Nodes in the graph represent vessel interconnection points, and the edges represent vessel segment interconnections. Segments corresponding to vessels running perpendicular to the slices and those running at oblique angles are treated separately. Vessels running perpendicular to the slices are represented by their geometric centres. These centres become nodes in the graph. Vessel segments corresponding to vessels running at oblique angles are thinned and pruned in order to create a basic skeleton from which interconnections and additional nodes are derived. The skeleton is also modified with respect to vessels in adjacent slices (See Fig. 1). At the end, interconnections are computed between neighbouring MR or CT slices based on the Euclidean distance between the nodes on separate slices, and the size of the nodes.

B.2.3 Main processing step

In this step the initial vessel graph is improved using genetic algorithms [12,13]. This is a global search procedure that finds the best graph according to a set of fitness functions. The aim is to find the graph with highest possible fitness. Likely vessel structures are awarded with a high fitness, and improbable structures are penalised with low fitness. Five fitness functions were defined:

1. Distance fitness: The high resolution of modern MR and CT scanners

makes it reasonable to assume that vessel segments belonging to the same vessel will rarely be far apart. Interconnecting segments that are close is therefore awarded with high fitness, while interconnections involving physically distant segments are given low fitness.

2. Curvature fitness: Sharp turns in vessel paths are infrequent. This fitness function will therefore reward straight vessels, while penalising heavily curved vessels.

3. Dimension fitness: Since blood vessels normally increase or decrease in thickness along a given direction, vessel interconnections in our graph that do not conform with this are given low fitness.

4. Branching fitness: Vessels rarely branch in more than two subvessels. Thus, we penalise branches which result in more than two vessels.

5. Loop fitness: Vessels should not form local loops and graphs containing loops are given low fitness.

Currently, all algorithms with the exception of the generation of the liver mask are fully automatic.

Fig. 2.

B.3 Results

We have applied our methods to 3 full CT sequences of the liver. The resulting 3D vessel graph from each sequence clearly shows the structure of the blood vessels within the liver (see Fig. 2). By using the global search mechanism, a likely vessel structure was found, and improbable vessel structures, as for instance vessel loops, were resolved in a reliable manner. The quality of our results has been verified by inspection by radiologists and surgeons. The preprocessing and initialisation steps are computationally simple, whereas the global search using genetic algorithms is computationally demanding. This is due to the large search space resulting from a detailed vessel graph and numerous fitness functions. The computation time for the entire procedure is on the order of one hour on a modern personal computer.

B.4 Discussion and conclusion

In conclusion, we have developed an application that automatically finds the most probable vessel interconnections in the liver. However, the masks we

