



# Object-based Image Analysis for Detection and Segmentation Tasks in Biomedical Imaging

by

**Michael Schwier**

A thesis submitted in partial fulfillment
of the requirements for the degree of

*Doctor of Philosophy*
*in*
*Computer Science*

**Dissertation Committee:**

Prof. Dr. Horst Karl Hahn (PhD Advisor)
Jacobs University Bremen · Fraunhofer MEVIS

Prof. Dr. Herbert Jaeger
Jacobs University Bremen

Prof. Dr. Gitta Domik-Kienegger
University of Paderborn

Date of Defense: July 14, 2016

Jacobs University Bremen

# Abstract

Object-based image analysis (OBIA) is a concept for analyzing images based on regions instead of pixels. OBIA allows to effectively incorporate features of regions as well as their contextual and hierarchical relations into the analysis process. While object-based image analysis is common in the field of geographic information science and remote sensing, it has rarely found its was into biomedical image analysis. This thesis explores the applicability and capabilities of OBIA for addressing image processing and analysis tasks on biomedical images, by approaching several relevant and challenging tasks from different imaging domains.

To begin with, a formalization as well as a powerful and flexible implementation of the OBIA concept is proposed. This is the foundation on which the applications are based.

The first application that is approached concerns the detection of the spine and vertebrae in CT images. A sophisticated processing pipeline is presented that uses hand-crafted rules in an explicit OBIA approach. The method effectively utilizes geometrical and contextual patterns which are particularly apparent in the spine. Furthermore, an alternative implementation is presented, in which crucial steps and rules in that pipeline are replaced by stages based on supervised machine learning. By comparing these two different OBIA-based approaches on the same task, we can learn the advantages and challenges of either of them.

Automatic detection of pregnancy in pigs on ultrasound images is the second application. Using OBIA, we build several image processing stages, first, to enable the automatic acquisition of the ultrasound images by tracking the animal and the remote controlled transducer on video images, and second, to automatically analyze the ultrasound images for signs of pregnancy. A smart segmentation hierarchy and dedicated features combined in a machine learning approach are the key components for the latter stage. The resulting overall system is—to our best knowledge—a world novelty and became commercially available.

The third application aims at the reconstruction of vessels from histological whole slide sections of murine liver samples. We present a two stage OBIA classification scheme that effectively deals with the challenges of different stainings, which induce a high variability in visual appearances of histological images. A multi-scale approach is proposed to reduce computation time. Since whole slide images are inherently 2D, a registration method is presented, to reconstruct a virtual 3D histological image as a prerequisite for the vessel

reconstruction algorithm.

One of the major contributions of this thesis is the demonstration of the capability and applicability of OBIA to tackle biomedical image analysis problems. Together with the OBIA foundation, the presented solutions for three different applications reveal the strength of OBIA, but also some challenges. Furthermore, the developed algorithms also pose a valuable scientific contribution in their own right, with some of them even presenting world-novel algorithms that have already found their way into commercial application.

# Preface

All things considered, the story of this thesis began about nine years ago. Not that I had started my PhD right away or that I had consistently worked on it since then—not at all. But back then I started to work at MeVis Research (before it became Fraunhofer MEVIS) with the thought, "I could imagine to do a PhD one day." Not really determined yet I admit, but with time and progress in my research work it turned into, "I think I'd really like to do a PhD," and eventually into, "I am gonna do a PhD." Well, let's not fool anyone, there were also quite a few, "Argh, I am never going to finish this PhD!" and, "Why am I doing this to myself?" phases in between.

That I really made it until the end is also thanks to many people who supported me in so many different ways—from the personal level to the professional level. So to all of you who were there for me: When you read these lines, you should know I mean you and that I am deeply grateful. Thank you so much ...

# Contents

# Chapter 1

# Introduction

Nowadays computer assistance is common in medical practice. From the acquisition of images such as computed tomography (CT) or magnetic resonance imaging (MRI) to the diagnosis as well as planning of surgeries and interventions, image processing and analysis methods play an important role. However, most of the software assistants employed in clinical practice are manual or semi-automatic tools. The development of fully automatic detection and recognition algorithms remains one of the most challenging tasks in medical image analysis.

It is striking how hard it is to teach a computer to recognize structures, which for human vision do not pose any particular difficulties. To why it is so difficult, Szeliski [158] states that "In part, it is because vision is an *inverse problem*, in which we seek to recover some unknowns given insufficient information to fully specify the solution." Palmer [110] notes, "one of the most important facts about vision is that despite the logically certain conclusion that optical information is insufficient to solve the inverse problem uniquely, our visual systems manage to arrive at the right solutions with remarkable regularity" and mentions that "most perceptual theorists have concluded that there must be some additional source of information [...] that is used in the process of seeing."

A groundbreaking theory on how the human vision achieves this was introduced by Helmholtz [54], who described human perception as an experience-based unconscious inductive process, inferring the most likely environmental situation given a visual image. Helmholtz's basic principle endured until today [110] and influenced for example Cutting [29] who describes "hidden premises" as driving factor for a deductive inferring process he calls "directed perception".

What we can now learn from these foundational theories is that for solving vision problems computationally we need to introduce some form of models or model knowledge to "disambiguate between potential solutions" [158].

However, there are more aspects to be considered, regarding the representation of an image as an array of pixels. Pixels themselves are a very limited source of information since they only contain color information and an absolute position. The goal of any computer vision approach must be to overcome this limitation. In his fundamental work on vision

David Marr [84] describes the process of building up a complex representation of an image. Thus an additional explanation for the advantage of human vision is that it is based on a different perception concept: We perceive an image neither as a whole nor as an array of pixels, but instead we identify individual objects (even only abstract shapes/regions) as well as a hierarchy of those objects and evaluate them in their context.

Thus, to be able to transfer those perception concepts—at least to a certain extent—to computerized image processing, a different approach from the classical pixel-based processing has to be taken. Object-based image analysis (OBIA) is a concept that aims to achieve that. With OBIA we leave the limitations of pixel-based processing and instead analyze an image based on regions and their semantic context, which comes closer to the way humans comprehend it.

While object-based image analysis is quite common in the field of geographic information science and remote sensing [10], it has rarely found its way into medical image analysis. The goal of this work is thus to explore the capabilities of OBIA for biomedical applications. As we will see in Chapter 2, not only the range of possible applications within this domain is large, but one can also use different basic approaches within the OBIA methodology. To appreciate the diversity of possibilities and to be able to explore the suitability of OBIA for it, three tasks are approached, all with quite different goals and incorporating different image modalities. The first task is the detection of the spine and vertebrae in computed tomography images. The second application is a system for automatic pregnancy detection in pigs, which involves ultrasound images as well as real world video images. The third task covers the topic of vessel segmentation in histological whole slide images of murine liver samples. All three topics are good candidates for an OBIA-based approach for different reasons and on different levels. They will be further introduced in Chapter 2 in relation to OBIA in general, while Chapter 3, 4, and 5 cover each application task in detail.

This thesis also presents the foundation for the application developments: The OBIA framework itself. Chapter 2 contains a formalization of the OBIA concept as well as an overview of the most crucial and novel aspects of its implementation.

## 1.1 A Note on the Style of Writing

For the most part of this thesis I follow the academic habit of writing “we” instead of using “I”, even if the latter would be semantically correct. One reason is that the main chapters are partly based on papers, which were written just like that already. Furthermore, to my ears it sounds more humble, does not distract the attention from the content and also includes you, the reader of this thesis.

## 1.2 Contributions

The following gives a brief overview of my main contributions contained in this thesis:

- Object-Based Image Analysis
  - Concept and implementation of the OBIA framework*
  - Formalization of the OBIA concept
  - Context Features
- Spine and Vertebrae Detection in CT
  - Algorithms for detecting the spine and vertebrae in two methodological variants (implicit/explicit) and their evaluation and comparison
- Ultrasound-based Automatic Pregnancy Detection in Pigs
  - Algorithm for the detection of embryonic vesicles and its evaluation
  - Algorithms for uterus and ultrasound-probe tracking on video images and their evaluation
  - Algorithm to detect dirt on the camera pane and its evaluation
- Histological Vessel Reconstruction on Murine Liver Samples
  - Algorithm for the rigid registration of histological whole slide images and its evaluation
  - Algorithm to detect and reconstruct vessel structures from histological whole slide image stacks and its evaluation
- Overall
  - Exploration and assessment of capabilities and applicability of OBIA for biomedical image analysis

* **Note:** The development of the methodological concept for OBIA and the implementation of the framework itself—in form and scope relevant and used in this thesis—was a joint effort of André Homeyer, Teodora Chitiboi and me. For the most part it is impossible to separate individual contributions to the OBIA framework, since developing it was truly a team effort. Long conceptual discussions, redesign and refactoring steps made it what it is today. Hence the contribution of the *Object-Based Image Analysis* chapter should be considered a joint contribution, with two exceptions: The formalization of the OBIA concept and the Context Features are my sole contributions.

# Chapter 2

# Object-Based Image Analysis

**Acknowledgments** Developing a methodological concept and framework for object-based image analysis at our institute was motivated by our needs to handle image regions and their features efficiently, and to perform sophisticated classification and reasoning tasks on them. Conventional image processing approaches could not provide this. The work was started by my colleague André Homeyer and me. Shortly after, Teodora Chitiboi joined the team. While up to date several other colleagues and students contributed to our OBIA framework, the foundation and the parts that are relevant for this thesis were basically created by us three or implemented by students under our supervision.

For the most part it is impossible to separate individual contributions to our OBIA framework since developing it was truly a team effort. Long conceptual discussions, redesign and refactoring steps made it what it is today. Hence what I describe in this chapter is my view on what has been a joined endeavor for several years and is still ongoing.

**Publications** A paper on our generic implementation of object-based image analysis was published in the *Proceedings of the International Conference on Computer Vision Theory and Applications* [58]. Furthermore my work on detecting hypodense lesions in liver CT partly inspired and motivated the development of our OBIA framework, even though in its final version it contained only intermediate approaches of what should become the OBIA framework. It was published in the *International Journal of Computer Assisted Radiology and Surgery* [141]. Parts of this chapter are based on an introduction to OBIA which was part of our paper on *Automated Spine and Vertebrae Detection in CT Images using Object-based Image Analysis* published in the *International Journal for Numerical Methods in Biomedical Engineering* [137] ©2013 John Wiley & Sons, Ltd. The description of the *Context Features* is based on the respective section of our paper on *Segmentation of Vessel Structures in Serial Whole Slide Sections using Region-based Context Features* published in the *Proceedings of SPIE Medical Imaging 2016* [140] ©2016 SPIE.

The representation of an image as an array of pixels (color/intensity values) is merely dictated by the capturing and storing device but is not inherently made for image or vision processing. An essential part of vision, as also stated by Marr [84], is the transformation of the initial image representation into a description that exhibits useful information.

The idea of object-based image analysis (OBIA) is to partition the image into regions which become the base units for the analysis. Contrary to pixels, regions exhibit a wealth of properties such as intensity statistics or shape features. An object in the sense of object-based image analysis is an abstract representation of such an image region and its features. Not only do the objects themselves bear semantic information, but also the spatial context is considered during the object-based analysis. This is an important advantage, since the context and spatial relations between objects are essential information for recognition (see e.g. Marr [84] or Torralba [163]). Based on those initial objects and their features the basic OBIA processing is an iterative process of semantically identifying and merging regions into more meaningful objects which exhibit different features and appear in different contextual relations to each other. Thus, an OBIA analysis may create multiple dynamic hierarchies of objects and super-objects which are supposed to lead to recognizing the target structures.

The term "object-based image analysis" originated in the field of geographic information science [144] and has been rapidly gaining significance in this discipline over the last 15 years [10]. Hay and Castilla [52] tried a first definition of object-based image analysis and discussed its strengths and weaknesses. Blaschke [10] presents a comprehensive overview of publications on the application of OBIA approaches in the remote sensing field. He also mentions that many of these works are based on a software product developed by the company Definiens. Lang and Tiede [69] describe this software platform for object-based image analysis which provides a graphical user interface for developing classification rules. However, the software is limited in terms of flexibility regarding querying, modifying and extending the data model as well as integrating it into existing image processing platforms.

Furthermore, several works were published following the same or similar basic principles without explicitly using the term "object-based image analysis", like the approach to model search in images by Mori [94] or the recent work of Peng et al. [113] on automatic image segmentation. We are using the the term "object-based image analysis" because we believe that it describes very well the basic principle.

Also the GeoMap as presented by Meine [87] provides a framework for OBIA-like image analysis and goes a step further in its concept for segmentation representation by offering a combination of boundary- and region-based representations in a unified approach.

Further approaches that follow a similar concept are described by Maillot et al. [82] and Renouf et al. [124]. However, their concept differs from ours, since they follow a top-down approach: A domain expert provides an abstract concept definition, and the required segmentation and feature calculations are automatically derived from that.

We believe, though, that in general it is not feasible to match a concept based on real word experience directly to features derived from an image. Our OBIA concept follows

a bottom-up approach, which requires to consider the artificial nature of the features we are actually able to calculate on images and to build reasoning concepts based on that. However, OBIA does not strictly enforce a bottom-up strategy all the way. The dynamic hierarchical structure that is created also allows higher level information to influence or refine interpretations on a lower level. This actually complies with the modern understanding of most visual science theorists regarding vision being a bi-directional process [110].

In this chapter we present an OBIA framework that we have developed over the course of several years. This framework is the basis for all applications of OBIA in this thesis. In the following sections we will present the theoretical foundation and notation of our object-based image representation. We will also look into basic prerequisites and the principal methodological approach for working with OBIA in practice. Furthermore we will define all object features that are included in OBIA and are used in the methods of the later chapters. This chapter concludes with a short discussion of the applicability of the OBIA concept to tasks in biomedical image analysis, which also introduces the OBIA application chapters.

## 2.1 Object-based Image Representation

This section will give a brief formalization of OBIA in the way we understand it and implemented it. For the representation of images in an object-based manner we employ the general concept of the *attributed relational graph* (ARG) data model, which has already been successfully used to represent image content [21, 116, 3].

Our object-based image representation is defined as follows: Let $D = (O, A)$ be a directed graph, where $O$ is a set of nodes in which each $o_i \in O$ corresponds to one region of the initial over-segmentation. $A$ is a set of arcs (directed edges), where an arc $a_{i,j} \in A$ is defined as $a_{i,j} = (o_i, o_j)$ and is considered to be directed from $o_i$ to $o_j$. In the terminology of object-based image analysis we refer to nodes $o_i \in O$ as "objects" and to arcs $a_{i,j} \in A$ as "relations" between objects.

Furthermore, the set $A$ comprises of two subsets $N$ and $M$ such that $A = N \cup M$ and $N \cap M = \emptyset$ (with $\emptyset$ being the empty set). The subset $N$ represents neighbor relations between two regions such that $n_{i,j} \in N$ exists if $o_i, o_j \in O$ correspond to image regions that on the pixel level are connected via a 26-neighborhood in the 3D case, or 8-neighborhood in the 2D case. Relations in $N$ are always bidirectional, thus $\forall\ n_{i,j} \in N, \exists\ n_{j,i} \in N$. The subset $M$ represents hierarchical relations. In the object-based image representation, new objects can be formed by merging existing objects. A relation $m_{i,j} \in M$ indicates that $o_j$ is a part of the merged object $o_i$. Thus a merged object $o_i$ is defined by its sub-objects via a set of relations $\{m_{i,j}, .., m_{i,k}\} \subset M$.

Each $o_i \in O$ and each $a_{i,j} \in A$ may hold an arbitrary number of attributes describing its features. We will use the notation $o_i.featurename$ for attributes of objects and $a_{i,j}.featurename$ for attributes of relations. Our object-based image representation deliberately allows an arbitrary number of features, since it always depends on the task and

implementation, which features are actually required.

### 2.1.1 Notes on the Implementation of OBIA

Developing recognition algorithms that use the presented object-based image representation means to analyze and modify this graph structure in order to identify sub-graphs as the target structures we are looking for. For that it is important to be able to efficiently access and query the data structure. We developed a generic implementation for object-based image analysis [58], based on the representation described above. In the following we will give a short overview of the most important aspects of this implementation.

Since our OBIA data structure is an ARG it can be expressed in terms of the relational data model [21], which is the foundation of relational databases. Relational databases are a well established technology and have the valuable properties of being scalable beyond the limits of working memory, providing data persistence, and enabling fast and complex queries on the basis of the Structured Query Language (SQL). Therefore for data management our OBIA implementation is based on SQLite (www.sqlite.org), a public domain relational database implementation.

To make our implementation easily expandable we do not enforce a fixed database schema but rather establish a set of conventions for storing three types of data in three types of database tables: objects, relations and attributes.

Object tables store the identifiers of objects from $O$ and only contain the ID of all objects. Relation tables store the relations from $A$ and contain the source and destination object IDs. Separate tables are used for relations from $N$ and $M$. Especially the latter set is subdivided into separate tables for each hierarchy level. Attribute tables store the feature values for objects and relations. Object attributes are identified by the object ID as key, relation attributes are identified by the combination of source and destination object IDs.

For the purpose of image analysis it is not only necessary to efficiently query objects by their IDs, features or relations, but also to query objects by their location. To allow this the bounding box coordinates of all objects are implicitly stored in a special R-Tree data structure [46], which enables fast spatial queries.

All further OBIA functionality like feature extractors and classification functionality is implemented in C++ for computational efficiency. A simple interface allows for integrating OBIA into virtually any image processing platform. For our work we integrated it already into the MeVisLab image processing platform [126].

## 2.2 Object-based Image Analysis in Practice

As indicated before, the prerequisite for object-based image analysis is an appropriate partitioning of the image into regions—a so called over-segmentation. Typical properties required for the regions (segments) created by such an over-segmentation are homogeneity

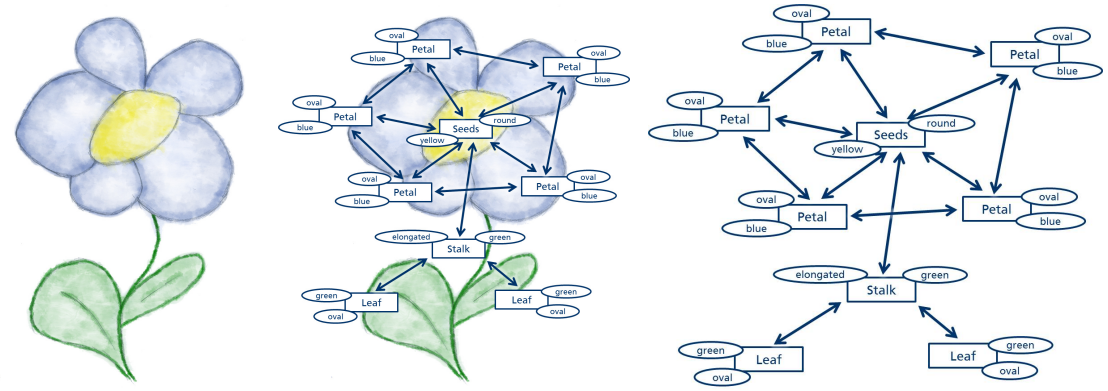



**Figure 2.1:** Illustration of the principle of an OBIA representation of a complex object: (a) Original object. (b) Original object and its OBIA representation as attributed relational graph. (c) OBIA representation of the object as attributed relational graph.

in color/intensity or texture and compactness of the regions, while the borders of the regions should also respect edges in the image. For this an adaptation of the watershed transformation [166] or a region merging method such as the one described by Redding et al. [122] could be feasible, preceded by appropriate image filtering. Furthermore, under the term *superpixels* several methods were introduced, specifically aiming to provide over-segmentations for the aforementioned purpose, like the works of Ren and Malik [123], Levinshtein et al. [72], Moore et al. [93] and Achanta et al. [2]. However, the appropriate over-segmentation algorithm has to be selected depending on the task and the image type, and more customized segmentation methods might also be considered.