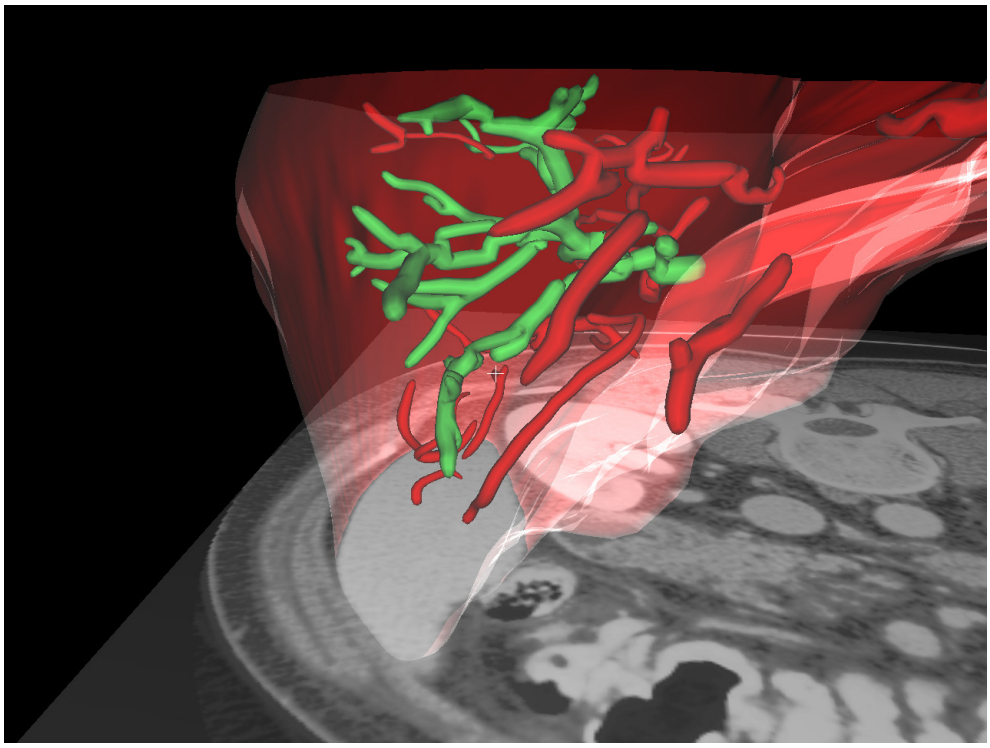


Figure B.2: A visualised vessel graph with one continuous branch marked green. In our application the liver contour and CT slices are visualised for verification.

currently use exclude the vena cava. Noise in the CT images as well as lack of interconnections via the vena cava, may give raise to isolated vessel segments that should otherwise have been connected (see Fig. 2). In the future we plan to implement improved preprocessing methods, and use better liver masks that for instance include the vena cava. Based on the interconnections it is a simple task to render the vessel structure of the liver in 3D. Given the improvements previously mentioned, our application can provide valuable assistance for radiologists and surgeons during liver surgery planning.

Acknowledgements

This project was funded by Norwegian University of Science and Technology. Rikshospitalet in Norway provided us with the datasets used for testing in this article.

Bibliography

[1] S. Chaudhuri, S. Chatterjee, N. Katz et al. Detection of blood vessels in retinal images using two-dimensional matched filters. IEEE Transactions on Medical Imaging, 8(3):263-269, 1989.

[2] J. N. Kapur, P. K. Sahoo, and A. K. C. Wong. A new method for gray-level picture thresholding using the entropy of the histogram. Computer Vision, Graphics, and Image Processing, 29:273-285, 1985.

[3] N. Inaoka, H. Suzuki, and M. Fekuda. Hepatic blood vessel recognition using anatomical knowledge. In Murray H. Loew, editor, Medical Imaging VI: image processing, pages 509-513. SPIE, 1992.

[4] C. Zahlten, H. Jurgens, C. J. G. Evertsz et al. Portal vein reconstruction based on topology. European Journal of Radiology, 19(2):96-100, 1995.

[5] L. Soler, H. Delingette, G. Malandain et al. Fully automatic anatomical, pathological, and functional segmentation from ct scans for hepatic surgery. Computer Aided Surgery, 6:131-142, 2001.

[6] G. Bertrand and G. Malandain. A new characterization of threedimensional simple points. Pattern Recognition Letters, 15(3):196-175, 1994.

[7] G. Malandain, G. Bertrand, and N. Ayache. Topological segmentation of discrete surfaces. International Journal of Computer Vision, 10(2):183-197, 1993.

[8] R. C. Gonzalez and R. E.Woods. Digital Image Processing. Prentice Hall, second edition, 2002.

[9] G. Glombitza, W. Lamade, A. M. Demiris et al. Virtual planning of liver resections: image processing, visualization and volumetric evaluation. International Journal of Medical Informatics, 53:225-237, 1999.

[10] P. Soille. Morphological Image Analysis. Springer-Verlag, 2003.

[11] M. Sonka, V. Hlavac, and R. Boyle. Image Processing, Analysis and Machine Vision. PWS Publishing, second edition, 1999.

[12] David E. Goldberg. Genetic Algorithms in Search, Optimization & Ma-

chine Learning. Addison-Wesley, 1998.

[13] Wolfgang Banzhaf, Peter Nordin, Robert E. Keller, and Frank D. Francone. Genetic Programming: An Introduction. Morgan Kaufmann Publishers, Inc, 1998.

Appendix C

Polygon Mesh Generation of Branching Structures

Jo Skjermo and Ole Christian Eidheim
Norwegian University of Science and Technology,
Department of Computer and Information Science.

Published in 14th Scandinavian Conference on Image Analysis, 2005

abstract. We present a new method for producing locally non-intersecting polygon meshes of naturally branching structures. The generated polygon mesh follows the objects underlying structure as close as possible, while still producing polygon meshes that can be visualized efficiently on commonly available graphic acceleration hardware. A priori knowledge of vascular branching systems is used to derive the polygon mesh generation method. Visualization of the internal liver vessel structures and naturally looking tree stems generated by Lindenmayer-systems is used as examples. The method produce visually convincing polygon meshes that might be used in clinical applications in the future.

C.1 Introduction

Medical imaging through CT, MR, Ultrasound, PET, and other modalities has revolutionized the diagnosis and treatment of numerous diseases. The radiologists and surgeons are presented with images or 3D volumes giving them detailed view of the internal organs of a patient. However, the task of analyzing the data can be time-consuming and error-prone. One such case is liver surgery, where a patient is typically examined using MR or CT scans prior to surgery. In particular, the position of large hepatic vessels must be

determined in addition to the relative positions of possible tumors to these vessels.

Surgeons and radiologists will typically base their evaluation on a visual inspection of the 2D slices produced by CT or MR scans. It is difficult, however, to deduce a detailed liver vessel structure from such images. Surgeons at the Intervention Centre at Rikshsopitalet in Norway have found 3D renderings of the liver and its internal vessel structure to be a valuable aid in this complex evaluation phase. Currently, these renderings are based on a largely manual segmentation of the liver vessels, so we have explored a way to extract and visualize the liver vessel structure automatically from MR and CT scans.

The developed procedure is graph based. Each node and connection corresponds to a vessel center and a vessel interconnection respectively. This was done in order to apply knowledge based cost functions to improve the vessel tree structure according to anatomical knowledge. The graph is used to produce a polygonal mesh that can be visualized using commonly available graphic acceleration hardware.

A problem when generating meshes of branching structures in general, is to get a completely closed mesh that does not intersect itself at the branching points. We build on several previous methods for mesh generation of branching structures, including methods from the field of visualization for generation of meshes of tree trunks. The main function of a tree's trunk can be explained as a liquid transportation system. The selected methods for the mesh generation can therefore be extended by using knowledge of the branching angles in natural systems for fluid transportation. This enables us to generate closed and locally non-intersecting polygon meshes of the vascular branching structures in question.

C.2 Previous Work

In the field of medical computer imagery, visualization of internal branching structures have been handled by volume visualization, as the data often was provided by imaging systems that produced volume data. However, visualization of polygon meshes is highly accelerated on modern commonly available hardware, so we seek methods that can utilize this for our visualization of branching vascular transportation structures.

Several previous works have proposed methods for surface mesh generation of trees that handles branching. We can mention the parametric surfaces used in [1], the key-point interpolation in Oppenheimer [14], the "branching ramiforms" [2] (that was further developed by Hart and Baker in [9] to ac-

count for ”reaction wood”), and the ”refinement by intervals” method [11].

In [12], [17], *rule based mesh growing* from L-systems was introduced. The algorithm used mesh connection templates for adding new parts of a tree to the mesh model, as L-system productions was selected during the generation phase. The mesh connection templates were produced to give a final mesh of a tree, that could serve as a basis mesh for subdivision. This method could only grow the mesh by rules, and could not make a mesh from a finished data set.

The work most similar to our was the *SMART* algorithm presented in [6], [7]. This algorithm was developed for visualization of vascular branching segments in the liver body, for use in a augmented reality aided surgery system. The algorithm produced meshes that could be used with Catmull-Clark subdivision [3] to increase surface smoothness and vertex resolution.