To develop OBIA algorithms we can basically distinguish two approaches. The first one is to explicitly formulate rules (explicit approach). The OBIA representation as described in section 2.1 allows for this. This is a particular strength of OBIA above pixel-based processing: Given countless features to describe object properties as well as relations of objects to each other, we can formulate such rules almost in a way humans would describe a complex object in terms of its more simple parts (see Figure 2.1).

The second approach is to use machine learning techniques to classify the object space (implicit approach). Depending on the number of features and their dependencies, setting up rules and thresholds can quickly become too complex to handle manually. Training a classifier with reference samples is much more promising in such cases.

In both approaches a key element for success is the choice of features. The features have to be selected or designed in such a way that they are capable of capturing the discriminating properties of the target structures. This can be by a combination of common features as well as specifically designed features for a given domain or target structure.

In regard to the model knowledge mentioned earlier, the OBIA representation must enable us to express this model knowledge—explicitly or implicitly—through a learning

process. For this not only object features play an important role but also the spatial and hierarchical relations that are captured. And since the sole fact that a particular relation exists might not alone be very helpful, another particular strength of OBIA is that all relations can have features describing a relation's properties in more detail.

As we can see, OBIA is not a new all-in-one solution to replace classic image processing but rather builds on existing technologies such as segmentation algorithms, feature extractors and classification concepts [10]. OBIA is supposed to blend in with those technologies and expand the possibilities of image analysis where it is limited by pixel-based approaches. For practical applications this means that the OBIA approach cannot usually be considered without regard to the overall image processing and analysis setup in which it is integrated.

Since the OBIA representation is rather complex we developed inspection tools that can be used in MeVisLab [126] to explore the OBIA data space and visualize it in combination with the original image. Furthermore we used the data mining tools WEKA [172] and RapidMiner [89] as well as the Python-based machine learning library and Scikit-learn [112, 16] for data analysis and evaluations.

## 2.3 Features

In this section we describe all features that are part of our OBIA implementation and are relevant for this thesis. Note that we do not cover all feature groups that are included in the full OBIA framework, but only those which we actually used in the development of the methods described in the later chapters. Furthermore some task-specific features, which were developed for a particular application, will not be covered here but in the relevant application chapter.

### 2.3.1 Intensity Features

For an object $o_i \in O$ the intensity values of all pixels in the region that corresponds to $o_i$ are considered. Intensity features are calculated on one color channel only. If an image has more than one color channel, the desired color channel has to be specified in parentheses. For example $o_i.mean(R_{RGB})$ refers to the mean value in the red channel of an RGB coded color image.

Table 2.1 summarizes the definitions of intensity-based features.

### 2.3.2 Shape Features

Shape features are based on the mask of the region that corresponds to $o_i$. We define two groups of shape features: basic shape features and MBR shape features.

**Table 2.1:** Definitions for intensity features.

| **Intensity Features** | |
|---|---|
| Feature Name | Definition |
| $o_i.lowerQuartile$ | The lower quartile or 25th percentile of the intensity distribution. |
| $o_i.upperQuartile$ | The upper quartile or 75th percentile of the intensity distribution. |
| $o_i.median$ | The median or 50th percentile of the intensity distribution. |
| $o_i.min$ | The minimum intensity value. |
| $o_i.max$ | The maximum intensity value. |
| $o_i.mean$ | The mean intensity value. |
| $o_i.stdDev$ | The standard deviation of the intensity distribution. |
| $o_i.sum$ | The sum of all intensity values. |

#### 2.3.2.1 Basic Shape Features

In table 2.2 the basic shape features for an object $o_i \in O$ and their definitions are summarized. For the definitions some intermediate parameters are required, which we will describe in the following:

We define the covariance matrix of $o_i$ as

$$cov(o_i) = \begin{bmatrix} \mu_{20}(o_i) & \mu_{11}(o_i) \\ \mu_{11}(o_i) & \mu_{02}(o_i) \end{bmatrix} \tag{2.1}$$

with $\mu_{pq}(o_i)$ being the central moments of the binary image region that corresponds to $o_i$. For details on central moments see e.g. Burger and Burge [17]. Hence we can derive the eigenvalues $\lambda_1$ and $\lambda_2$ of $cov(o_i)$ as

$$\lambda_1 = \frac{\mu_{20}(o_i) + \mu_{02}(o_i) - \sqrt{(\mu_{20}(o_i) - \mu_{02}(o_i))^2 + 4 \cdot \mu_{11}^2(o_i)}}{2} \tag{2.2}$$

$$\lambda_2 = \frac{\mu_{20}(o_i) + \mu_{02}(o_i) + \sqrt{(\mu_{20}(o_i) - \mu_{02}(o_i))^2 + 4 \cdot \mu_{11}^2(o_i)}}{2} \tag{2.3}$$

#### 2.3.2.2 MBR Shape Features

The MBR shape features are based on the minimum bounding rectangle (MBR) of an object $o_i$. To compute the MBR we use the rotating calipers algorithm introduced by Toussaint [164]. Toussaint's algorithm requires as input a list of vertex points of the convex hull of the object. Therefore we employed Graham's scan algorithm [45] to calculate the convex hull of the segmentation region that corresponds to $o_i$. Table 2.3 lists the features we derive from the MBR of an object.

**Table 2.2:** Definitions for basic shape features.

| **Basic Shape Features** | |
|---|---|
| Feature Name | Definition |
| $o_i.centerX$, $o_i.centerY$, $o_i.centerZ$ | Coordinates of the center of gravity. |
| $o_i.circularity$ | $\frac{4\pi \cdot o_i.size}{o_i.perimeter^2}$ based on the isoperimetric inequality. |
| $o_i.eccentricity$ | $\frac{[\mu_{20}(o_i) - \mu_{02}(o_i)]^2 + 4 \cdot [\mu_{11}(o_i)]^2}{[\mu_{20}(o_i) + \mu_{02}(o_i)]^2}$ from Burger and Burge [17]. |
| $o_i.ellipticity$ | $\frac{\sqrt{\pi \cdot o_i.size \cdot Rm^2}}{o_i.perimeter^2 \cdot \frac{\lambda_1}{\lambda_2}}$ with $Rm = 3 \cdot (1 + \frac{\lambda_1}{\lambda_2}) - \sqrt{(3\frac{\lambda_1}{\lambda_2} + 1) \cdot (\frac{\lambda_1}{\lambda_2} + 3)}$ based on Ramanujan's elliptic perimeter approximation [119]. |
| $o_i.elongation$ | $1 - \sqrt{\frac{\lambda_1}{\lambda_2}}$ |
| $o_i.perimeter$ | Perimeter in mm estimated by a method proposed by Lindblad [73], which is based on marching squares using optimized weights for each marching squares configuration. |
| $o_i.princX$, $o_i.princY$ | Components of the eigenvector corresponding to $\lambda_2$. |
| $o_i.size$ | Area of the object in $mm^2$ based on the pixel count and pixel size of the object's segmentation mask. |
| $o_i.slopeXY$ | $\text{atan2}(\mid o_i.princY \mid, \mid o_i.princX \mid) \cdot \frac{180.0}{\pi}$ |

**Table 2.3:** Definitions for MBR shape features.

| **MBR Shape Features** | |
|---|---|
| Feature Name | Definition |
| $o_i.angle_{MBR}$ | $\text{atan2}(\vec{c_{1,2}}.y, \vec{c_{1,2}}.x) \cdot \frac{180.0}{\pi}$ with $\vec{c_{1,2}} = \begin{pmatrix} o_i.corner2x - o_i.corner1x \\ o_i.corner2y - o_i.corner1y \end{pmatrix}$ |
| $o_i.aspectRatio_{MBR}$ | $\frac{o_i.width_{MBR}}{o_i.height_{MBR}}$ for $o_i.width_{MBR} \leq o_i.height_{MBR}$<br>$\frac{o_i.height_{MBR}}{o_i.width_{MBR}}$ for $o_i.width_{MBR} > o_i.height_{MBR}$ |
| $o_i.corner1x$, $o_i.corner2x$, $o_i.corner3x$, $o_i.corner4x$, $o_i.corner1y$, $o_i.corner2y$, $o_i.corner3y$, $o_i.corner4y$ | Coordinates of the four corners of the MBR. To determine the indices for the corners we consider the highest corner point. If to the left (negative x-direction) of this highest point only one or no other corner point is found, the highest corner point is considered to be corner 1. Otherwise it is considered to be corner 2. The remaining indices continue in a clockwise order. |
| $o_i.height_{MBR}$ | $\sqrt{(o_i.corner3x - o_i.corner2x)^2 + (o_i.corner3y - o_i.corner2y)^2}$ |
| $o_i.rectFit_1$ | $\frac{o_i.size}{o_i.size_{MBR}}$ |
| $o_i.size_{MBR}$ | $o_i.height_{MBR} \cdot o_i.width_{MBR}$ |
| $o_i.squareFit_1$ | $\frac{o_i.size}{\left(\frac{o_i.height_{MBR} + o_i.width_{MBR}}{2}\right)^2}$ |
| $o_i.squareFit_2$ | $\frac{o_i.size_{MBR}}{\left(\frac{o_i.height_{MBR} + o_i.width_{MBR}}{2}\right)^2}$ |
| $o_i.width_{MBR}$ | $\sqrt{(o_i.corner1x - o_i.corner2x)^2 + (o_i.corner1y - o_i.corner2y)^2}$ |

### 2.3.3 Texture Features

As texture features we employ the local binary patterns presented by Ojala et al. [107]. We use the $LBP^{riu2}_{P,R}$ operator from the paper with $P = 8$ and $R = 1$. The operator returns a feature vector of ten values describing the texture of an object $o_i \in O$. We denote this feature vector as $o_i.localBinaryPattern$.

As with the intensity features the local binary patterns are computed on one color channel. In case the image has more than one color channel, the desired one has to be specified in parentheses. For example $o_i.localBinaryPattern(V_{HSV})$ refers to the local binary patterns computed on the value channel of an HSV coded color image.

### 2.3.4 Border Relation Features

The border relation features describe properties of the border between two objects. Hence they are defined for relations $n_{i,j} \in N$. Their calculation is based on the masks of the regions that correspond to $o_i, o_j \in O$. Table 2.4 lists the border relation features.

There are two basic types of border relations features. One is simply based on counting the pixel or voxel sides of an object. While fast and easy to calculate, due to the binary representation of the object it usually does not properly reflect the implied boundary or surface as perceived by a human. Therefore we also provide a measure based on a surface estimator introduced by Lindblad [73], who used marching squares/cubes [75] configurations and provided optimized boundary/surface estimates for them.

### 2.3.5 Context Features

The idea behind context features is that a description of the neighborhood of an object might give valuable information about the nature of the object itself. For this we developed a general descriptor which captures the neighborhood pattern regarding a selected feature. Here we are only defining in-plane 2D context features. For the following explanations on the computation of the context features consider the supporting illustration in Fig 2.2.

To calculate the context feature for an object $o_i \in O$ we divide the circular neighborhood of $o_i$ into eight bins, each spanning 45°. For each pixel directly neighboring $o_i$ we calculate the angle to the center of gravity of $o_i$ to determine to which bin it should be assigned. Then the corresponding ID of the neighboring object is added to the bin. Hence, each bin captures the ratio of neighboring objects along the border within the degree range of the bin. Let $\omega_{i,b}(o_j)$ be the ratio of object $o_j$ in $bin_b$ of object $o_i$. For example for $bin_5$ in our illustration (Fig. 2.2) we would determine $\omega_{i,5}(o_{blue}) = 33\%$ and $\omega_{i,5}(o_{orange}) = 67\%$. Those ratios are used as weights in the following computation.

A context feature is now represented as three feature vectors, each with eight entries (one for each bin). The vectors represent the minimum, maximum and weighted average values of another feature for each bin. We denote the whole context feature as $contextPattern(inputFeature)$ where $inputFeature$ can be any other already existing

**Table 2.4:** Definitions for border relation features.

| **Border Relation Features** | |
|---|---|
| Feature Name | Definition |
| $n_{i,j}.border2dSize$ | Sum of all outer sides of border pixels of $o_i$ which touch a border pixel of $o_j$. Scale in mm. |
| $n_{i,j}.border2dRelative$ | $n_{i,j}.border2dSize$ divided by the sum of all outer sides of border pixels of $o_i$. |
| $n_{i,j}.border3dSize$ | Sum of all outer sides of border voxels of $o_i$ which touch a border voxel of $o_j$. Scale in $mm^2$. |
| $n_{i,j}.border3dRelative$ | $n_{i,j}.border3dSize$ divided by the sum of all outer sides of border voxels of $o_i$. |
| $n_{i,j}.mcBorder2dSize$ | Size in mm of the border shared between $o_i$ and $o_j$ estimated by a method proposed by Lindblad [73]. |
| $n_{i,j}.mcBorder2dRelative$ | $n_{i,j}.mcBorder2dSize$ divided by the size of the full 2D boundary estimated by a method proposed by Lindblad [73]. |
| $n_{i,j}.mcBorder3dSize$ | Size in $mm^2$ of the border shared between $o_i$ and $o_j$ estimated by a method proposed by Lindblad [73]. |
| $n_{i,j}.mcBorder3dRelative$ | $n_{i,j}.mcBorder3dSize$ divided by the size of the full 3D boundary estimated by a method proposed by Lindblad [73]. |

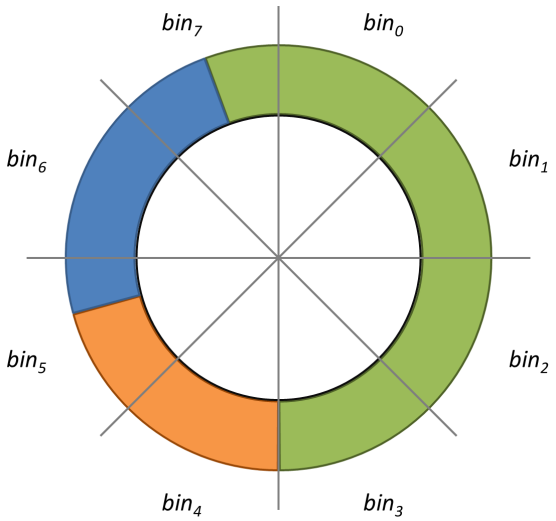



**Figure 2.2:** Illustration for the computation of the context feature. Consider the white object in the middle as the object $o_i \in O$ for which the context feature shall be computed and the blue, green and orange its directly neighboring objects (lets denote these objects $o_{blue}$, $o_{green}$ and $o_{orange}$ respectively). The gray lines indicate the division into eight bins from the center of gravity of the main object.

feature (e. g. *stdDev* or *size*). The definition for individual values of the three vectors is as follows:

$$o_i.contextPattern(inputFeature).min[b] = \min_{o_j \in O | \omega_{i,b}(o_j) > 0} (o_j.inputFeature) \tag{2.4}$$

$$o_i.contextPattern(inputFeature).max[b] = \max_{o_j \in O | \omega_{i,b}(o_j) > 0} (o_j.inputFeature) \tag{2.5}$$

$$o_i.contextPattern(inputFeature).avg[b] = \sum_{o_j \in O | \omega_{i,b}(o_j) > 0} \omega_{i,b}(o_j) \cdot o_j.inputFeature \tag{2.6}$$

Additional to the in-plane 2D context feature we also provide context features derived from the top and bottom neighboring slices. It is still required that all objects are defined in 2D (in one xy-plane), even though we practically operate on a 3D image.

For the $contextPattern_{Top}(inputFeature)$ and $contextPattern_{Bottom}(inputFeature)$ we consider all pixel positions overlapping with our object $o_i$ on the top and bottom neighboring slides, respectively. For the calculation of the angle to determine the binning the z coordinates are simply ignored. The computation of the weights and actual feature values is then exactly the same as described before.

## 2.4 Application of Object-based image Analysis in Biomedical Imaging

The typical tasks that would come to mind first, if thinking about applications for OBIA, are segmentation and automatic detection of specific target structures. Of those there are plenty in the field of biomedical image analysis and that is what the OBIA concept aims at.

However, there are also some much more mundane tasks for which OBIA can be useful. For example the data representation of OBIA makes it very easy and powerful for handling large numbers of image regions and their properties. So even if we do not require to further identify or merge such regions, we can use OBIA to manage them, query their properties, or calculate statistics on them.

Thus depending on the overall task OBIA can be the main driving method supported by other image processing methods or it can itself play a smaller supporting role.

As we can see, the field of possible applications for OBIA in biomedical image analysis is large. Thus in this thesis three tasks are presented that allow to explore the different approaches of OBIA in a variety of imaging domains.

1. **Spine detection in CT:** Computed tomography is one of the most common imaging technologies in radiology. It allows to capture anatomical structures in high detail which are represented by a standardized scale describing the radiodensity. The spine and especially the vertebral bodies of the spine exhibit in itself a very clear contextual concept. A simplified model of it can be described as a flexible column of discs. Those properties make it a good candidate for exploring an explicit OBIA approach.

2. **Ultrasound-based automatic pregnancy detection in pigs:** Ultrasound images are generally harder to interpret than for example CT or MRI images. In the context of pregnancy detection for pigs this is especially true since the veterinarian ultrasound probes are usually much cheaper and thus produce a significantly lower image quality than the ones used in human medicine. However, while humans with the appropriate training can identify the embryonic vesicles on the ultrasound images with very high certainty, the actual properties that discriminate those vesicles from similar structures are hard to describe explicitly. Therefore this task is a candidate for an implicit OBIA approach. Furthermore, the overall goal was to develop a completely automatic system that also acquires the images automatically. For that the correct position on the animal to attach the ultrasound probe to has to be detected. This sub-task also gives the opportunity to apply OBIA to video image analysis.

3. **Histological vessel reconstruction on murine liver samples:** A new technology called “whole slide imaging” allows scanning and digitizing histological samples at microscopic resolutions. This opens completely new opportunities and challenges

for digital image processing and analysis, not only because the size of one two-dimensional image is in the range of several giga-pixels. Due to the high resolution it allows insights into anatomical structures which cannot be matched by any other imaging technology available. In this project the goal is to reconstruct vessels of murine liver samples in 3D. For this first the 2D image slices need to be registered to reconstruct a 3D volume and then the vessels have to be segmented. Using shape features and contextual information to segment the vessel structures suggests itself. Hence using OBIA for this would be a valid approach. Furthermore, considering the large image sizes, using an OBIA-based multi-scale approach would also help to significantly reduce the computational effort.

Overall the above mentioned topics cover a good range of diverse image modalities, they have different goals and require OBIA approaches on different levels. Furthermore they are not just theoretical examples but practical problems, with some even aimed to directly reach commercialization. Thus we believe that they are appropriate to explore and demonstrate the applicability of OBIA in biomedical image analysis.

However, we mentioned before that OBIA is supposed to complement classical image processing approaches and not replace them. Thus the solutions presented in this thesis are not exclusively based on OBIA, but also contain partly novel approaches to solving parts of the given tasks without OBIA. We believe it is worth always discussing the complete approach in this thesis and not exclusively focusing on OBIA, since on the one hand it shows the whole research effort involved and on the other hand fully demonstrates the integration of OBIA into a complete processing pipeline. The following chapters 3, 4, and 5 describe and evaluate in detail our approaches and solutions on the three above mentioned topics.

# Chapter 3

# Spine and Vertebrae Detection in CT

**Acknowledgments** I would like to thank my students (at the time) Teodora Chitiboi and Till Hülnhagen for helping with the implementation of the inital version of the spine and vertebrae detection, which became our first significant publication [137] on this topic. Furthermore I thank Teodora Chitiboi for the inital data analysis and valuable inputs.