The *SMART* algorithm defined local coordinate axis in a branching point. The average direction of the incoming and outgoing segments was one axis, and an *up* vector generated at the root segment was projected along the cross sections to define another axis (to avoid twist). The child closest to the average direction was connected with quads, at a defined distance. The *up* vector defined a square cross section, and four directions, at a branching point. All remaining outgoing segments were classified into one of these directions according to their angle compared with the average direction. The child closest by angle in each direction was connected with the present tile, and this was recursively repeated for any remaining children.

Furthermore, the *SMART* algorithm did not include any method for automatic adjustment of the mesh with respect to the areas near forking points, and could produce meshes that intersected locally if not manually tuned. Our method automatically generates meshes without local intersection as long as the underlying structures loosely follows natural branching rules.

C.3 Main Algorithm

The proposed algorithm is loosely based on the *SMART* algorithm. It also uses knowledge of the branching angles in natural systems for fluid transportation as described in Sect.C.3.1.

C.3.1 Natural Branching Rules

Leonardo Da Vinci presented a rule for estimating the diameter of the segments after a furcation in blood vessels, as stated in [15]. The *Da Vinci* rule states that the cross-section area of a segment is equal to the combined cross

section area of the child segments, as seen in the following section.

$$\pi r_0^2 = \pi r_1^2 + \pi r_2^2 + ... + \pi r_n^2 \tag{C.1}$$

A generalization, as seen in C.2, was presented by Murray in [13]. Here, the *Da Vinci* rule has been reduced so that the sum of the diameters of the child segments just after a furcation is equal to the diameter of the parent just before the furcating, where d_0,d_1, and d_2 are the diameters of the parent segment and the child segments, respectively. α was used to produce different branching. α values between 2 and 3 are generally suggested for different branching types.

$$d_0^\alpha = d_1^\alpha + d_2^\alpha \tag{C.2}$$

From this Murray could find several equations for the angle between 2 child branches after a furcation. One is shown in C.3, where x and y are the angles between the direction of the parent and each of two child segments. As seen from the equation, the angles depend on the diameter of each of the child segments. Murray also showed that the segments with the smallest diameter have the largest angle.

$$\cos(x+y) = \frac{d_0^4 - d_1^4 - d_2^4}{2d_1^2 d_2^2} \tag{C.3}$$

Thus, we assume that the child segment with the largest diameter after a furcation, will have the smallest angle difference from the direction of the parent segment. This forms the basis for our algorithm, and it will therefore produce meshes that do not locally intersect as long as the underlying branching structure mostly follows these rules that are based on observations from nature. We now show how this is used to produce a polygon mesh of a branching structure.

C.3.2 New Algorithm

The algorithm is based on the extended *SMART* algorithm, but uses the *Da Vinci* rule to ensure mesh consistency. It uses data ordered in a DAG (Directed Acyclic Graph) where the nodes contains the data for a segment (direction vector, length and diameter). The segments at the same level in the DAG are sorted by their diameters. An *up* vector is used to define the vertices at each segments start and end position. The vertex pointed to by the *up* vector, is set to be a corner of a square cross section of a segment. The sides of this square defines the directions used in sorting any child segments. The sorting results in four sets of child segments (one for each direction of

the square), where the child segments in each set are sorted by the largest diameter.

To connect segments, we basically sweep a moving coordinate frame (defined by a projected *up* vector) along a path defined by the segments data. However, at the branching points we must build another type of structure with vertices, so we can add the child segments on to the polygon mesh. This is done by considering the number of child segments, and their diameters and angles compared to the parent segment.

Starting at the root node in the DAG we process more and more child segments onto the polygon mesh recursively. There are four possible methods for building the polygon mesh for any given segment. If there are one or more child segments in the DAG, we must select one of the methods described in Sect.C.3.3 (for one child segment), Sect.C.3.5 (for more then one child segment), or in Sect.C.3.4 (for cases where the child segment with largest diameter has an angle larger then 90 degrees with respect of the parent segment). If there are no child segments, the branch is at its end. The segment is closed with a quad polygon over four vertices generated on a plane defined by the projected *up* vector and the segments diameter at the segments end.