**Publications** A preliminary approach to spine detection was published in the *Proceedings of the III ECCOMAS Thematic Conference on Computational Vision and Medical Image Processing: VipIMAGE 2011* [136]. An extended version of the approach was published in the *International Journal for Numerical Methods in Biomedical Engineering* [137]  The latter was the basis for this chapter, but the approach has been significantly altered and extended compared to the paper. The only steps which stayed basically the same as in the paper are the *Pre-Processing* and the *Pre-Segmentation and Seed Detection*. The *Iterative Spine Reconstruction* follows the same ideas, but has been altered and extended. *Classification Refinement* and *Vertebrae Detection* have been redeveloped. Due to these changes all evaluations were redone for this thesis. Furthermore the *Alternative Methods* are a completely new development which I performed for this thesis.

Automatic detection of the spine and vertebrae would be useful in several medical contexts, such as radiation therapy planning or simply to support radiological reporting by automatically labeling single vertebrae correctly. The individual vertebrae are an often used landmark for indicating the localization of other structures in medical reports. Furthermore, an automatic spine detection could be a prerequisite for further computer aided detection processes like the detection of bone metastases, or the assessment of other pathologies such as fractures. Also vertebrae landmarks could be used to initiate automatic detections of other anatomic structures in the correct locations.

The problem of spine and vertebrae detection is a particularly interesting topic in regard to object-based image analysis because the spine shows some rather clear contextual concepts which as a human we can grasp easily. It is therefore a good candidate for exploring the possibilities of an explicit object-based image analysis approach. Hence one reason we chose to approach this particular task is as a show case to try object-based image analysis techniques on a real-world medical imaging problem.

It did not seem realistic to directly attempt a full 3D reconstruction with segmentation into individual vertebrae and intervertebral discs in any kind of vield of view. Therefore in this work we focus on detecting the spine on images that show the whole lumbar part and at least some of the cervical part. Limiting ourselves to the whole spine (including all cervical vertebrae) would make the search for cases for development even harder, because often the head section is not scanned, to spare it from radiation.

Furthermore we limit our detection to a 2D detection on the sagittal slices around the centerline of the spine. The detection will focus on the vertebral bodies. Hence every reference to "spine" in this chapter means the column of vertebral bodies, not including the extensions of the vertebral bodies. The same holds if we refer to "vertebra".

## 3.1 Related Work

Several works have been published regarding detection and segmentation techniques for the spine. The majority of the methods were developed for magnetic resonance (MR) imaging while considerably fewer publications refer to computed tomography (CT) as imaging modality. The proposed methods differ in their specific goals and prerequisites. In the following we will give an overview of this related work.

Huang et al. [62, 61] presented a method that uses AdaBoost and Bayesian weak classifiers, performing on the Haar wavelet domain, for detecting vertebrae in MR images. This stage is followed by a RANSAC [37] estimation of the spinal curve to eliminate outliers and an iterative normalized-cut for segmentation. They report an average detection rate of 90%. Of their evaluation cases only 2 contained the whole spine.

Another work on MR images was presented by Peng et al. [115]. Their first step is to find the intervertebral discs on sagittal slices based on a synthetic 2D model of an intervertebral disc. The result is used to fit a polynomial function and investigate the intensity profile along this function. The final segmentation is done by detection of edge

points and gap filling on an edge image. They report an average detection rate of 94%, but only tested on 5 images.

Carballido-Gamio et al. [18] employ normalized cuts in 3D to segment vertebral bodies in MR images. Their method requires a manual selection of the slice showing the best view of the spinal canal as well as a manual selection of the vertebral bodies for which to retrieve the NCut segmentation. They tested their method on 6 subjects and conclude that the accuracy of their approach has still to be improved.

In their work Tang and Pauli [160] aim to extract the spine curve in MR images. They use a gradient-based method to find possible candidates for vertebral discs. Those candidates are then evaluated by graph-search and an outlier filter based on Active Shape Models. Subsequently an atlas-based registration finds the vertebrae positions from which the spine curve is derived. They report promising results for the spine curve estimation but their method does not provide a real segmentation.

Corso et al. [26] presented a method for the labeling of intervertebral discs in lumbar MR images. They propose a two-level probabilistic model that combines pixel-level information to describe appearance and object-level information such as spatial relationships between discs. They report a detection accuracy of 96.2% but their evaluation was limited to lumbar cases and the method does not provide a segmentation.

Shi et al. [148] presented a method for detecting and segmenting intervertebral discs in MR images. First, the user has to manually select the "best" slice from the dataset. Then the spinal cord is detected using a statistical model and a Hough-Transform. Again a manual step is required to label the first disc. Subsequently the following discs are tracked along the spinal cord.

Schmidt et al. [132] published a probabilistic graphical approach to detect and identify intervertebral discs on MR images without segmenting them. Their method employs a randomized tree classifier that works on 15x15x15 sub-volumes as local feature vectors and predicts possible locations for them. Afterwards the statistical model is used to find the most likely configuration of these parts. Their method does not yield a segmentation of the spine or vertebrae but only a labeling of the intervertebral discs. Furthermore, since the model assumes 24 vertebrae their method fails on images with incomplete spines or fractures.

Another approach for automatically segmenting vertebral bodies and intervertebral discs in MR images of the lumbar and lower thoracic spine was proposed by Neubert et al. [102]. Their approach employs statistical shape models which are represented as 3D meshes with gray level profiles along the mesh point normals. Based on the gray level profiles, an initial guess for the location of a vertebral body or intervertebral disc is refined by iteratively matching possible displacements against the training set. They report a mean Dice score of 0.85 for the vertebral body segmentation and a mean Dice score of 0.78 for the intervertebral disc segmentation.

Moura et al. [100, 101, 99] presented consecutive works on the 3D reconstruction of the spine on biplanar X-ray images. The user is required to set a few control points to which a

deformable articulated model is fitted. In their most recent work [99] they evaluate their method on images of 30 patients with moderate to severe scoliosis. They report root mean square (RMS) reconstruction errors on the vertebrae endplates of 2.0mm and 2.1mm for moderate and severe scoliotic cases respectively. For the pedicles they report RMS errors of 3.5mm and 4.0mm. Related to this work are preceding publications by Moura et al. in which they discuss a method to improve the calibration of biplanar radiography [97, 98].

A method for CT images, though aimed only at detecting lumbar vertebrae was presented by Herring and Dawant [55]. They apply a thresholding on the intensity values to gain a rough pre-segmentation of the bone structures. Via the marching cubes method they reconstruct a 3D mesh which is registered with different vertebra mesh-models. The presented method requires manual input in form of landmarks and the iso threshold for the marching cubes. Furthermore, the evaluation was done on only 2 datasets.

Yao et al. [176] present a method to extract and partition the spinal column in CT images. They also first employ a simple thresholding to obtain a rough initial segmentation. Then they over-segment the 2D axial slices with an adaption of the watershed transform and apply a graph-based search to find the segments that belong to the spinal canal. A vertebra model is fitted to the image to segment the vertebral region and an analysis of the intensity profile along the spinal cord is employed to partition the spine. They report a successful partitioning of the spine in 97% of the cases but major leakages of the segmentation into the sacrum and pelvis in 20% of the cases plus general minor leakages.

An approach for very precise segmentation of vertebrae is described by Mastmeyer et al. [85]. They propose a multi-step procedure that first coarsely separates vertebrae, then uses deformable models for segmentation and subsequently applies a segmentation refinement. Their results show good segmentation results but the method requires manually set landmarks for each vertebra.

Klinder et al. [64] describe a comprehensive framework that aims to segment and identify vertebrae in CT images. Their approach combines a multitude of methods and makes considerable use of a-priori knowledge. An iterative matching algorithm first finds the spinal cord. Then a curved planar reformation based on the spine curve is applied to the image, to be followed by vertebrae detection using generalized Hough transformed models of vertebrae. Individual vertebrae are identified by registering appearance models to the results of the previous step and the final segmentation is based on an extended adaption of shape models. They report a detection rate of 92%, though it has to be noted that their success criterion was based on detecting all vertebrae completely shown in the image. Their results seem most promising but they come at the cost of a very high computation time.

An interactive method for segmenting the spine in 3D was proposed by Ghebreab and Smeulders [42]. They rely on a deformable spine model that is based on a strings and necklaces approach, which basically combines energy minimization, landmark-based segmentation and statistical deformable models. However, their approach was only demonstrated

visually (on screen-shots of 3D renderings) on one example of a lumbar spine scan.

In summary, we see that a multitude of different approaches exists but the community is far from a conclusive solution, especially concerning spine and vertebrae detection in CT images. The presented approaches show different drawbacks: Some only concentrate on specific parts of the spine, require manual user input, or are computationally expensive. Furthermore, some methods used non-pathological data for their evaluation, while others also included cases with e.g. fractures or lesions.

## 3.2 Methods

The spine and vertebrae detection algorithm is implemented in several major steps, which we will describe individually in the following subsections. See also the flowchart in Fig. 3.1 for an illustration of the algorithm steps.

In section 3.2.1 we will first give a brief description of the types of images our method applies to and the data we used for developing the method. The first step in the detection algorithm is a pre-processing of the input image which we describe in section 3.2.2.

The basic strategy for our detection algorithm is to first find a few vertebral bodies in the image to serve as seed objects. The focus in this step is to gain a very high precision on the seed detection, while accepting low detection rates. Based on those seed objects we can then employ contextual information and prior knowledge about the shape of the spine to reconstruct the spine step by step, which is followed by a detection of individual vertebra positions.

To perform this analysis in an object-based manner a previous over-segmentation of the image is required. We combined finding the seed objects and determining an appropriate over-segmentation in one iterative algorithm which is explained in section 3.2.3.

Following the pre-segmentation and seed detection we reduce the number of slices further to focus the subsequent steps on the central slices which show the whole or at least a larger part of the spine. This slice selection process is described in section 3.2.4.

The full reconstruction of the spine is performed in an iterative process which works on a per-slice basis as described in section 3.2.5. Subsequently a classification refinement step (see section 3.2.6) incorporates 2D and 3D overlap information to improve the spine detection.

After the overall spine has been reconstructed a dedicated classification step detects the positions of individual vertebrae within the spine as explained in section 3.2.7

A visual impression of the main steps of the algorithm on an exemplary image can be found in figure 3.8.

### 3.2.1 Data

Since our method operates fully automatically, the only required input is an upper body CT scan. However, the detection method is designed to work only on sagittally oriented

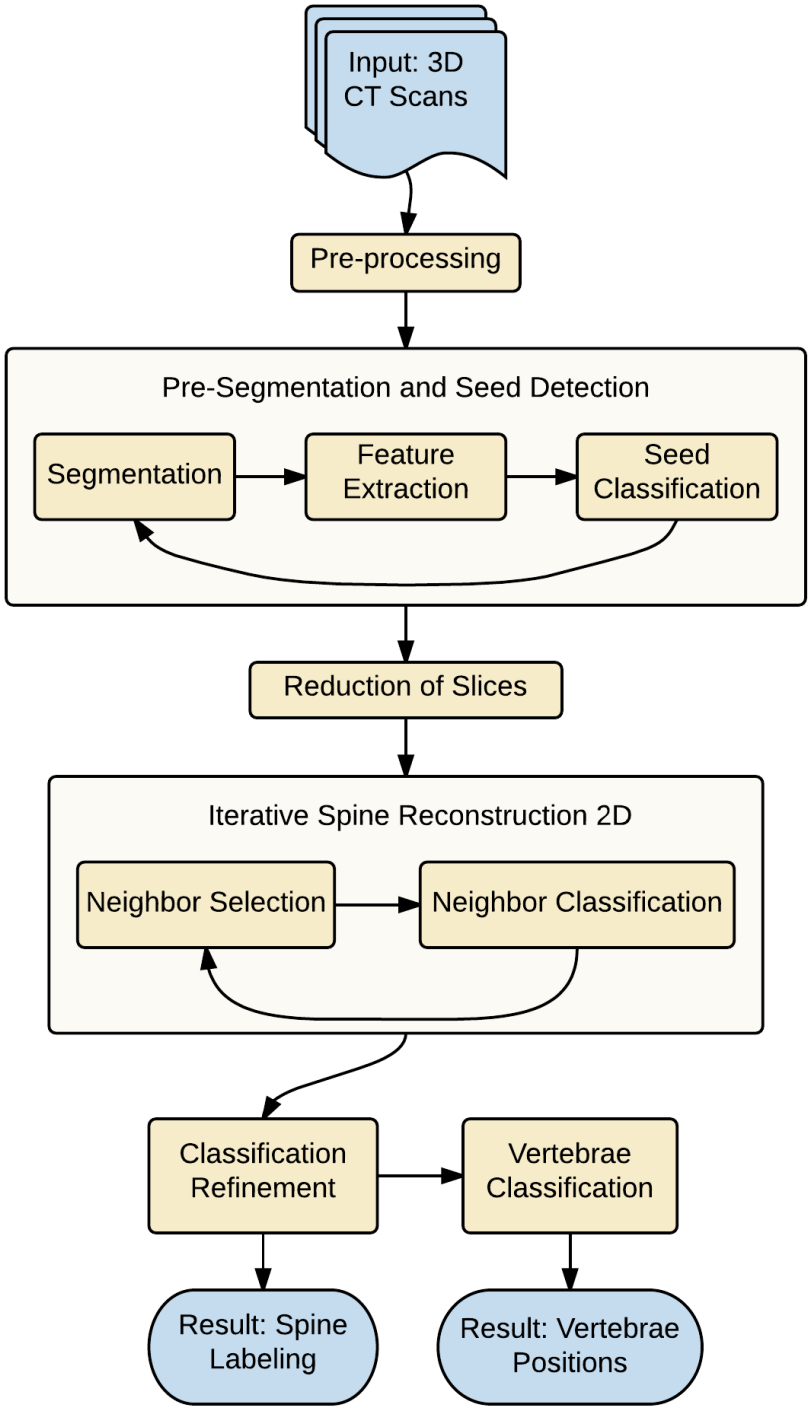



**Figure 3.1:** Overview of the spine detection algorithm.

images. Therefore images in an axial or coronal reformation are automatically converted to a sagittal reformation. Furthermore all images are re-sampled to a voxel size of 2.0mm in z-dimension (in respect to the sagittal reformation). The voxel sizes for the x- and y-dimension range between 0.6mm–0.8mm.

For the development and evaluation of our method we had 20 CT scans available. From those we selected 7 cases which we used for data analysis during the development to derive classification strategies and parameters. All other cases remained to be exclusively used for the evaluation.

The images include pathologically changed spines as well as healthy ones. Unfortunately not all available datasets show the whole spine. On some of them the field of view was selected in such a way that the upper cervical vertebrae were not captured. In those cases we added 150px of empty space on the top to get roughly the same image format on all images.

For each dataset we created a reference labeling indicating which areas are part of the spine and which are to be considered background. Additionally each vertebra from the lowest lumbar (L5) to the second cervical (C2) was labeled individually. Areas of the spine above C2 or below L5 were labeled as "neutral". Since our method currently is not yet designed to detect the atlas, coccyx, and sacrum areas, we didn't want to count them as false negatives in case they were not detected as part of the spine, but also not as false positives in case they were nevertheless captured.

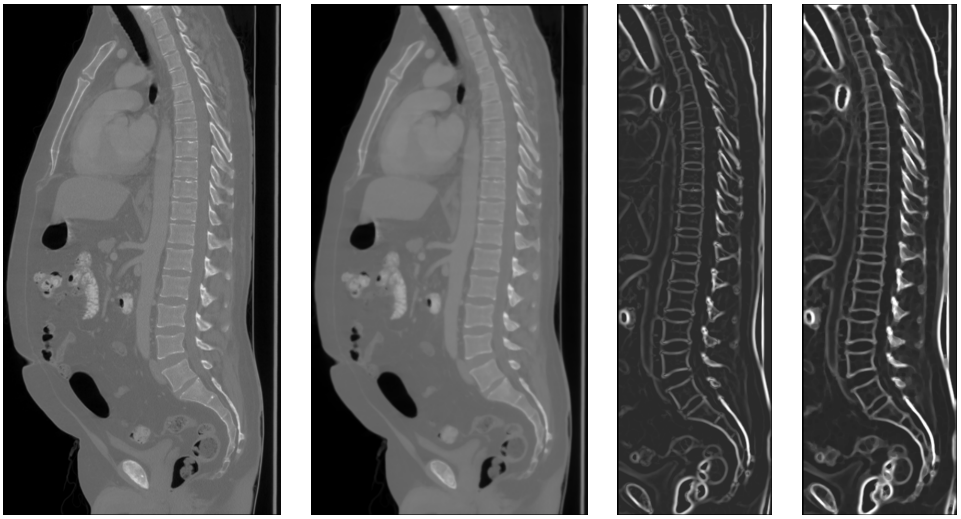
### 3.2.2 Pre-Processing

The pre-processing step is intended to deal with noise in the images and to create a gradient magnitude image for the subsequent pre-segmentation algorithm.

We limit all our further calculations to the sagittal slices in a range of $42mm$ around the central slice. In our cases this was sufficient to include the main central slices showing the spine. If the images are acquired in such a way that the spine is not in the center, the range can be increased at the cost of increasing computation time.

On the sub-volume formed by those remaining slices we apply a 5x5x5 Gaussian filter and a 3x3x3 Median filter to remove noise. Since the vertebrae boundaries have the property of being brighter than their surroundings we can enhance their borders by applying an edge/line-detection filter. Here we employ the method presented by Sato et al. [131] which calculates the filter output as a function of the Hessian matrix. To close little gaps, this is followed by a morphological closing operation with a 5x5x5 kernel. This results in an image that delineates the edges of bone structures well enough to subsequently perform a watershed transformation to create the over-segmentation of the image. See also Fig. 3.2 for examples of the filtering steps.

During the pre-processing step we can also already disregard all regions that do not lie close to bone structures. Not considering those regions in the further processing saves computation time and simplifies the object-based detection algorithm.



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

**Figure 3.2:** Pre-processing steps of the input images. (a) Original image. (b) Gaussian and Median filtered image. (c) Image after applying filter of Sato et al. [131]. (d) Image after morphological closing.

To get a mask that covers only bone structures and their vicinity first a simple thresholding selects all voxels with a value above 180 Hounsfield Units (HU). On the resulting mask image a strong dilation is performed, followed by a connected components analysis to keep only the biggest structure. Now again a strong dilation is performed, which results in a coarse mask of the main bone structures and their vicinity. Some gaps might still occur in the mask, thus a connected components analysis is performed on the background voxels and all small enclosed components are included in the mask. The resulting mask is now used to limit the regions for the further processing.

### 3.2.3 Pre-Segmentation and Seed Detection

An over-segmentation of the input image is a prerequisite for the following object-based image analysis, since it defines the initial objects $O$, as well as their neighbor relations $N$ from which the initial object-based image representation $D$ (see 2.1) is derived.

For the over-segmentation we employ a watershed algorithm on a per-slice basis, thus the resulting regions only span in x and y direction but they exist in the 3D space. However, simply applying a watershed transformation on the gradient magnitude image will not lead to satisfying results, because the segmentation would be too fine. What we are looking for is an initial over-segmentation that is fine enough to respect important borders in the image, especially regarding the vertebrae. However, the over-segmentation should also not be too fine-grained, since the resulting objects would not represent any meaningful structures anymore and thus not bear significant feature information. Therefore we employ

the watershed method introduced by Hahn and Peitgen [48], which provides a parameter called "pre-flooding height" which allows for setting the granularity level of the watershed segments (see Fig. 3.6). However, it is not possible to find a fixed value for the pre-flooding height that results in an appropriate over-segmentation in all cases. Thus we developed a strategy that automatically determines a good pre-flooding height.

The perfect initial over-segmentation would delineate each vertebral body on an image slice as one segment. Of course this is in general not possible, therefore our aim is to reach an over-segmentation that approximates this ideal. Our strategy for this combines the search for a good over-segmentation with finding seed objects which are required to initiate the full reconstruction of the spine. Seed objects have to be vertebral bodies which are already completely segmented as one piece and can be identified with a very high precision. Therefore both tasks complement each other: On the one hand we need a good over-segmentation to be able to find good seed objects and on the other hand, the number of seed objects in an over-segmented image is a good indicator for the quality of the watershed segmentation.