C.3.3 Normal Connection

If there is one child segment (with angle between the present and the child segment less then 90 degrees), we connect the child segment to the present segment, and calculates the vertices, edges and polygons as described in this section. Each segment starts with four vertices on a plane at the segments start position. As the algorithm computes a segment, it finds four vertices at the segment's end. It then closes the sides of the box defined by these eight vertices (not the top and bottom).

The first step is to calculate the average direction between the present segment, and the child segment. This direction is the *half direction*. Next, the up vector is projected onto the plane defined by the *half direction* and the segments end point. A square cross section is then defined on the plane at the segment's end position, oriented to the projected *up* vector to avoid twist. The length of the *up* vector is also changed to compensate for the tilting of this plane compared to the original vector. The corners of the cross section are the four end corner vertices for the present segment. These vertices, along with the four original vertices, defines the box that we close the sides of with quad polygon faces. In Fig.C.1, the mesh of a stem made of four segments connected in sequence can be seen. After processing the segment, the child segment is set as the present segment.

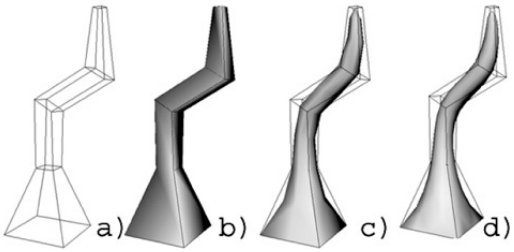


Figure C.1: Simple mesh production. a) the produced mesh, b) shaded mesh, c) one subdivision, d) two subdivisions

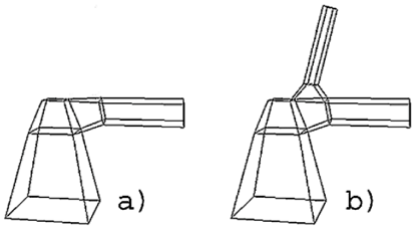


Figure C.2: Mesh production for direction above 90 degrees. a) first child added (direction of 91 degrees), b) next child

C.3.4 Connect Backward

When the first child segment direction is larger than 90 degrees compared to the present segments direction. special care has to be taken when producing the mesh (the main part of the structure bends backward). We build the segment out of two parts, where the last part is used to connect the child segments onto the polygon mesh.

The first step is to define two new planes. The *end plane* is defined along the direction of the segment at the distance that equal to the segments' length, plus the first child's radius (from the segments start position). The *middle plane* is defined at a distance equal to the diameter of the first child, along the negative of the present segments direction (from the segments end position). Two square cross sections are defined by projecting the *up* vector into the two newly defined planes. The cross section at the segments top can be closed with a quad surface, and the sides between the segments start and the middle cross section can also be closed with polygons. The sides between the middle and the top cross sections that has no child segments in its direction, can also be closed with polygons.

All child segment (even the first one) should be sorted into sets, defined by their direction compared to four direction. The directions are defined by the *middle* cross section, and each set should be handled as if they were sets of normal child segments, as described in Sect. C.3.5. Vertices from the newly defined *middle* and *end* cross sections are used to define the start cross sections (the four start vertices) for each of the new directions. An example can be seen in Fig. C.2.

C.3.5 Connect Branches

If there is more than one child segment, we start with the first child segment. The first segment has the largest diameter, and should normally have the smallest angle compared to its parent segment.

To connect the first branching child segment onto the mesh, we first use the same method as in section C.3.3. We make a square cross section at the end of the present segment, and its sides are closed by polygons. The distance from the segments start to the new cross section can be decreased to get a more accurate polygon mesh (for instance by decreasing the length by half of the calculated $l + x$ value found later in this section). In the example where vessels in a liver was visualized (Sect. C.4.2), the length was decreased as we just described.

A new square cross section is also defined along the *half direction* of the first child, starting at the center of the newly defined cross section. These two new cross sections defines a box, where the sides gives four cross sections (not the top or bottom side). The first child segment (and its children) are recursively added to the top of this box (on the cross section along the *half direction*), while the rest are added to the four sides. Note that the end position of a segment is calculated by vector algebra based on the parent segment's end position, and not on the cross section along the *half direction*. This means that the segment's length must be larger than the structure made at the branching point, to add the child.

When the recursion returns from adding the main child segment (and its child segments), the remaining child segments are sorted into four sets. The sorting is again done by the segments angle compared to the sides of the cross section around the present segment's end point. One must remember to maintain ordering by diameter while sorting.

The vertices at the corners of the two new cross sections defines a box where the sides can be used as new cross sections for each of the four directions (not the top and bottom sides). For each of the four directions, a new *up* vector is defined as the vector between the center of the directions cross section, and a corresponding vertex on the present segment's end cross

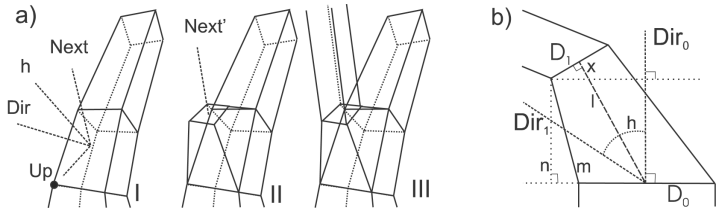


Figure C.3: a) Adding a second child segment. I) new *up*, *direction* and *half direction* vectors for this direction, II) first part of child segment, III) resulting mesh. b) Finding minimum distance m to move along the half vector to ensure space for any remaining branches (as seen from a right angle)

section. Figure C.3 a) shows the *up* and *half direction* when adding a child segment in one of the four directions.

The main problem one must solve is to find the distance along the *half vector* to move before defining the start cross section for the main child segment. This to ensure that there is enough space for any remaining child segments. If the distance moved is too small, the diameter of any remaining child segments will seem to increase significantly after the branching. A too large distance will result in a very large branching structure compared to the segments it connects.

Our main contribution is the automatic estimation of the distance to move, to allow space for any remaining child segments. In Fig. C.3 b), we can see the situation for calculating the displacement length m for a given half angle.

The length of m must at least be as large as the root of the $d_1^2 + d_2^2 ... + d_n^2$, where $d_1, d_2..$ are the diameter of the child segments. This because we know from the *Da Vinci rule* and murray's findings that every child segment at this point will have equal or smaller diameter then the parent segment (hence the sorting by diameter of child segments). Note that the *half angle (h)* will be less or equal to 45 degrees, as any larger angle will lead to the segment being handled as in section C.3.4 (as the main angle then will be larger then 90 degrees).

We could find the exact length of m, but observe that as long as the length of n is equal to d_1, we will have enough space along m. Setting $n = d_1^2 + d_2^2 ... + d_n^2$ gives C.4 for calculating the length to move along the *half vector*.

$$l + x = \sqrt{d_1^2 + d_2^2 + ... + d_n^2}/cos(h) + tan(h) * d_1/2 \qquad \text{(C.4)}$$

The error added by using $x + l$ instead of m, will introduce a small error in the mesh production. However, we observe that setting $n = d_1$ seems to give

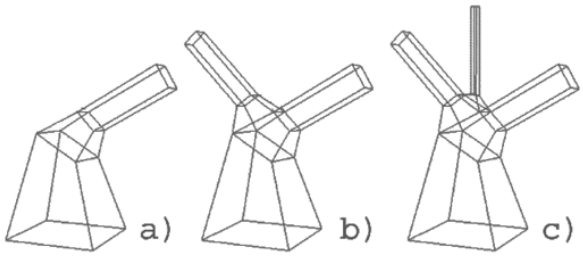


Figure C.4: The mesh production in a branching point. a) First child added, b) next child added, c) third child added.

adequate results for most cases. An example result from using the algorithm can be seen in Fig. C.4.

C.4 The Examples

A preliminary application has been produced in OpenGL to test the algorithm. This section shows this application at work. We have used it to produce polygon meshes for both naturally looking tree stems from a L-system generator, as well as meshes of the derived portal vein from a CT scan of a liver. Normal Catmull-Clark subdivision was used for the subdivision step.