Hence the basic idea of our strategy is to try different pre-flooding heights and run a classifier on the resulting over-segmentations that identifies vertebral bodies. The more vertebral bodies are found, the more appropriate the over-segmentation is considered to be. In the following subsections we will first describe the seed objects classification and then how this classification is used in an iterative manner to create the initial over-segmentation.

#### 3.2.3.1 Seed Object Classification

The goal of the seed object classification is to find vertebral bodies with a very high precision. This is crucial because the seed objects are used as starting points for the subsequent context-based reconstruction of the spine.

To establish the seed classifier we manually set a good pre-flooding level for the watershed algorithm on each of our training datasets, such that as many vertebral bodies as possible are delineated as one segment. Each segment represents an object according to the object-based image representation $D = (O, A)$ (see section 2.1). Now on each slice we manually labeled all well segmented vertebral bodies as positive samples. All objects that were not part of the spine were labeled as negative samples. Objects that were part of the spine but were not well segmented (i.e. under-segmented or over-segmented objects) were not labeled.

Using the tool RapidMiner [89] we performed a feature space analysis on the set of labeled objects, aiming to derive a rule-based classifier with the desired high precision. Out of the pool of features available in our framework we found seven distinctive for vertebral bodies. After inspecting their covariance shown in Table 3.1, we have reduced the number to the five most essential which show little or no correlation. These represent the main characteristics of vertebral bodies that also human observers would use for their identification (on a sagittal per-slice view): The most apparent property we observe is the similarity of a vertebral body to a rectangle, which is represented by $rectFit_1$. Furthermore

**Table 3.1:** Correlation matrix for the set of features.

| | $squareFit$ | $rectFit_1$ | $mean$ | $stdDev$ | $upQuartile$ | $size$ | $eccentricity$ |
|---|---|---|---|---|---|---|---|
| $squareFit$ | 1 | 0.818 | 0.005 | -0.017 | 0.018 | -0.179 | -0.220 |
| $rectFit_1$ | 0.818 | 1 | 0.034 | -0.008 | 0.043 | -0.114 | -0.195 |
| $mean$ | 0.005 | 0.034 | 1 | 0.348 | 0.943 | -0.022 | -0.160 |
| $stdDev$ | -0.017 | -0.008 | 0.348 | 1 | 0.538 | 0.003 | -0.039 |
| $upQuartile$ | 0.018 | 0.043 | 0.943 | 0.538 | 1 | -0.054 | -0.149 |
| $size$ | -0.179 | -0.114 | -0.022 | 0.003 | -0.054 | 1 | -0.115 |
| $eccentricity$ | -0.220 | -0.195 | -0.160 | -0.039 | -0.149 | -0.115 | 1 |

*eccentricity* accounts for the fact that vertebral bodies appear rather compact and not stretched. Another important shape descriptor is the *size* of an object. As features for describing the intensity characteristics of vertebral bodies we used *upQuartile* and *stdDev*.

Figure 3.3 shows the histograms for the features we considered for the classification. We can see that none of the features alone is discriminative enough. $rectFit_1$ actually shows a discriminative tendency but the overlap of positive and negative samples is still strong.

If we look at several features combined, however, we get a different result. Lets first consider the correlation of the shape features: The bubble chart in figure 3.4 depicts the cross-distribution of $rectFit_1$ and *eccentricty* while the *size* of an object is represented by the bubble size. We can clearly see the clustering of positive samples in the upper left corner. Thus setting the thresholds for $rectFit_1 > 0.75$ and $eccentricity < 0.2$ already yields a very good sub-selection. Furthermore we can observe that most positive samples are rather similar in size.

To refine the classification we examine the subset of samples in the upper-left corner. Figure 3.5 shows a deviation plot of this subset considering the features *stdDev*, *upQuartile* and *size*. We can clearly see that the *size* feature is most discriminative withing the remaining objects, while for *stdDev* and *upQuartile* we observe an overlap. However, the standard deviation of the features is very small for the positive samples (see the narrow shaded areas in Figure 3.5) in relation to the negative samples, thus limiting those features accordingly will rid us of many false samples.

Based on these observations we can define a small set of classification rules for seed objects:

$$o.rectFit_1 > 0.75 \tag{3.1}$$

$$o.eccentricity < 0.2 \tag{3.2}$$

$$360\text{mm}^2 < o.size < 1260\text{mm}^2 \tag{3.3}$$

$$o.upQuartile > 125\text{HU} \tag{3.4}$$

$$40 < o.stdDev < 150 \tag{3.5}$$

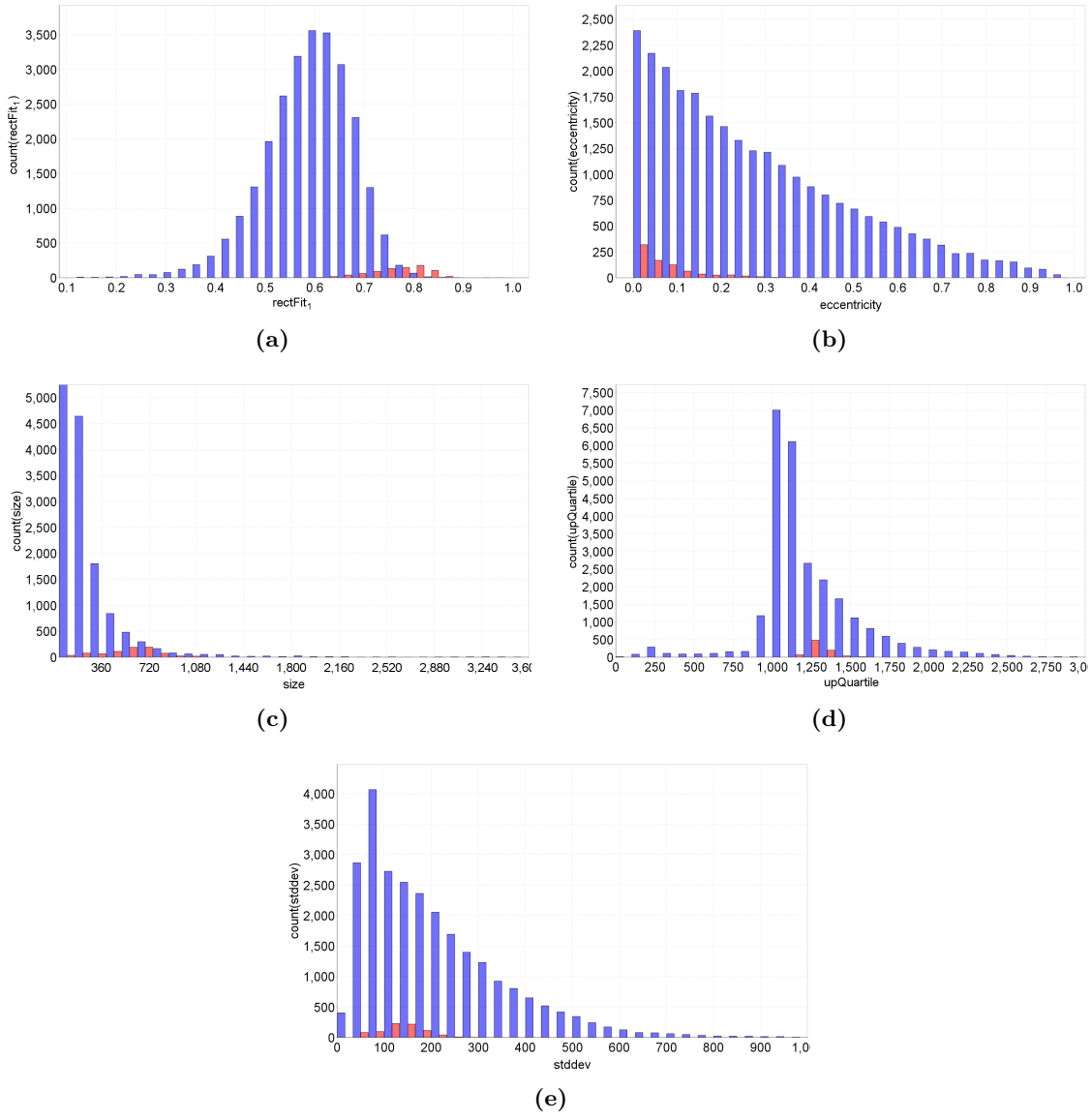



**Figure 3.3:** Histograms of feature distributions for positive (red) and negative (blue) samples: (a) $rectFit_1$. (b) *eccentricity*. (c) *size*. (d) *upQuartile*. (e) *stdDev*.

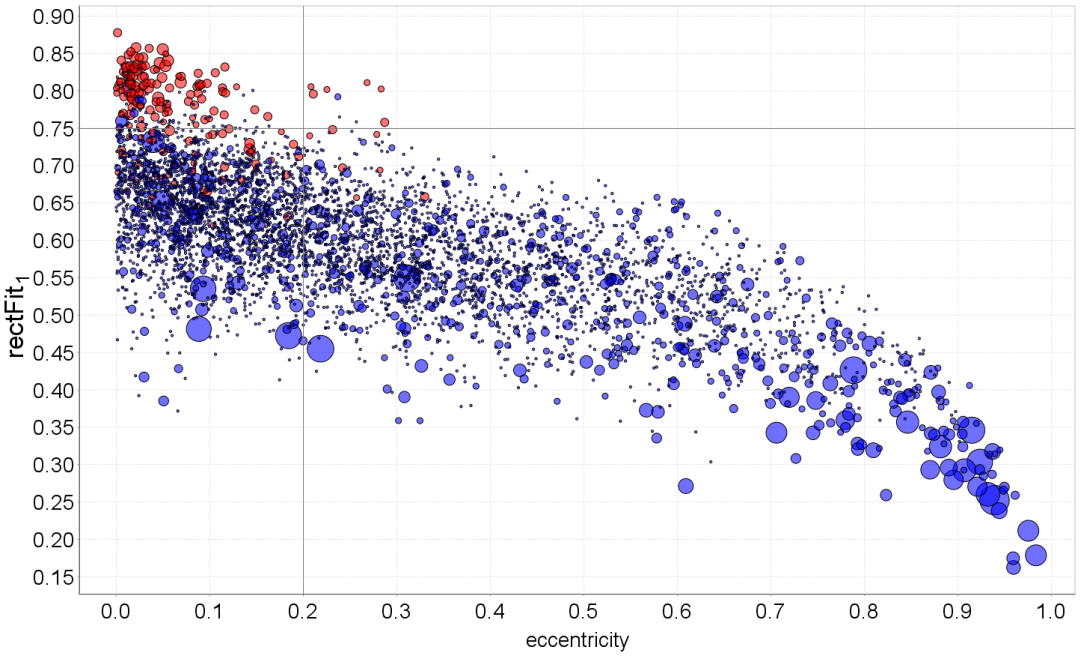



**Figure 3.4:** Bubble chart: cross distribution of *eccentricity* vs. $rectFit_1$; bubble size represents *size* of the object; red = positive samples; blue = negative samples.

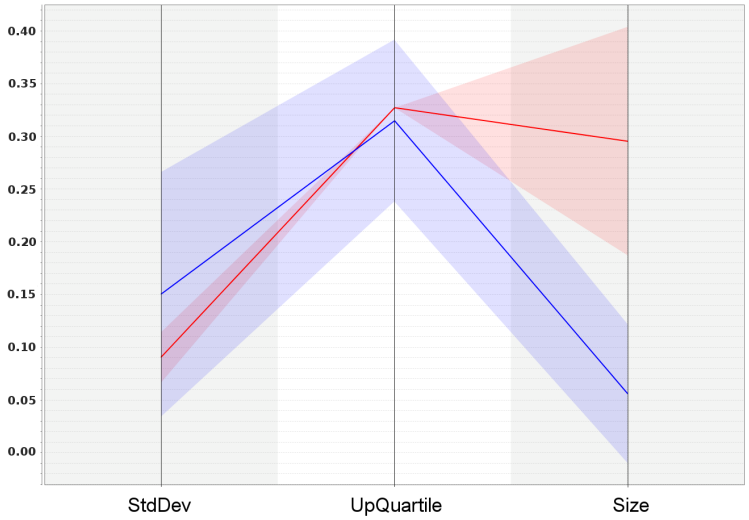
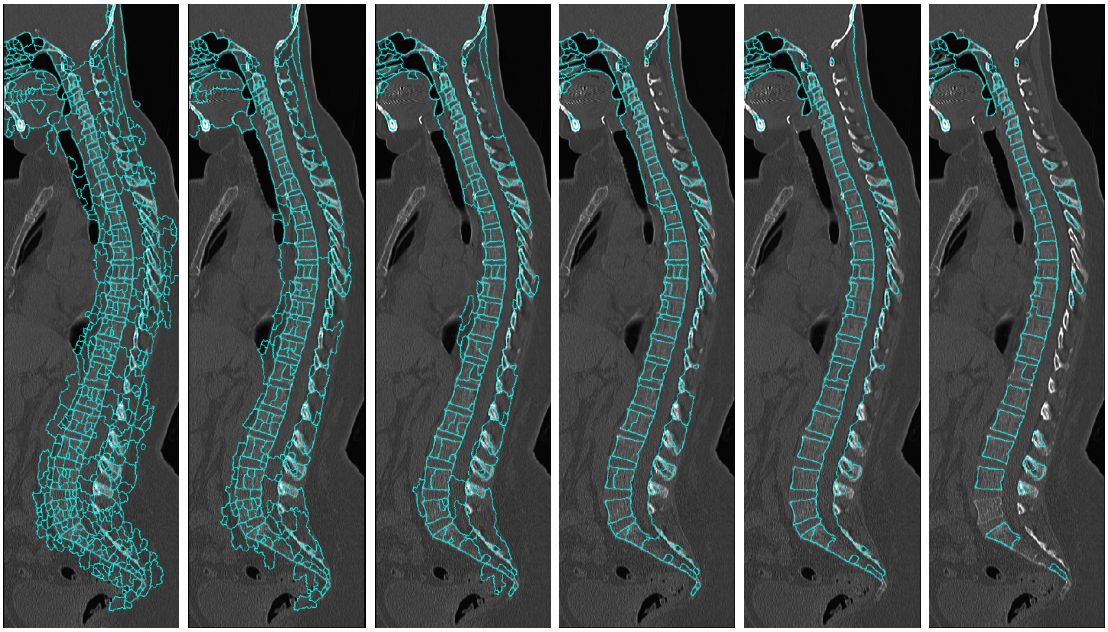



**Figure 3.5:** Normalized deviation plot for *stdDev*, *upQuartile* and *size*: Lines represent the average value and the shaded areas depict the standard deviations; red = positive samples; blue = negative samples.



(a) (b) (c) (d) (e) (f)

**Figure 3.6:** Possible watershed segmentations with different settings for the pre-flooding level: (a) $\rho = 5$, (b) $\rho = 10$, (c) $\rho = 15$, (d) $\rho = 20$, (e) $\rho = 25$, (f) $\rho = 30$.

#### 3.2.3.2 Iterative determination of the pre-segmentation

The iterative search for an appropriate over-segmentation is implemented in a straightforward manner: We start the watershed transformation with an empirical guess of the pre-flooding height $\rho = 16$ which we increase iteratively by a step size of 1. On each segmentation we run the seed classification that was described in section 3.2.3.1. Let $\alpha_\rho$ be the number of positive findings in the segmentation according to $\rho$. The iteration stops if for 4 subsequent steps $\alpha_\rho$ did not increase. The same iterative procedure is also performed decreasing from $\rho = 16$. See figure 3.6 for examples of different segmentation results depending on the setting of $\rho$. Notice how the images on the left are too over-segmented, while the images on the right are too under-segmented. A good pre-segmentation lies somewhere in the middle.

We denote $\rho_{best}$ the pre-flooding height which yielded the segmentation with the highest $\alpha_\rho$. Hence, $\rho_{best}$ gives us an over-segmentation which has the most well segmented individual vertebrae and is thus the best starting point for our analysis. However, it is still possible that some vertebrae are under-segmented. Those objects might pose problems for the detection later and we would like to have the option to split those segments into smaller pieces. Therefore, we define another pre-flooding level $\rho_{base} = max(1, \rho_{best} - 10)$.

The initial object-based image representation is derived from these over-segmentations: Let $O_0$ be the objects based on the segmentation parameterized with $\rho_{base}$ and $N_0$ the

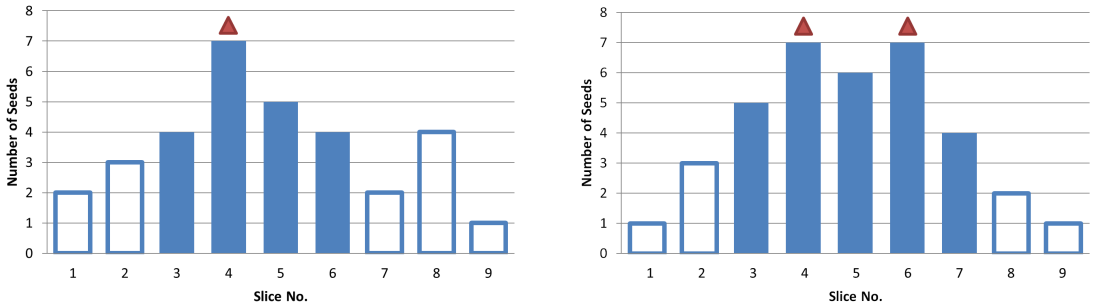



**Figure 3.7:** Exemplary illustration on how the slice reduction is determined. The triangle marks the maximum position(s). The solid bars indicate the interval of slices which will be considered for the further processing.

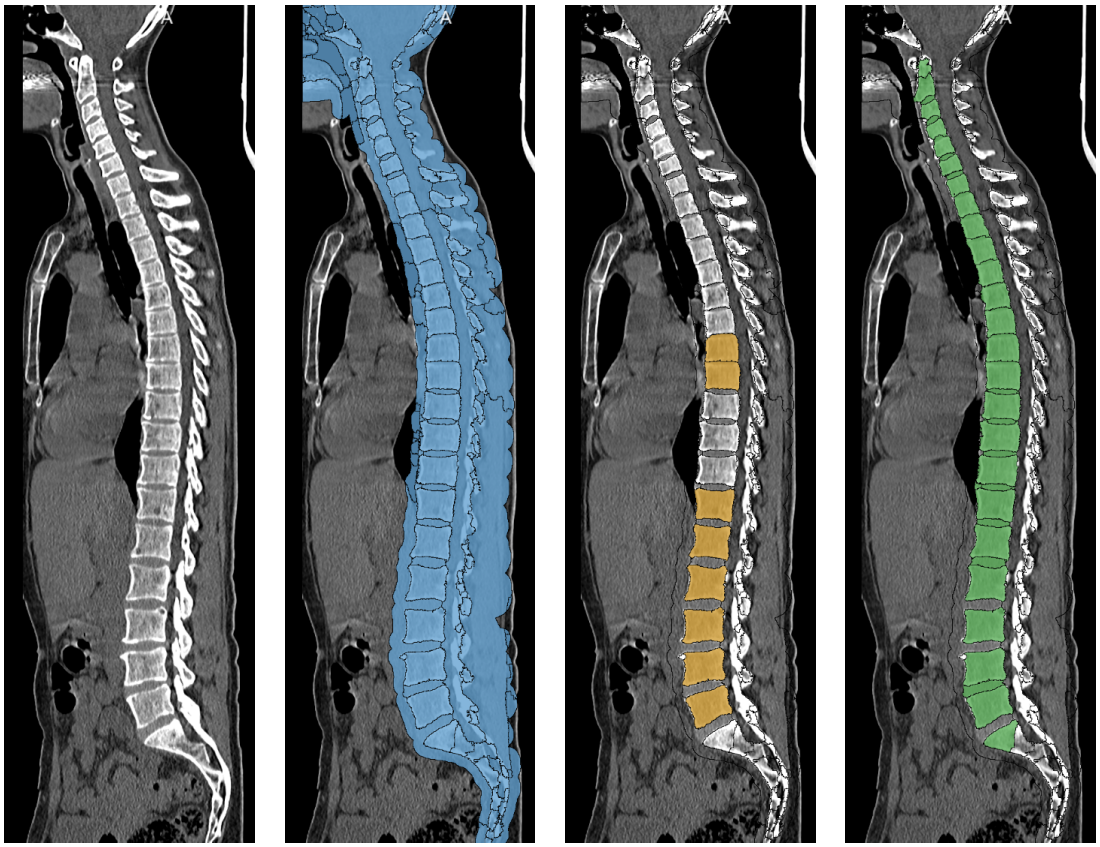
neighbor relations between them. Now for each segment of the $\rho_{best}$-based watershed segmentation we take the set $\{o_i, .., o_j\} \in O_0$ of objects which are covered by this segment and create a new object $o_n$ and according hierarchical relations $\{m_{n,i}, .., m_{n,j}\}$ which are summarized in $O_1$ and $M_1$, respectively. Also new neighbor relations $N_1$ between objects of $O_1$ are derived implicitly. This results in the initial object-based image representation $D = D_{0,1} = D_0 \cup D_1$ with $D_0 = (O_0, N_0)$ and $D_1 = (O_1, N_1 \cup M_1)$.