C.4.1 Lindenmayer Generated Tree Stems

An extension to the application accepted an L-system string, representing a tree stem after a given number of Lindenmayer generation steps, as input. The extension interpreted the L-system string into a DAG that the application used to produce a base polygon mesh from. The application then subdivided this mesh to the level set by the user. An example with a shaded surface can be seen in Fig. C.5.

C.4.2 Delineation of Hepatic Vessels from CT Scans

Several processing steps has to be completed in order to visualize the hepatic vessels from a CT scan. In the preprocessing phase, histogram equalization [8] is first conducted to receive equal contrast on each image in the CT scan. Next, the blood vessels are emphasized using matched filtering [4]. After the preprocessing phase, the blood vessels are segmented using entropy based thresholding [10] and thresholding based on local variance with modifications

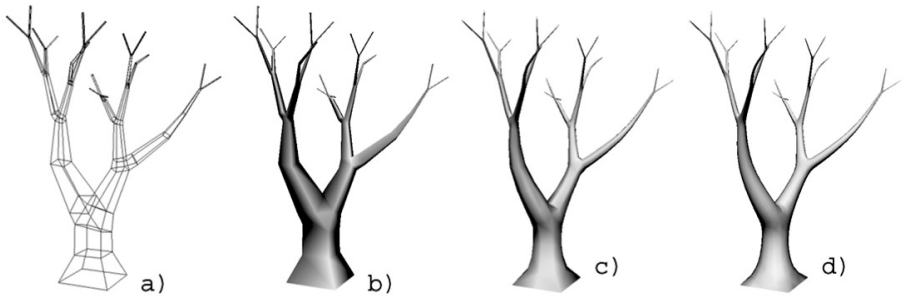


Figure C.5: A tree defined by a simple L-system. a) the produced mesh, b) shaded mesh, c) after one subdivision, d) after two subdivisions

using mathematical morphology [16]. A prerequisite to our method is that the liver is segmented manually beforehand.

After the vessel segments are found, the vessels' centers and widths must be calculated. These attributes are further used in a graph search to find the most likely vessel structure based on anatomical knowledge. First, the vessel centers are derived from the segmentation result using the segments' skeletons [16]. The vessels' widths are next computed from a modified distance map [8] of the segmented images.

The last step before the vessel graph can be presented is to make connections between the located vessel centers. Centers within segments are interconnected directly. On the other hand, interconnections between adjacent CT slices are not as trivial. Here, as previously mentioned, we use cost functions representing anatomical knowledge in a graph search for the most likely interconnections [5]. The resulting graph is finally visualized using the outlined algorithm in this paper.

A few modification to the existing graph is made in order to make it more visually correct. First, nodes with two or more interconnected neighbors have their heights averaged since the resolution in the y-direction is normally lower than that in the image plane. Second, if two interconnected nodes are closer than a predefined limit, the two nodes are replaced by one node positioned between them. Fig. C.6 shows the resulting visualization of the derived portal vein from a CT scan of a liver.

C.5 Findings

Our method for automatically calculating the distance for sufficient space for any remaining child segments after the first child segment has been added,

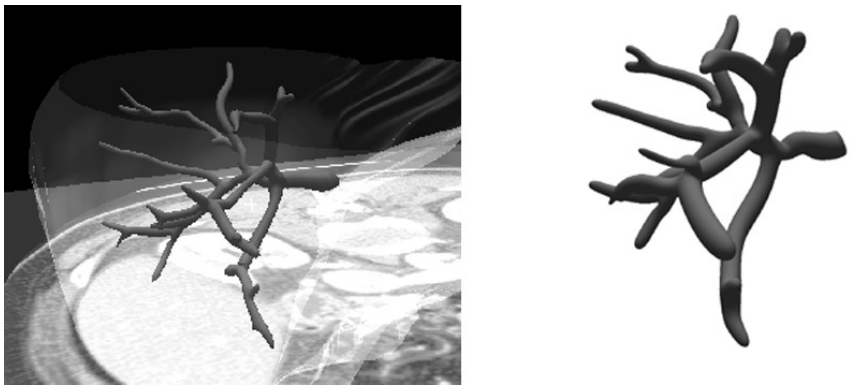


Figure C.6: Left: Portal vein visualized from a CT scan of a liver (the CT scan data can be shown at the same time for any part of the liver, for visual comparison). Right: The same structure withouth the scan data.

seems to produce good results. The preliminary results from our method applied to visualization of hepatic vessels in the liver gives good results when compared with the CT data they are based on, but these results have only been visually verified (however the first feedbacks from the Intervention Centre at Rikshsopitalet in Norway has been promising). However, a more throughout comparison with existing methods, and verification against the data set values should be completed before using the method in clinical applications.