### 3.2.4 Reduction of Slices

In this step we aim to reduce the amount of slices we are going to analyze. Our goal is to detect the spine in CT slices along the centerline of the spine, in which all or most vertebral bodies are visible. Slices on which only a small part of the spine is visible are not our focus. The automatic selection of these slices from the initial subset of sagittal slices can be achieved by analyzing the distribution of seed objects on those slices.

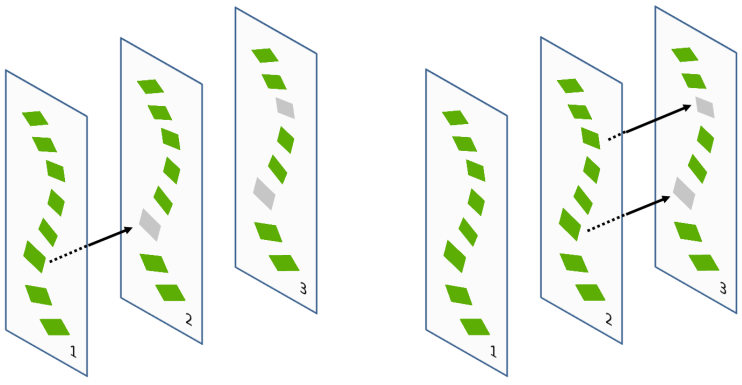
For that we consider the number of detected seed objects per slice. In the optimal case the distribution over the slices would resemble a Gaussian distribution, with a maximum number of seed objects present on the slices around the center of the spine. In practice though, the distribution may be noisy due to poor contrast and noise in the image as well as structural changes to the spine which cause the pre-segmentation to create regions that do not meet the strict seed objects criteria.

We first determine the slice with the maximum number of seed objects. If the maximum is present on more than one slice then the first and last occurrence are considered. Now we span an interval around the maximum position(s) until the first slice with a number of seeds lower than half of the maximum. See Fig. 3.7 for an illustration of the interval definition. For the further processing only the slices that correspond to that interval will be considered.



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

**Figure 3.8:** Overview of the main steps of the algorithm (on an initially exceptionally well segmented slide): (a) Initial image. (b) Best pre-segmentation and initial objects. (c) Seed objects. (d) Final result.




(a) (b)

**Figure 3.9:** Example illustration of the inter-slice propagation of seed objects. (a) One step. (b) Next step.

### 3.2.5 Iterative Spine Reconstruction

The iterative reconstruction of the spine starts with the seed objects that were determined by the seed classifier (see section 3.2.3.1) on the $\rho_{best}$ pre-segmentation. We denote the set of seed objects $S$. At first we propagate the seed objects to all neighboring slices (see Fig. 3.9 for an illustration). Starting from the slice $z = 1$ we check the z-overlap of each object $o_i \in O_1$ with $o_i.centerZ = z$ with all objects $S_{z-1} = \{s_j \in S \mid s_j.centerZ = z - 1\}$. If the z-overlap for an object $o_i$ is greater than 90%, then $o_i$ is added to $S$. After all objects of one slice have been processed we set $z = z + 1$ and repeat the propagation until the last slice of the image is reached. Then the direction is reversed and the same propagation is performed in the opposite direction.

Based on the objects in $S$ we are now searching for further objects in their vicinity which are most probably also part of the spine. The selection process takes into account local model-based assumptions regarding the position of yet undetected objects to previously detected spine objects as well as their individual object features.

The outline of the iterative reconstruction strategy is as follows: Let $S_k$ be the set of unprocessed seed objects and $S_k^+$ the set of already processed seed objects at iteration step $k$. Thus initially $S_0 = S$ and $S_0^+ = \emptyset$. In each iteration $k$ we take one object $s_i \in S_k$ and consider $C_{s_i}$ the set of direct and secondary neighbors as candidates:

$$\begin{aligned} C_{s_i} = \{o_j \in O_1 \mid (n_{i,j} \in N_1 \wedge s_i.centerZ = o_j.centerZ) \\ \vee (n_{i,k}, n_{k,j} \in N_1 \wedge o_k \in O_1 \wedge \\ s_i.centerZ = o_k.centerZ \wedge \\ o_k.centerZ = o_j.centerZ)\} \end{aligned} \tag{3.6}$$

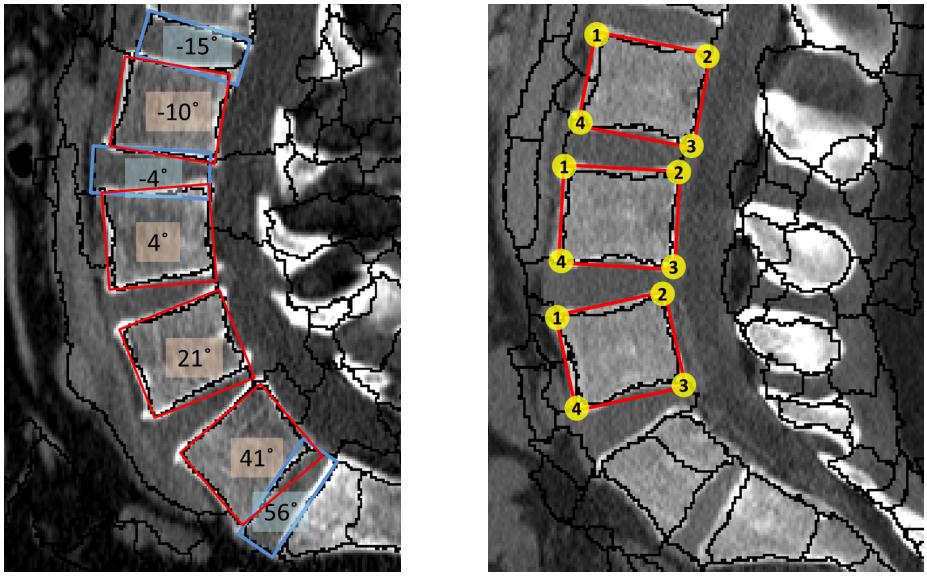
All neighbor candidates are then filtered based on their position relation (details see section 3.2.5.1) and their object features (details see section 3.2.5.2). The objects classified as positive in both steps are added to $S_{k+1}$ to be processed in future steps, while $s_i$ is removed from $S_{k+1}$ and added to the set of already processed objects $S_{k+1}^+$.

#### 3.2.5.1 Neighbor classification using spatial relation information

To decide if a candidate in $C_{s_i}$ is actually a continuation of the spine, a crucial information is its relative location to $s_i$. Here we can employ previous knowledge about the typical shape of a spine: The spine is, simply put, a band of vertebral bodies and inter-vertebral discs with a certain flexibility.

One condition we expect to be fulfilled is therefore that the relative orientation between a candidate and the seed object $s_i$ does not exceed a certain angle threshold. To approximate this we rely on the orientation angle from the minimum bounding rectangle feature. Hence for a $c \in C_{s_i}$ the following must apply:

$$\mid c.angle_{MBR} - s_i.angle_{MBR} \mid < maxAngleDiff \tag{3.7}$$




(a) (b)

**Figure 3.10:** Spatial properties used to distinguish spine objects (a) Minimum bounding rectangles of vertebrae (red) and inter-vertebral spaces (blue) and their orientation angles. (b) Corner position numbering for minimum bounding rectangles.

Fig. 3.10a shows an exemplary illustration of the minimum bounding rectangles and orientations of vertebral bodies and inter-vertebral spaces.

A further spatial constraint is that consecutive spine objects need to have a good alignment, meaning a candidate should be rather close to the corresponding seed and especially in x-direction the displacement must be small. To express this we use the corner positions of the minimum bounding rectangle feature, since the rotating calipers method [164] provides a well-defined numbering of the corner points as illustrated in Fig. 3.10b.

Now first we determine which of the two objects $c$ and $s_i$ is located on top of the other. If $\mid c.corner2y - s_i.corner3y \mid > \mid s_i.corner2y - c.corner3y \mid$ then $o_{top} = c$ and $o_{bottom} = s_i$ otherwise $o_{top} = s_i$ and $o_{bottom} = c$.

Two conditions now restrict the y-distance between the seed and candidate object (note that the image origin is in the top left corner):

$$o_{bottom}.corner2y - o_{top}.corner3y < maxDistY \tag{3.8}$$

$$o_{bottom}.corner2y - o_{top}.corner3y > minDistY \tag{3.9}$$

with

$$maxDistY = yMaxDistFactor \cdot s_i.height_{MBR} \tag{3.10}$$

**Table 3.2:** Parameter settings for neighbor classification using spatial relation information.

| Parameter Name | Value |
|---|---|
| $maxAngleDiff$ | 24.0° |
| $minDistY$ | -10.0mm |
| $yMaxDistFactor$ | 2.0 |
| $maxGapX$ | 13.0mm |

Another condition defines the permitted displacement in x-direction:

$$\mid o_{bottom}.corner2x - o_{top}.corner3x \mid < maxDistX \tag{3.11}$$

with

$$maxDistX = min(s_i.width_{MBR}, maxGapX) \tag{3.12}$$

The motivation for condition 3.8 is to only accept candidates in a certain distance to the seed. We define $maxDistY$ as a multiple of the seed's height, since the height is a good estimate for the scale of vertebrae in the local vicinity. In y-direction we want to consider situations in which the detection process has to "jump" over a misshaped region to the next spine object. A misshaped region could be a strongly pathologic vertebra or just an unfortunate segmentation that diverts the classification rules. We therefore set $yMaxDistFactor = 2.0$.

Condition 3.9 is simply supposed to allow at most a small overlap of the minimum bounding rectangles in y-direction to make sure the objects can really be considered to be on top of each other and not passing by on the side.

At last we want to avoid including objects on the left of the spine or to the right of the spinal cord. Therefore we employ condition 3.11. Since one spine object has to be roughly on top of another, using the width of the seed object's minimum bounding rectangle is a natural restriction for the allowed displacement of opposing corners in x-direction. However, for the large lower vertebrae this could already be too much, which is why the allowed $maxDistX$ is additionally limited by a constant upper bound.

All candidate objects $c \in C_{s_i}$ which pass all those conditions are collected in the subset $C^*_{s_i} \subseteq C_{s_i}$ and considered in the subsequent classification step (see following section 3.2.5.2).

To actually apply the above mentioned conditions, the parameters $maxAngleDiff$, $minDistY$, $yMaxDistFactor$, and $maxGapX$ have to be set to a fixed value. The final setting of those is summarized in table 3.2. How we determined these particular values is described in the evaluation section 3.3.2.

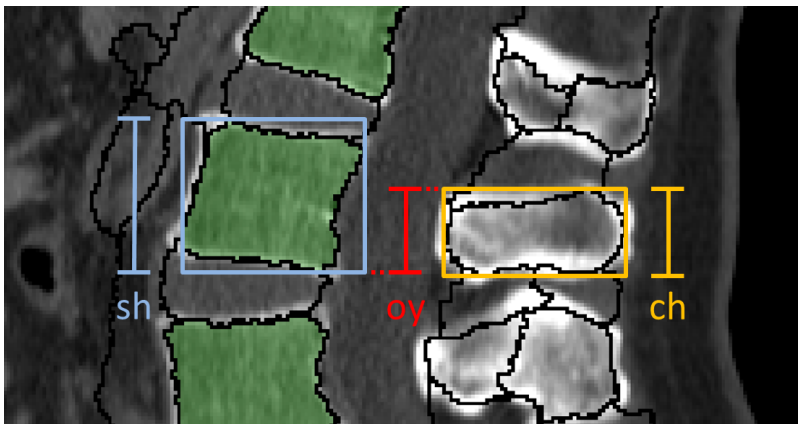



**Figure 3.11:** Calculation of the seed overlap features: *sh* is the height of the seed's bounding box, *ch* is the height of the candidate's bounding box, *oy* is the y-extent overlap of *sh* and *ch*, hence $c.overlapFromSeed = \frac{oy}{sh}$ and $c.overlapToSeed = \frac{oy}{ch}$.

#### 3.2.5.2 Neighbor classification based on object features

Now for each $c \in C^*_{s_i}$ we apply further restrictions based on object features describing typical characteristics that we expect for spine objects. For that we assign $c$ to one of three categories of spine segments:

- $C^L_{s_i}$: Large vertebral bodies, regularly shaped;
- $C^S_{s_i}$: Small vertebral bodies or a fragment (due to over-segmentation) of a larger vertebral body, very compact;
- $C^I_{s_i}$: Intervertebral disks, relatively small, elongated, principal axis orientated rather horizontally.

To perform the categorization, we focused on shape and intensity features since they also come naturally to a human when trying to describe the objects in question.

We also included two features that are rather specific for this classification task. Since we assume that our initial seed objects $S_0$ are definitely part of the spine and should also cover the whole width of the spine in their particular location, we can use this knowledge to break possible leaks in the iterative reconstruction. The idea is to forbid candidates to pass by an initial seed object. To be able to express this, we need a feature that captures this particular spatial relation between seeds and a candidate. Hence we calculate for each $c$ the y-direction overlap of their bounding boxes with the bounding boxes of seed objects in $S_0$. $c.overlapFromSeed$ represents the ratio of the overlap size to the height of the seed, while $c.overlapToSeed$ represents the ratio of the overlap size to the height of $c$. The illustration in Fig. 3.11 clarifies the definitions visually. If more than one seed overlaps with $c$ then always the largest overlap ratio is considered as feature value.

Based on the seed overlap features as well as some shape and intensity features we developed a decision tree for the categorization (or dismissal) of candidate objects, which is depicted in Fig. 3.12. Building such a decision tree manually is of course a rather

**Table 3.3:** Parameter settings for neighbor classification based on object features (see also the decsision tree in Fig. 3.12).

| Parameter Name | Value |
|---|---|
| $minSizeLargeVert$ | 360.0mm |
| $maxSizeLargeVert$ | 1440.0mm |
| $minSizeSmallVert$ | 36.0mm |
| $maxSizeInvertSpace$ | 720.0mm |
| $minRectangularity1$ | 0.7 |
| $minRectangularity2$ | 0.6 |
| $minRectangularity3$ | 0.5 |
| $maxEccentricity1$ | 0.5 |
| $maxEccentricity2$ | 0.3 |
| $maxEccentricity3$ | 0.3 |
| $minSquareness$ | 0.6 |
| $maxSlope$ | 32.0° |
| $avgUpQuartileOffset$ | -100.0 |
| $maxFromSeedOverlap$ | 0.45 |
| $maxToSeedOverlap$ | 1.0 |

heuristic approach and requires an in-depth inspection and understanding of the feature space and feature correlations, given the decision problem at hand. For that we used the data mining tool RapidMiner [89] as well as our own OBIA inspection tools to analyze the 7 training datasets. However, in this particular step of the classification pipeline the complexity of necessary rules and their interdependencies became significantly higher than in the previous steps. The more complex the correlations are the more difficult it becomes to explicitly express one's rather intuitive understanding of the classification problem. Hence the development of this decision tree became an iterative and time consuming task. Therefore we would like to mention at this point that we also tried an alternative approach, which is discussed in section 3.4.

Nevertheless we were able to set up a manual decision tree that produced good results on the training datasets. The rules in the decision tree depend on several thresholds that have to be set manually. The final setting of these is summarized in table 3.3. How we determined these particular values is discussed in the evaluation section 3.3.2.

The rule regarding the upper quartile of a candidate object also requires a global variable $avgUpQuartile$ which is calculated as

$$avgUpQuartile = \frac{\sum_{s_x \in S_0} s_x.upperQuartile}{\mid S_0 \mid} \tag{3.13}$$

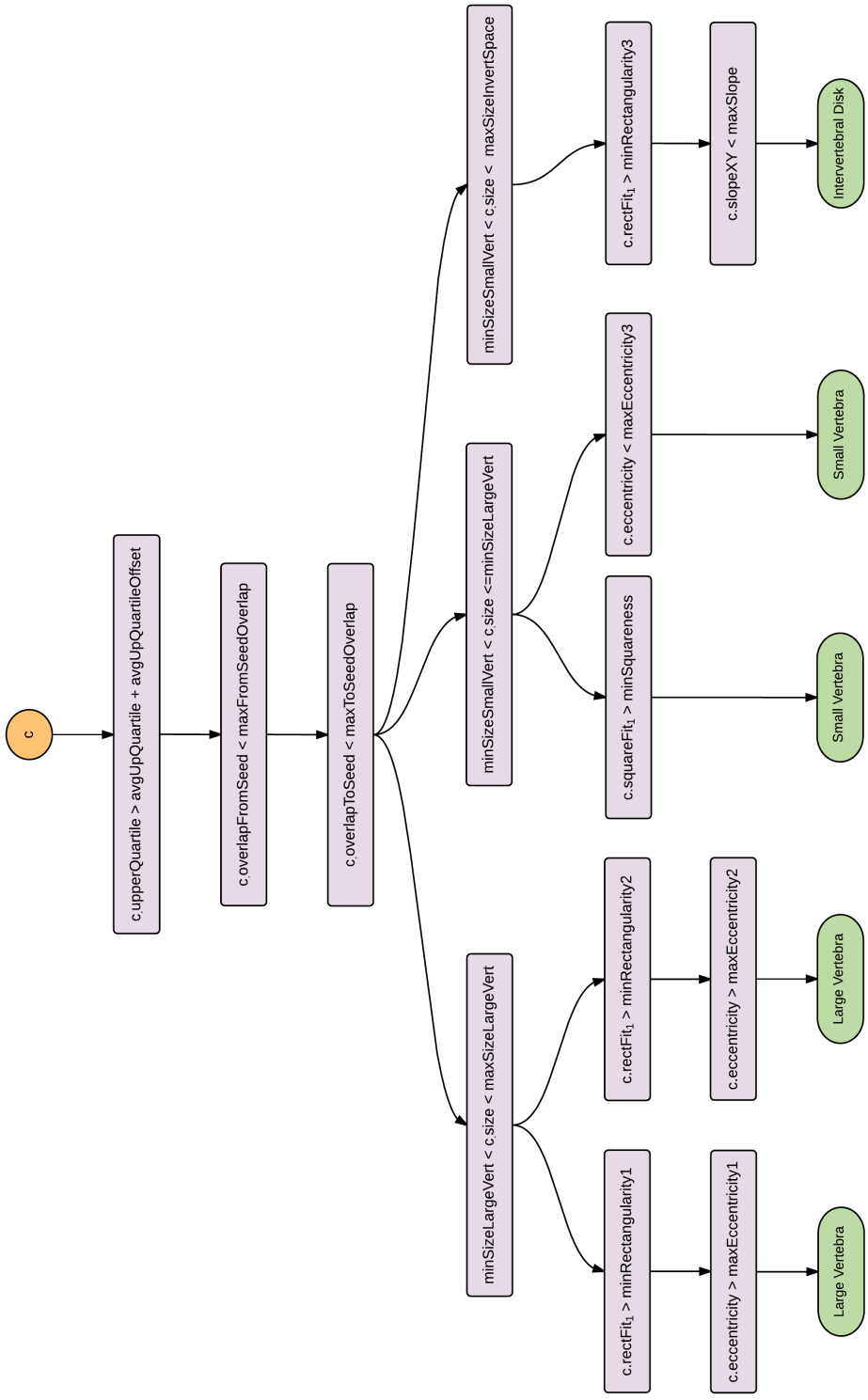



**Figure 3.12:** Decision tree representing the neighbor classification based on object features for a $c \in C^*_{s_i}$. If more than one path is valid, the leftmost is preferred. If no path is valid, $c$ is discarded.

After feeding all candidates from $C^*_{s_i}$ through the decision tree, all objects in the subsets $C^L_{s_i}$, $C^S_{s_i}$, $C^I_{s_i}$ represent spine objects identified in the current iteration, and are thus possible seeds for the next iteration:

$$S^+_{k+1} = S^+_k \cup \{s_i\} \tag{3.14}$$

$$S_{k+1} = (S_k \cup C^L_{s_i} \cup C^S_{s_i} \cup C^I_{s_i}) \setminus \{s_i\} \tag{3.15}$$

The iteration stops when all the seed objects have been processed and thus $S_k = \emptyset$. Then $S^+_k$ contains all spine objects detected so far.

### 3.2.6 Classification Refinement

After finishing the iterative reconstruction, two refinement steps are performed that consider the overlap with neighboring spine objects on the same slice as well as on neighboring slices. The underlying idea is that an object, which is identified as part of the spine, should be well connected with other spine objects.

First we define a feature that measures the in-plane connection of any object $o_i \in O_1$ to spine objects as:

$$o_i.spineConnection = \sum_{n_{i,j} \in N_1^{2D}} n_{i,j}.mcBorder2dRelative \tag{3.16}$$

with

$$N_1^{2D} = \{n_{i,j} \in N_1 \mid o_i \in O_1 \wedge o_j \in S^+_k \wedge o_i.centerZ = o_j.centerZ\} \tag{3.17}$$

Furthermore we define two features which measure the overlap of any object $o_i \in O_1$ to spine objects on the adjacent slices in positive and negative z-direction:

$$o_i.spineOverlap^+ = \frac{\sum_{n_{i,j} \in N_1^+} n_{i,j}.border3dSize}{o_i.size} \tag{3.18}$$

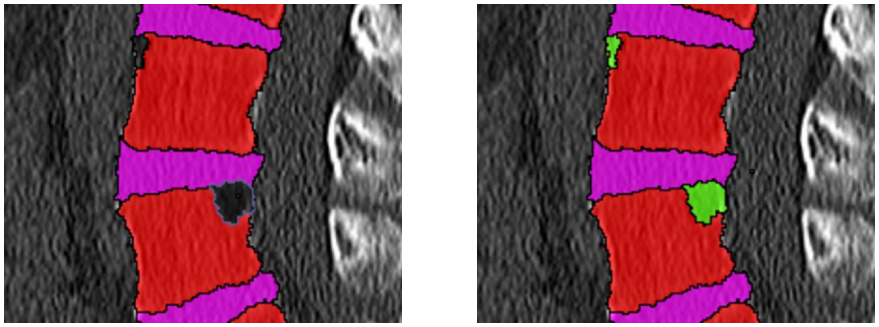
$$o_i.spineOverlap^- = \frac{\sum_{n_{i,j} \in N_1^-} n_{i,j}.border3dSize}{o_i.size} \tag{3.19}$$

with

$$N_1^+ = \{n_{i,j} \in N_1 \mid o_i \in O_1 \wedge o_j \in S^+_k \wedge o_i.centerZ + 1 = o_j.centerZ\} \tag{3.20}$$

$$N_1^- = \{n_{i,j} \in N_1 \mid o_i \in O_1 \wedge o_j \in S^+_k \wedge o_i.centerZ - 1 = o_j.centerZ\} \tag{3.21}$$

The first refinement step is to remove all objects from the set of spine objects which have little overlap to other spine objects on adjacent slices. Since we performed the



(a) (b)

**Figure 3.13:** Example of small missing pieces added by overlap testing on objects of $O_1$ in the second classification refinement step. (a) Small pieces missing after iterative spine reconstruction (section 3.2.5). (b) Inclusion by testing the overlap and connection to neighboring spine objects.

iterative reconstruction strictly on a per slice basis, this is a good sanity check to find possible leaks that happened only in one slice. We denote the set of objects to be removed from the spine $S^-$ and apply the following condition:

$$\begin{aligned} S^- = \{s_i \in S_k^+ \,|s_i.spineOverlap^+ < removeSpineOverlapThresh \wedge \\ s_i.spineOverlap^- < removeSpineOverlapThresh\} \end{aligned} \tag{3.22}$$

Afterwards the remaining set of spine objects is defined as:

$$S^{+'} = S_k^+ \setminus S^- \tag{3.23}$$

The second refinement step is supposed to include objects which were not detected as part of the spine yet, but have a strong connection to neighboring spine objects. This can especially happen to small fragments (see Fig. 3.13) which appear because the over-segmentation, for example, does not guarantee to perfectly capture each vertebra as one piece, especially when pathologies of the spine are involved. The rules of the iterative reconstruction are too strict to capture such fragments—and they have to be, to prevent leaking. The final set of spine objects $S_{final}^+$ is created as follows:

$$\begin{aligned} S^o = \{o_i \in O_1 \,|o_i.spineConnection > addSpineConnectThresh \wedge \\ (o_i.spineOverlap^+ > addSpineOverlapThresh \vee \\ o_i.spineOverlap^- > addSpineOverlapThresh)\} \end{aligned} \tag{3.24}$$

$$S_{final}^+ = S^{+'} \cup S^o \tag{3.25}$$

We again require a few thresholds to be set manually to apply the above mentioned rules. Table 3.4 shows the final settings of these and in section 3.3.2 they are evaluated.

**Table 3.4:** Parameter settings for the classification refinement step.

| Parameter Name | Value |
|---|---|
| $removeSpineOverlapThresh$ | 0.6 |
| $addSpineOverlapThresh$ | 0.0 |
| $addSpineConnectThresh$ | 0.6 |

### 3.2.7 Vertebrae Detection

After the spine has been detected along the vertebral bodies we want to detect the positions of individual vertebrae among all spine objects. The goal is to have exactly one marker indicating a valid position inside a vertebral body for each vertebra. As mentioned before, in this work we only considered the vertebrae from the second cervical to the last lumbar. We excluded the sacrum and the coccyx, since they are already not considered in our spine classification stage, due to their different appearance. The atlas (first cervical) was excluded for the same reason.

Furthermore, our detection should not only determine the positions of vertebral bodies but also attempt to assign the correct name to each of these positions (C2–C7, T1–T12, L1–L5).

Since manually developing the decision tree for classifying candidates into spine and non-spine objects (see section 3.2.5.2) was already a time-consuming effort, we decided to involve a machine learning approach for the vertebrae detection. Considering that now we do not have a two-class problem anymore but need to distinguish between 23 vertebrae and non-vertebrae, a completely manual/heuristic approach did not seem feasible anymore.

Basis for the classification is a Random Forest classifier [14]. It was trained on all training datasets using all spine objects ($S^{+}_{final}$) from the spine detection stage. The classifier was trained for 24 classes: one for each of the 23 vertebra names and one "negative". All objects, which were not labeled as a particular vertebra in the reference, were considered "negative", all others were considered samples for their respective vertebra class. The training samples were not balanced. For our cases it meant that the number of negative samples was significantly higher than the number of each of the vertebrae class samples.

The following features are used to perform the classification:

- $angle_{MBR}$
- $aspectRatio_{MBR}$
- $rectFit_1$
- $squareFit_1$
- $squareFit_2$
- $size_{MBR}$
- $eccentricity$
- $slopeXY$
- $size$

- $relPosX$
- $relPosY$
- $relPosZ$
- $spineClass$

Since we already know that the candidates are supposed to be within the spine we do not rely on intensity features anymore but mainly on shape features. We especially focus on shape features which are based on the minimum bounding rectangle because the similarity to a rectangle is the most apparent attribute of a well defined vertebra on a sagittal slice. However, we added a few features which were not defined before: $relPosX$, $relPosY$ and $relPosZ$ are defined as

$$o_i.relPosX = \frac{o_i.centerX}{ImageSize_x} \tag{3.26}$$

$$o_i.relPosY = \frac{o_i.centerY}{ImageSize_y} \tag{3.27}$$

$$o_i.relPosZ = \frac{o_i.centerZ}{ImageSize_z} \tag{3.28}$$

for an $o_i \in O$ with $ImageSize$ representing the size of the currently processed image.

The feature *spineClass* holds a label for the different types of spine objects which are: initial seed, large vertebra, small vertebra, intervertebral disk, plus an extra label for objects added to the spine during the refinement step.

To detect the vertebrae positions we run the classifier on all objects that are labeled as spine. Since we want exactly one marker per vertebra, we do not directly use the results of the classification but determine the individual vertebrae successively starting with the lowest lumbar (L5) and working our way up. For algorithmic purposes we consider L5 as vertebra no. 1, counting the vertebrae successively upwards until C2 = vertebra no. 23.

The idea of the algorithm that selects the final vertebrae is as follows: Start by selecting the object with the highest class weight for vertebra 1. Then look in the close vicinity on top of this object and select the object with the highest class weight for vertebra 2. Now iteratively continue this process. So to improve the detection quality we do not only rely on the class weights but also include some basic domain knowledge about the expected geometrical order of vertebrae.

The flowchart in Fig. 3.14 describes the algorithm in detail and also explains how we handle cases when no following vertebra can be found which meets the restrictions. To apply the distance constraint, we require a *radius* feature which we simply define as:

$$o_i.radius = \sqrt{\frac{o_i.size}{\pi}} \tag{3.29}$$

The logic we define for the close vicinity is that the center position of a following vertebra must be in a distance range of $1 \cdot o_i.radius$ to $3 \cdot o_i.radius$ on top of the previous vertebra.

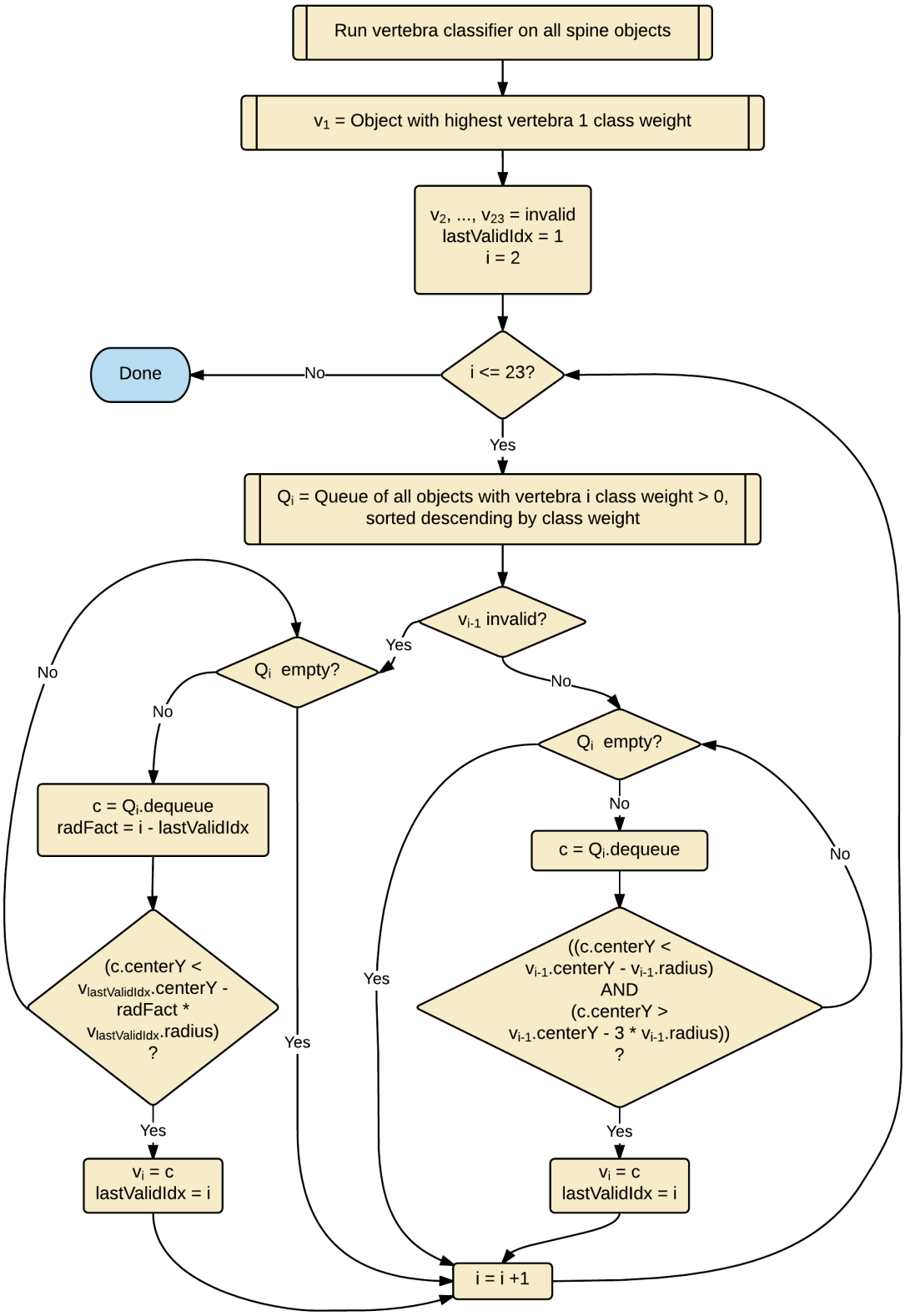



**Figure 3.14:** Flowchart illustrating the vertebrae detection algorithm.

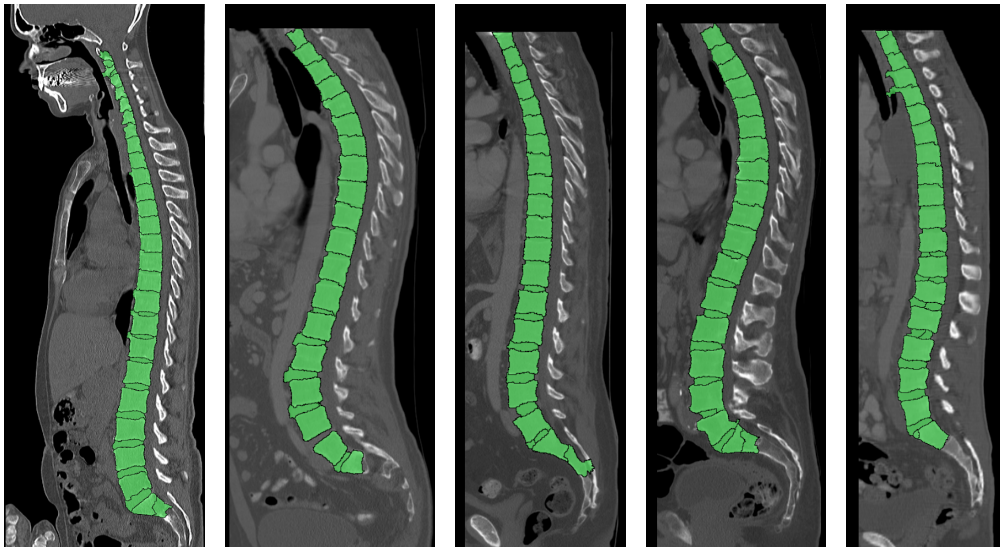


**Figure 3.15:** Examples of detection results.

This is a reasonable constraint because anatomically the size should not change significantly between neighboring vertebrae, they cannot overlap, and the distance between two vertebrae is not larger than the vertebrae themselves.

## 3.3 Evaluation and Results

As mentioned before, for the development and evaluation of our method we had 20 upper body CT scans available, 7 of which were used during the development of the methods, while the remaining 13 were reserved for the evaluation. In the course of this evaluation section we will denote the training cases as 1–7 and the test cases 8–20. We included the training cases in all evaluations to show and discuss the differences of the method's performance on "known" and "unknown" data.

In the evaluation we considered several aspects which are covered in individual subsections: First we describe the evaluation of the seed objects classification in section 3.3.1, since this is the basis for the subsequent iterative spine reconstruction. Because the spine reconstruction has so many parameters, we dedicated section 3.3.2 to explain how we evaluated and optimized those parameters to get to the final parameter setup. The performance of the spine detection based on the final parameter setup is then further evaluated in section 3.3.3. Afterwards section 3.3.4 goes into details on the performance of the detection of individual vertebra positions. Furthermore, in section 3.3.5 we give a short overview on the computing times of our algorithm.

Some visual results of our spine detection method can be seen in figure 3.15. The examples in figure 3.16 show that our method also works well on cases with minor to medium pathological changes, like fractures and lesions.

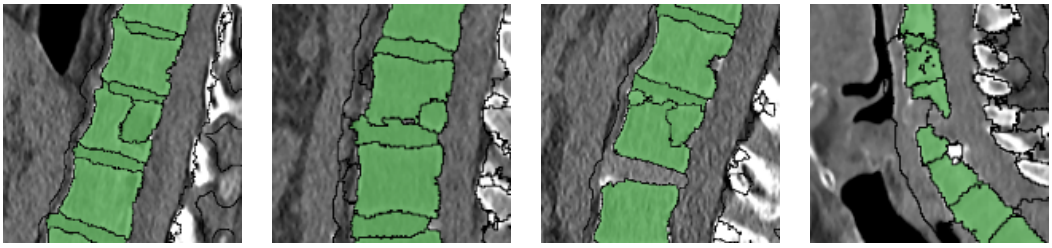


**Figure 3.16:** Difficult cases successfully detected (lesions and a fracture).

### 3.3.1 Evaluation of Seed Object Classification

We evaluated the performance of the seed classification in terms of precision and sensitivity on all 20 datasets, with precision and sensitivity defined as

$$Precision = \frac{TP}{TP + FP} \tag{3.30}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.31}$$

We consider true positives (TP) all objects which are detected as seed object and are labeled with one of the vertebra names in the reference. Accordingly, all objects which are not detected but are labeled as vertebra in the reference are considered false negatives (FN). False positives (FP) are all objects which are not labeled with a vertebra label or are labeled with more than one vertebra label in the reference. Additionally we also counted among the false positives those objects which are at least labeled as spine, even though they are not vertebrae. We denote them SP (spine positive). The reason for this is that an object which is a false positive by definition, is not so troublesome for the following algorithm steps if it is at least within the spine. However, seed objects outside the spine could misguide the iterative reconstruction.

The results in table 3.5 show that the classification reaches a very high precision in all cases, just as it was desired. As expected, the sensitivity is rather low. The results also show that the classification performance on the test data is not much worse than on the training data. Precision-wise the numbers look really good, however in 5 out of 13 test cases one or two false positive detections occur while among the training datasets this only happens for 1 out of 7 cases.

We investigated each false positive detection: For case 6 the over-segmentation caused two vertebrae to be merged together which were wrongly detected as one vertebra. In cases 8 and 18 a piece of the sacrum was detected as vertebra. In case 20 one detection only captured half of the actual vertebra because the over-segmentation was too fine-grained. All those cases are still at least spine positive (SP). However in cases 11 and 15 the false positive detections were actually objects out of the spine.