The algorithm is fast and simple, and can be used by most modern PC's with a graphic accelerator. The meshing algorithm mostly does its work in real-time, but the subdivision steps and any preprocessing slow things down a bit. Graphic hardware support for subdivision will hopefully be available in the relative near future. When this happens, the subdivision of the branching structures may become a viable approach even for large amounts of trees or blood vessels in real-time computer graphics.

Bibliography

[1] Bloomenthal J.: Modeling the mighty maple. Computer Graphics 19, 3 (July 1985) 305–311

[2] Bloomenthal J., Wyvill B.: Interactive techniques for implicit modeling. Computer Graphics 24, 2 (Mar. 1990) 109–116

[3] Catmull E., Clark J.: Recursively generated b-spline surfaces on arbitrary topological meshes. Computer Aided Design 10, 6 (1978) 350-355

[4] Chaudhuri S., Chatterjee S., Katz N., Nelson M., Goldbaum M.: Detection of blood vessels in retinal images using two-dimensional matched filters. IEEE Transactions on Medical Imageing 8, 3 (1989) 263–269

[5] Eidheim O. C., Aurdal L., Omholt-Jensen T., Mala T., Edwin B.: Segmentation of liver vessels as seen in mr and ct images. Computer Assisted Radiology and Surgery (2004) 201–206

[6] Felkel P., Fuhrmann A., Kanitsar A., Wegenkittl R.: Surface reconstruction of the branching vessels for augmented reality aided surgery. Analysis of Biomedical Signals and Images 16 (2002) 252-254 (Proc. BIOSIGNAL 2002)

[7] Felkel P., Kanitsar A., Fuhrmann A. L., Wegenkittl R.: Surface Models of Tube Trees. Tech. Rep. TR VRVis 2002 008, VRVis, 2002.

[8] Gonzalez R. C., Woods R. E.: Digital Image Processing, second ed. Prentice Hall, 2002.

[9] Hart J., Baker B.: Implicit modeling of tree surfaces. Proc. of Implicit Surfaces 96 (Oct.1996) 143–152

[10] Kapur J. N., Sahoo P. K., Wong A. K. C.: A new method for gray-level picture thresholding using the entropy of the histogram. Computer Vision, Graphics, and Image Processing 29 (1985) 273–285

[11] Lluch J., Vicent M., Fernandez S., Monserrat C., Vivo R.: Modelling of branched structures using a single polygonal mesh. In Proc. IASTED International Conference on Visualization, Imaging, and Image Processing (2001).

[12] Maierhofer S.: Rule-Based Mesh Growing and Generalized Subdivision Meshes. PhD thesis, Technische Universitaet Wien, Technisch-Naturwissenschaftliche Fakultaet, Institut fuer Computergraphik, 2002.

[13] Murray C. D.: A relationship between circumference and weight in trees and its bearing in branching angles. Journal of General Phyiol. 9 (1927) 725–729

[14] Oppenheimer P. E.: Real time design and animation of fractal plants

and trees. Computer Graphics 20, 4 (Aug. 1986) 55–64

[15] Richter J. P.: The notebooks of Leonardo da Vinc Vol. 1. Dover Pubns., 1970.

[16] Soille P.: Morphological Image Analysis. Springer-Verlag, 2003.

[17] Tobler R. F., Maierhofer S., Wilkie A.: A multiresolution mesh generation approach for procedural definition of complex geometry. In Proceedings of the 2002 International Conference on Shape Modelling and Applications (SMI 2002) (2002) 35–43