Another interesting observation is that the sensitivity is significantly better for the test cases. This indicates that the seed detection rules could probably be improved to be a bit

**Table 3.5:** Seed classification results (First 7 cases are the training cases). TP = True Positive; FP = False Positive; SP = Spine Positive (among FPs); FN = False Negative.

| # | TP | FP | SP | FN | Precision | Sensitivity |
|---|---|---|---|---|---|---|
| 1 | 43 | 0 | 0 | 223 | 1 | 0.16 |
| 2 | 141 | 0 | 0 | 148 | 1 | 0.49 |
| 3 | 90 | 0 | 0 | 189 | 1 | 0.32 |
| 4 | 60 | 0 | 0 | 167 | 1 | 0.26 |
| 5 | 117 | 0 | 0 | 137 | 1 | 0.46 |
| 6 | 54 | 1 | 1 | 182 | 0.98 | 0.23 |
| 7 | 57 | 0 | 0 | 214 | 1 | 0.21 |
| Avg.: | | | | | 0.997 | 0.305 |
| 8 | 107 | 1 | 1 | 109 | 0.99 | 0.50 |
| 9 | 69 | 0 | 0 | 86 | 1 | 0.45 |
| 10 | 70 | 0 | 0 | 136 | 1 | 0.34 |
| 11 | 86 | 1 | 0 | 118 | 0.99 | 0.42 |
| 12 | 99 | 0 | 0 | 97 | 1 | 0.50 |
| 13 | 93 | 0 | 0 | 129 | 1 | 0.42 |
| 14 | 98 | 0 | 0 | 103 | 1 | 0.49 |
| 15 | 92 | 2 | 0 | 92 | 0.98 | 0.50 |
| 16 | 91 | 0 | 0 | 98 | 1 | 0.48 |
| 17 | 104 | 0 | 0 | 54 | 1 | 0.66 |
| 18 | 112 | 1 | 1 | 76 | 0.99 | 0.60 |
| 19 | 34 | 0 | 0 | 167 | 1 | 0.17 |
| 20 | 105 | 1 | 1 | 123 | 0.99 | 0.46 |
| Avg.: | | | | | 0.995 | 0.460 |

stricter, which is not surprising with such little training data used. However, it is very likely that to improve on the last missing 0.5 percentage points of the precision we would have to accept a significant loss on sensitivity. Nevertheless the results on the test data indicate that our rules are already very good even though they were developed manually based on a small training set.

### 3.3.2 Evaluation of Spine Detection Parameters

The iterative spine reconstruction and the subsequent refinement depend on several parameters which we have heuristically set by thoroughly exploring the training data. To see if all the chosen settings were good or could be improved we performed an evaluation of all those parameters on the training datasets. We did this by testing a range of rea-

sonable values for each parameter individually while keeping all other parameters at their initial setting. Table 3.6 shows all parameters with their initial values and the tested value ranges.

For the evaluation we used three measurements: precision, sensitivity and vertebrae sensitivity which are defined as

$$Precision = \frac{TP}{TP + FP} \tag{3.32}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.33}$$

$$Vertebrae\ Sensitivity = \frac{VP}{VP + VN} \tag{3.34}$$

with

- $TP$ = all detected objects labeled as spine in the reference;
- $FP$ = all detected objects labeled as background in the reference;
- $FN$ = all undetected objects labeled as spine in the reference;
- $VP$ = all detected objects labeled with a specific vertebra name in the reference;
- $VN$ = all undetected objects labeled with a specific vertebra name in the reference.

We included the special vertebrae sensitivity to give us an idea if the spine segmentation is good enough such that the subsequent vertebrae detection can rely on it. The averaged performance of the spine detection on all training datasets under the initial parameter settings was:

- $Precision = 0.88$;
- $Sensitivity = 0.84$;
- $Vertebrae\ Sensitivity = 0.95$.

To assess if a parameter test value would improve the classification performance, we calculated the difference between these initial classification quality measurements and the classification quality under the test value. The plots in Fig. 3.17, 3.18, 3.19, and 3.20 show the results for all parameters for their respective test value ranges. The only parameters we did not test are $maxSizeLargeVert$ and $minSizeSmallVert$. For those we just looked at the training data and set them to generously cover the size range of all spine objects in the reference.

A look at the plots reveals that most parameters are actually set quite well. Most of them cannot be changed without having a positive and a negative effect on precision and sensitivity at the same time. Changing those parameter values could therefore be used to steer the classification more towards a higher sensitivity with lower precision, or a higher precision with lower sensitivity. The plots also show that a few parameters have very little influence on the result (e.g. $maxEccentricity1$). However, most parameters have a significant influence on the outcome.

We found three parameters that showed potential to influence the classification result positively without significant drawbacks. Lowering $maxGapX$ from 14 to 13 would increase the precision by 1 percentage point without changing the two sensitivity measures.

**Table 3.6:** Initial setting of all spine detection parameters, tested value ranges, and final setting. The separation lines indicate different parameter groups according to the classification step they are used in. First group parameters are from the neighbor classification using spatial relation information (see section 3.2.5.1), second group is the neighbor classification based on object features (see section 3.2.5.2), and the third group represents the classification refinement parameters (from section 3.2.6).

| Parameter Name | Initial Value | Tested Rage | Optimized Value |
|---|---|---|---|
| maxAngleDiff | 24 | 4, 6, ..., 44 | 24 |
| maxGapX | 14 | 4, 5, ..., 24 | 13 |
| yMaxDistFactor | 2.0 | 1.0, 1.1, ..., 3.0 | 2.0 |
| minDistY | -10 | -20, -19, ..., 0 | -10 |
| minSizeLargeVert | 360 | 260, 270, ..., 460 | 360 |
| maxSizeLargeVert | 1440 | – | 1440 |
| minSizeSmallVert | 36 | – | 36 |
| maxSizeInvertSpace | 720 | 320, 360, ..., 1120 | 720 |
| minRectangularity1 | 0.7 | 0.0, 0.05, ..., 1.0 | 0.7 |
| minRectangularity2 | 0.6 | 0.0, 0.05, ..., 1.0 | 0.6 |
| minRectangularity3 | 0.5 | 0.0, 0.05, ..., 1.0 | 0.5 |
| maxEccentricity1 | 0.5 | 0.0, 0.05, ..., 1.0 | 0.5 |
| maxEccentricity2 | 0.3 | 0.0, 0.05, ..., 1.0 | 0.3 |
| maxEccentricity3 | 0.3 | 0.0, 0.05, ..., 1.0 | 0.3 |
| minSquareness | 0.6 | 0.0, 0.05, ..., 1.0 | 0.6 |
| maxSlope | 20 | 10, 12, ..., 50 | 32 |
| avgUpQuartileOffset | -100 | -200, -190, ..., 0 | -100 |
| maxFromSeedOverlap | 0.45 | 0.0, 0.05, ..., 1.0 | 0.45 |
| maxToSeedOverlap | 1.0 | 0.0, 0.05, ..., 1.0 | 1.0 |
| removeSpineOverlapThresh | 0.6 | 0.0, 0.05, ..., 1.0 | 0.6 |
| addSpineOverlapThresh | 0.7 | 0.0, 0.05, ..., 1.0 | 0.0 |
| addSpineConnectThresh | 0.6 | 0.0, 0.05, ..., 1.0 | 0.6 |

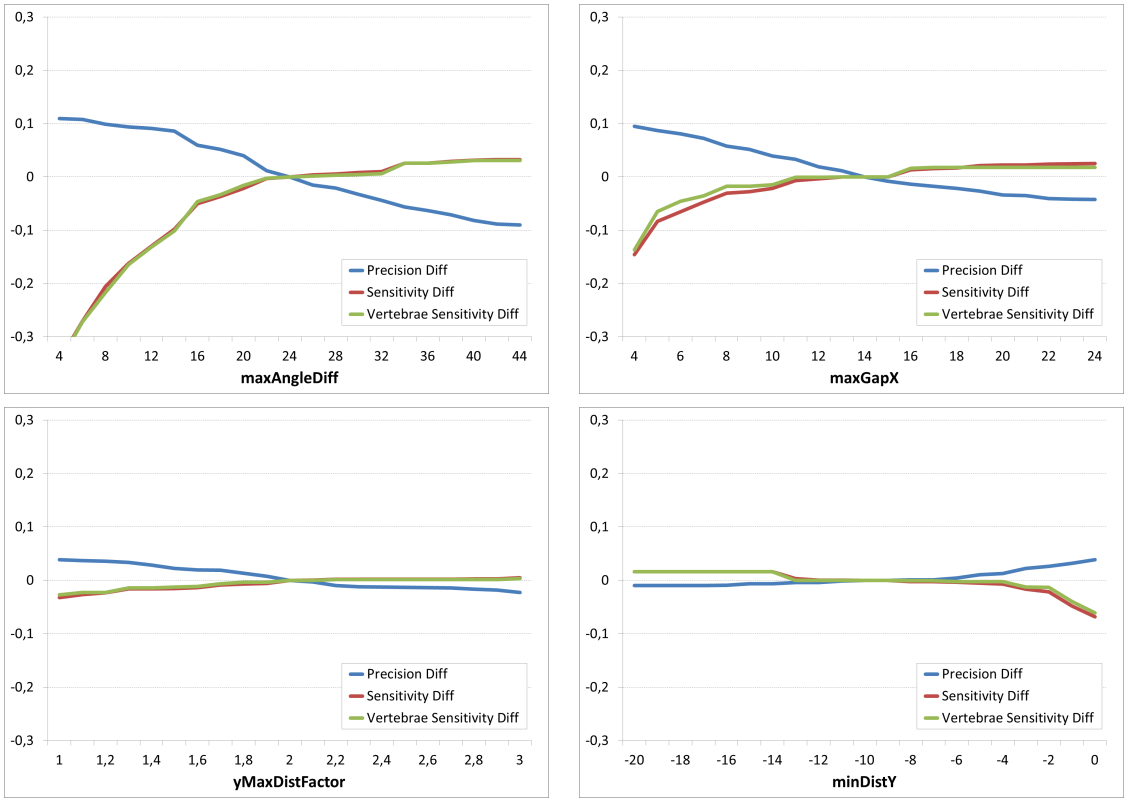



**Figure 3.17:** Parameter evaluation for neighbor classification using spatial relation information.

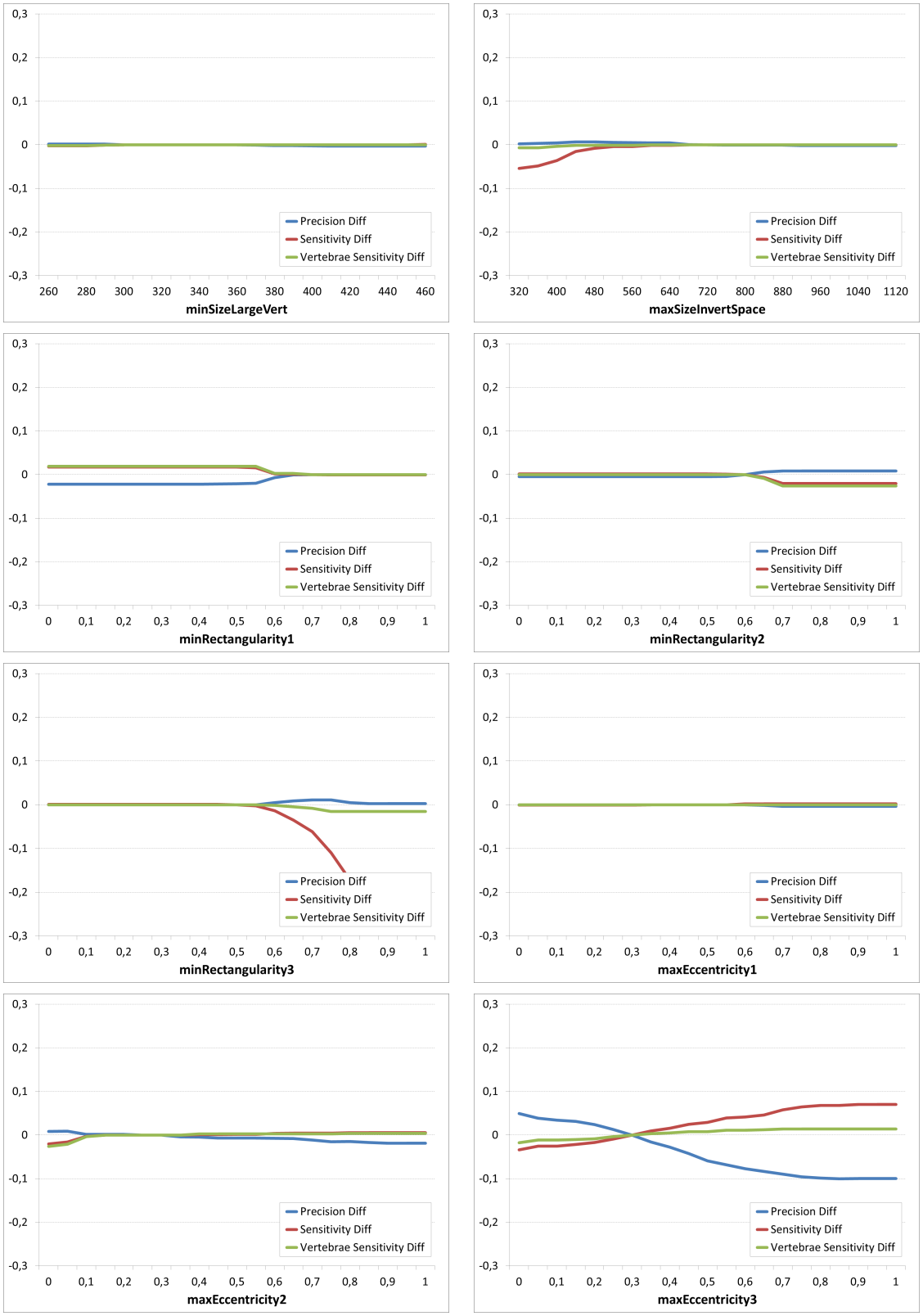



**Figure 3.18:** Parameter evaluation for neighbor classification based on object features.

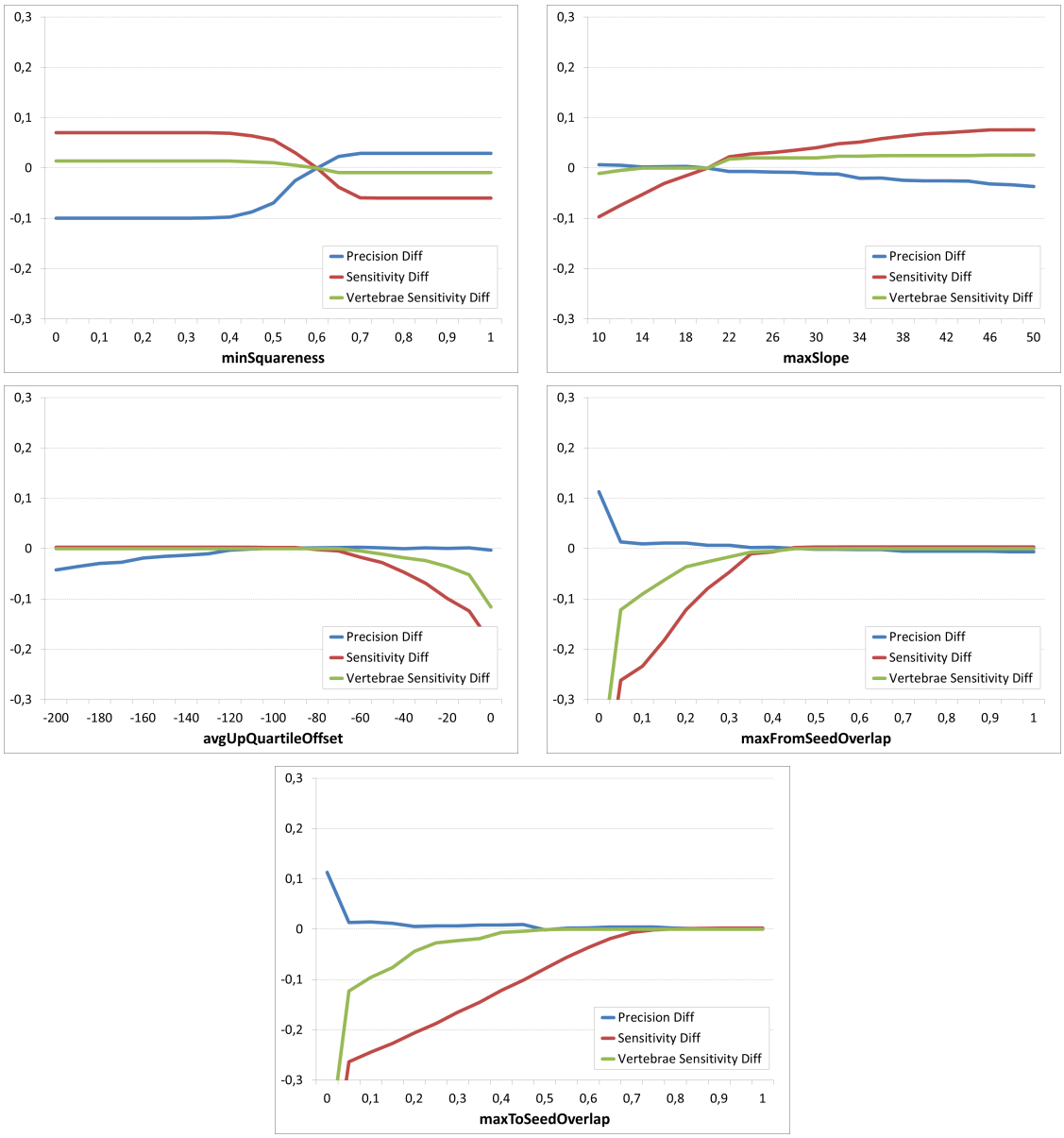



**Figure 3.19:** Parameter evaluation for neighbor classification based on object features (continued from Fig. 3.18).

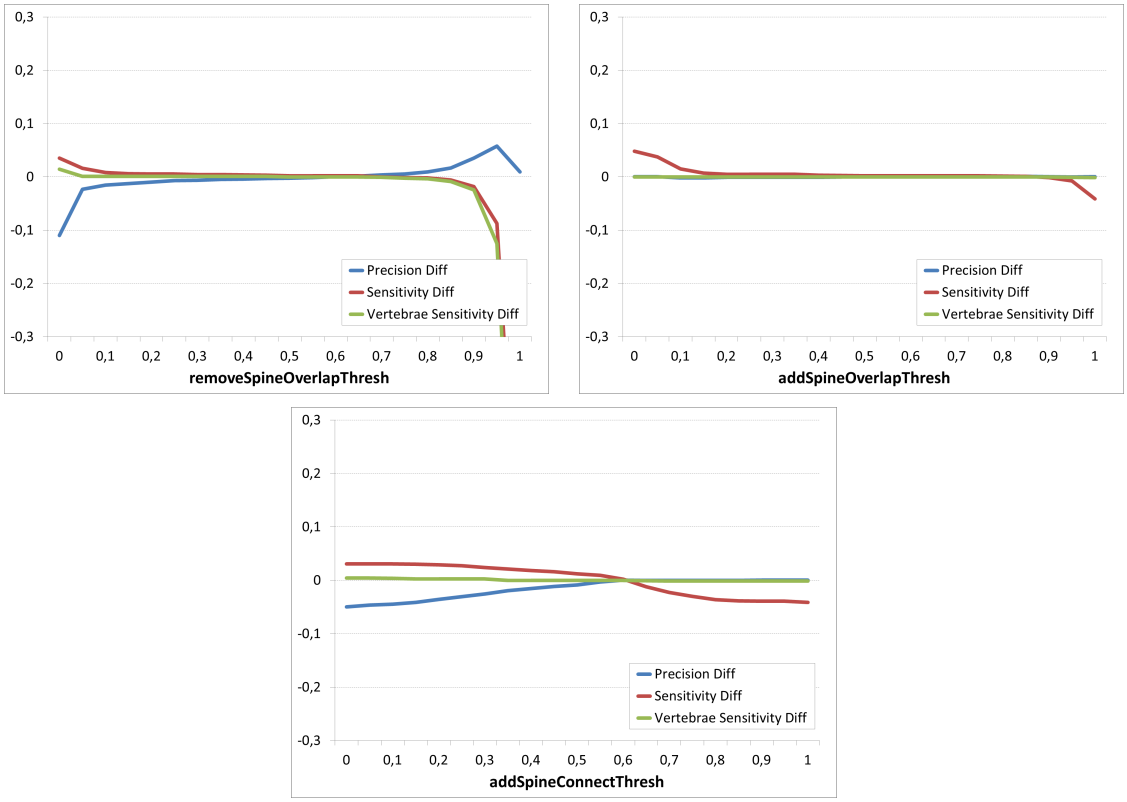



**Figure 3.20:** Parameter evaluation for classification refinement step.

**Table 3.7:** Comparison of the classification results under different vale settings for *maxGapX*, *addSpineOverlapThresh*, and *maxSlope* (all other parameters kept on their initial setting).

| *maxGapX* | *addSpine-OverlapThresh* | *maxSlope* | Precision | Sensitivity | Vertebrae Sensitivity |
|---|---|---|---|---|---|
| 14 | 0.7 | 20 | 0.88 | 0.84 | 0.95 |
| 14 | 0.7 | 32 | 0.87 | 0.89 | 0.98 |
| 14 | 0.0 | 20 | 0.89 | 0.89 | 0.95 |
| 14 | 0.0 | 32 | 0.87 | 0.92 | 0.98 |
| 13 | 0.7 | 20 | 0.90 | 0.84 | 0.95 |
| 13 | 0.7 | 32 | 0.89 | 0.89 | 0.98 |
| 13 | 0.0 | 20 | 0.90 | 0.89 | 0.95 |
| 13 | 0.0 | 32 | 0.88 | 0.92 | 0.98 |

According to the *addSpineOverlapThresh* plot, lowering this parameter value from 0.7 to 0.0 would increase the sensitivity by 4.8 percentage points, without affecting the precisions (or the vertebrae sensitivity). So these two are obvious candidates for optimization. A third very promising parameter is *maxSlope*, since the precision drops down only a little when increasing the value, while both sensitivities rise much stronger. So raising *maxSlope* from 20 to 32 would result in a decrease of 1.2 percentage points for precision, but raise sensitivity and vertebrae sensitivity by 4.8 and 2.3 percentage points respectively.

When changing the above mentioned parameters we have to consider that they might influence each other. To be sure if we should change all of them we compared the classification results on all combinations of new and initial value settings for *maxGapX*, *addSpine-OverlapThresh*, and *maxSlope*. Table 3.7 shows the results of this comparison. We can clearly see that changing all parameters is the best option since we gain 8 percentage points on sensitivity and 3 percentage points on vertebrae sensitivity while precision stays stable.

After changing those three parameters we performed the whole analysis on all parameters again to see if the adjustments affected any other parameters. However, most of the plots stayed similar to before and none indicated a clear possibility of improvement anymore. We therefore only show the plots for *maxGapX*, *addSpineOverlapThresh*, and *maxSlope* again under the new settings (see 3.21).

So far the results look very good on the training cases. However, it should be noted that, since we optimized our parameters on the basis of a per-parameter evaluation, we might be stuck in a local maximum.

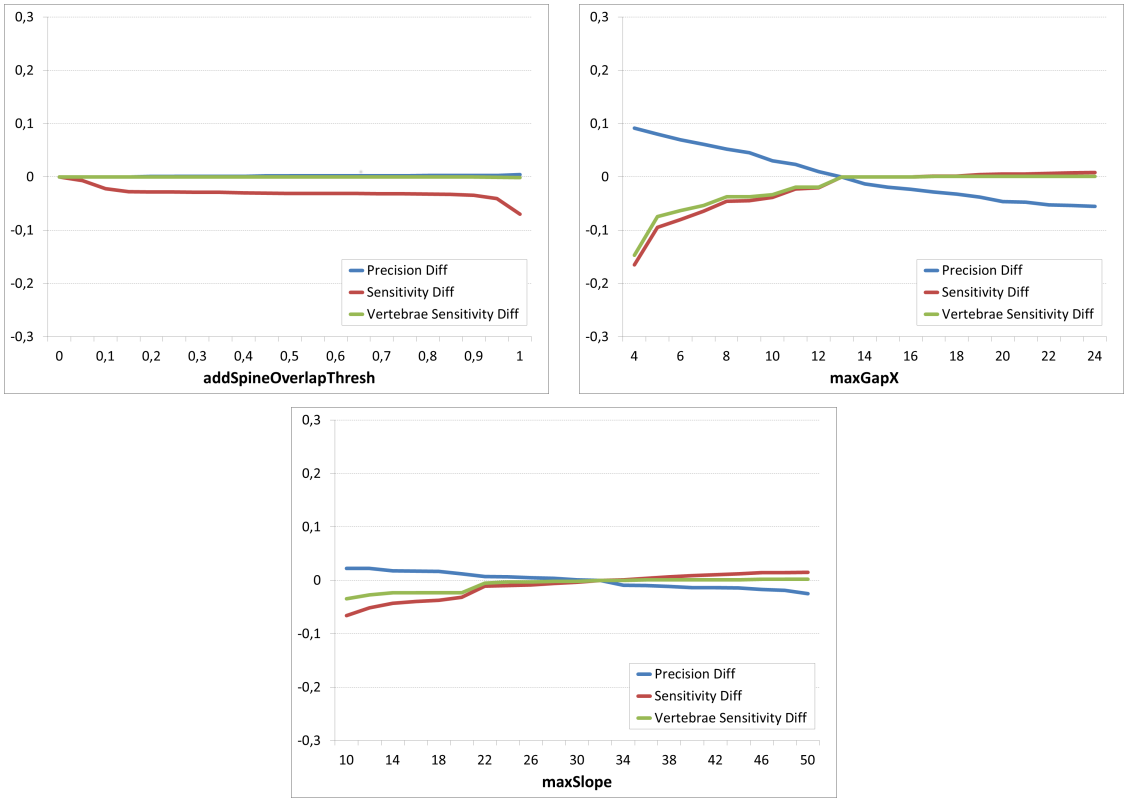



**Figure 3.21:** Re-evaluation of the parameters that have been adjusted after the initial parameter analysis.

### 3.3.3 Evaluation of Spine Detection

For the evaluation of the spine detection we used the same measures for precision, sensitivity, and vertebrae sensitivity as defined in the previous section (3.3.2). Table 3.8 shows the results of the detection on each of the datasets. We also wanted to compare how much improvement the classification refinement step brings. Therefore we computed the classification measures once directly after the iterative reconstruction and once after the refinement had also been performed.

The results show that the refinement step causes a significant improvement on precision and sensitivity for the training cases as well as for the test cases. The vertebrae sensitivity is not much affected by the refinement step. For the training cases it stays stable, for the test cases it even declines by 1 percentage point.

Comparing the average measures for training and test cases, we observe that the detection performs a little bit worse on the test cases. The final average precision on the test cases is 0.81 compared to 0.88 on the training cases. However, the average sensitivity and vertebrae sensitivity only differ slightly. The minimum measures for the test cases deviate stronger from the average as it is the case for the training cases.

Overall the detection results look very promising. We reach a high sensitivity of over 90% for the spine, so there are only a few parts of the spine that we miss. Also the precision is reasonably high with an average above 80%, but we are still including some false positives. The vertebrae sensitivity rates of above 95% are a very good prerequisite for the subsequent vertebrae identification.

### 3.3.4 Evaluation of Vertebrae Detection

As we saw in the previous section, the detection of objects that are (parts of) vertebrae is already very good. However the spine detection only labels them as part of the spine, so all we know for now is that we most likely have all vertebrae covered, which is a good prerequisite for the vertebrae detection.

Since the method in this step uses a Random Forest classifier trained with samples from the training data, the evaluation on the training cases was performed as a leave-one-case-out cross validation, hence not using any samples of the particular investigated case for training the classifier. For evaluating the test cases all training samples were used.

For the vertebrae detection we want to evaluate several qualities. First of all we are interested in how many individual vertebrae could be found, regardless of whether the assignment of the vertebra index is correct or not. In this respect we are also interested if any vertebra received more than one detection marker. Of course we also need to see how many wrong detections the algorithm creates. Finally we are interested in how many of the detected vertebrae also received the correct index.

We computed the following quantitative measures per case:

- $NV$ = total number of individual vertebrae in the reference;
- $NA$ = total number of assignments of a vertebra index to an object;

**Table 3.8:** Evaluation of the spine detection for each dataset (First 7 cases are the training cases). The table shows both, the results directly after the iterative reconstruction (see section 3.2.5) and the results after the additional classification refinement step (see section 3.2.6).

| | *Iterative Reconstruction only* | | | *Iterative Reconstruction + Refinement* | | |
|---|---|---|---|---|---|---|
| # | Precision | Sensitivity | Vertebrae Sensitivity | Precision | Sensitivity | Vertebrae Sensitivity |
| 1 | 0.85 | 0.88 | 1.00 | 0.92 | 0.92 | 1.00 |
| 2 | 0.82 | 0.93 | 0.97 | 0.90 | 0.94 | 0.96 |
| 3 | 0.71 | 0.82 | 0.97 | 0.84 | 0.91 | 0.96 |
| 4 | 0.67 | 0.86 | 0.97 | 0.80 | 0.90 | 0.98 |
| 5 | 0.75 | 0.84 | 0.96 | 0.87 | 0.89 | 0.97 |
| 6 | 0.81 | 0.92 | 1.00 | 0.92 | 0.95 | 1.00 |
| 7 | 0.83 | 0.84 | 0.98 | 0.94 | 0.93 | 0.97 |
| Avg.: | 0.78 | 0.87 | 0.98 | 0.88 | 0.92 | 0.98 |
| Min.: | 0.67 | 0.82 | 0.96 | 0.80 | 0.89 | 0.96 |
| 8 | 0.62 | 0.89 | 0.99 | 0.74 | 0.98 | 0.99 |
| 9 | 0.83 | 0.85 | 0.89 | 0.91 | 0.83 | 0.88 |
| 10 | 0.61 | 0.94 | 1.00 | 0.69 | 0.99 | 1.00 |
| 11 | 0.82 | 0.85 | 0.91 | 0.93 | 0.87 | 0.91 |
| 12 | 0.63 | 0.97 | 1.00 | 0.67 | 0.98 | 1.00 |
| 13 | 0.67 | 0.96 | 1.00 | 0.73 | 0.98 | 1.00 |
| 14 | 0.72 | 0.97 | 1.00 | 0.80 | 0.99 | 1.00 |
| 15 | 0.86 | 0.95 | 1.00 | 0.90 | 0.96 | 1.00 |
| 16 | 0.70 | 0.93 | 0.96 | 0.86 | 0.95 | 0.95 |
| 17 | 0.75 | 0.89 | 0.96 | 0.84 | 0.91 | 0.96 |
| 18 | 0.86 | 0.91 | 0.98 | 0.93 | 0.94 | 0.98 |
| 19 | 0.69 | 0.82 | 0.90 | 0.84 | 0.82 | 0.87 |
| 20 | 0.64 | 0.78 | 0.98 | 0.75 | 0.87 | 0.98 |
| Avg.: | 0.72 | 0.90 | 0.97 | 0.81 | 0.93 | 0.96 |
| Min.: | 0.61 | 0.78 | 0.89 | 0.67 | 0.82 | 0.87 |

- $CA$ = number of correct assignments of a vertebra index;
- $TP$ = number of vertebrae that were detected (regardless of the index assignment);
- $FP$ = number of detected objects not labeled as vertebra in the reference;
- $FN$ = number of undetected individual vertebrae;
- $MM$ = number of individual vertebrae that received more than one index assignment.

Based on these number we define the following additional measures:

$$Precision = \frac{TP}{TP + FP} \tag{3.35}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.36}$$

$$CA\ Rate = \frac{CA}{TP} \tag{3.37}$$

The summary of all these measures on all datasets is shown in table 3.9. The overall sensitivity and precision for vertebrae is good with 89% on average for the test cases. This means the algorithm currently misses about 2 vertebrae and detects 2 objects which are not vertebrae per case on average. Compared to the training cases the difference in sensitivity and precision is small. In fact, considering that the training cases all have 23 vertebrae, which is the upper limit for the vertebrae search, can already explain the difference in precision. Among the test cases are many with less than 23 vertebrae. So the way the algorithm is designed it might find additional objects that look enough like vertebrae above the last actual vertebrae, if it hasn't reached a total of 23 detections yet. One exemplary case is no. 12 in which the algorithm found 4 extra vertebrae.

The rate of correct assignments among the true positive detections reaches 74% on the test cases. The evaluation of the single cases shows that for 8 test cases the assignments are mostly correct while for the remaining the CA rate is very low. This ratio is similar among the training cases. A deeper investigation into the wrong assignments reveals that 73.2% of them are shifted by just one index, 24.4% are shifted by two, and 2.4% are shifted by 4 indices. The weak point of the algorithm shows when the consecutive detection misses a vertebra. Once one index is shifted all following ones will be shifted as well.

### 3.3.5 Evaluation of Time Performance

To assess the computation time for our method we considered the four main parts of the algorithm: pre-processing, pre-segmentation/seed detection, spine detection, and vertebrae detection. The measurements for the spine detection include the iterative reconstruction as well as the refinement step. As hardware we used a Dell Precision M3800 laptop computer with an SSD hard drive, 16GB memory and an Intel Core i7-4712HQ CPU at 2.30GHz. All computations were executed on a single core. Table 3.10 shows the runtime of the four individual parts as well as the overall runtime for all cases.

**Table 3.9:** Results of the vertebrae detection. NV = number of vertebrae in reference; NA = number of vertebrae index assignments; CA = number of correct vertebrae index assignments; TP = number of true positive detections of vertebrae; FP = number of false positive detections; FN = number of missed vertebrae; MM = multiple markers in the same vertebrae.

| # | NV | NA | CA | TP | FP | FN | MM | Precision | Sensitivity | CA Rate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 23 | 23 | 21 | 21 | 2 | 2 | 0 | 0.91 | 0.91 | 1.00 |
| 2 | 23 | 23 | 22 | 22 | 1 | 1 | 0 | 0.96 | 0.96 | 1.00 |
| 3 | 23 | 23 | 22 | 22 | 1 | 1 | 0 | 0.96 | 0.96 | 1.00 |
| 4 | 23 | 23 | 23 | 23 | 0 | 0 | 0 | 1.00 | 1.00 | 1.00 |
| 5 | 23 | 23 | 10 | 20 | 3 | 3 | 0 | 0.87 | 0.87 | 0.50 |
| 6 | 23 | 23 | 6 | 20 | 3 | 3 | 0 | 0.87 | 0.87 | 0.30 |
| 7 | 23 | 22 | 18 | 20 | 2 | 3 | 0 | 0.91 | 0.87 | 0.90 |
| Avg.: | | | | | | | | 0.92 | 0.92 | 0.81 |
| 8 | 17 | 19 | 2 | 17 | 2 | 0 | 0 | 0.90 | 1.00 | 0.12 |
| 9 | 21 | 17 | 9 | 16 | 1 | 5 | 0 | 0.94 | 0.76 | 0.56 |
| 10 | 23 | 23 | 9 | 19 | 4 | 4 | 0 | 0.83 | 0.83 | 0.47 |
| 11 | 23 | 20 | 10 | 17 | 3 | 6 | 0 | 0.85 | 0.74 | 0.59 |
| 12 | 18 | 22 | 3 | 15 | 7 | 3 | 0 | 0.68 | 0.83 | 0.20 |
| 13 | 17 | 18 | 17 | 17 | 1 | 0 | 0 | 0.94 | 1.00 | 1.00 |
| 14 | 18 | 17 | 14 | 14 | 3 | 4 | 0 | 0.82 | 0.78 | 1.00 |
| 15 | 18 | 19 | 18 | 18 | 1 | 0 | 0 | 0.95 | 1.00 | 1.00 |
| 16 | 17 | 16 | 13 | 15 | 1 | 2 | 0 | 0.94 | 0.88 | 0.87 |
| 17 | 18 | 18 | 17 | 17 | 1 | 1 | 0 | 0.94 | 0.94 | 1.00 |
| 18 | 18 | 17 | 14 | 17 | 0 | 1 | 0 | 1.00 | 0.94 | 0.82 |
| 19 | 18 | 18 | 17 | 17 | 1 | 1 | 0 | 0.94 | 0.94 | 1.00 |
| 20 | 22 | 23 | 19 | 19 | 4 | 3 | 0 | 0.83 | 0.86 | 1.00 |
| Avg.: | | | | | | | | 0.89 | 0.89 | 0.74 |

**Table 3.10:** Runtimes (in seconds) of the spine and vertebrae detection method on each dataset for individual parts of the algorithm and the overall runtime.

| # | Pre-Processing | Pre-Segmentation/ Seed Detection | Spine Detection | Vertebrae Detection | Sum |
|---|---|---|---|---|---|
| 1 | 5.05 | 35.16 | 7.52 | 0.01 | 47.74 |
| 2 | 5.39 | 32.67 | 8.79 | 0.02 | 46.86 |
| 3 | 7.12 | 42.48 | 10.36 | 0.01 | 59.97 |
| 4 | 3.95 | 24.54 | 12.05 | 0.01 | 40.56 |
| 5 | 6.44 | 36.43 | 15.15 | 0.02 | 58.04 |
| 6 | 6.61 | 57.07 | 6.47 | 0.01 | 70.16 |
| 7 | 6.14 | 53.87 | 6.96 | 0.01 | 66.98 |
| 8 | 5.17 | 30.62 | 27.10 | 0.02 | 62.91 |
| 9 | 6.33 | 28.83 | 5.13 | 0.01 | 40.30 |
| 10 | 8.31 | 62.37 | 26.12 | 0.02 | 96.82 |
| 11 | 7.00 | 35.33 | 6.65 | 0.01 | 49.00 |
| 12 | 6.83 | 43.69 | 13.85 | 0.01 | 64.39 |
| 13 | 3.93 | 20.47 | 18.89 | 0.02 | 43.31 |
| 14 | 4.01 | 20.87 | 19.64 | 0.02 | 44.54 |
| 15 | 4.52 | 30.80 | 16.20 | 0.01 | 51.54 |
| 16 | 4.84 | 20.11 | 7.16 | 0.01 | 32.12 |
| 17 | 5.11 | 19.85 | 12.93 | 0.01 | 37.90 |
| 18 | 5.16 | 21.32 | 7.28 | 0.01 | 33.77 |
| 19 | 3.66 | 18.56 | 16.23 | 0.01 | 38.46 |
| 20 | 6.79 | 49.32 | 51.05 | 0.02 | 107.18 |
| Avg: | 5.62 | 34.22 | 14.78 | 0.01 | 54.63 |

The most time-consuming part is the pre-segmentation and seed detection. This is caused by the iterative process that establishes the best watershed over-segmentation of the image. From each new tested watershed segmentation an object representation and some features have to be derived to determine the number of seed objects. How fast the best over-segmentation is found also significantly influences the runtime for this step. The pre-processing is reasonably fast with regard to the image processing operations performed.

The second most computationally intensive part is the spine detection. This is mainly due to the feature calculations that are required for the classification. The vertebrae detection is very fast because all required features already exist from the previous step. Running the Random Forest classifier is also very efficient and the execution of the loop, searching for 23 vertebrae, is barely noticeable. It should be noted that the training of the classifier is not included in the time measurements. In practice the training happens off-line and the classifier is just loaded from disk.

The average overall processing time of the spine and vertebrae detection is below one minute and hence makes it definitely feasible for an automatic pre-processing of images but could even be triggered on demand. The algorithm in its current state has not been optimized for speed, so we are sure there is a lot of potential of speeding it up, if it would be required.

## 3.4 Alternative Methods

The approach described in section 3.2 is rather heuristic. Especially the iterative spine reconstruction (section 3.2.5) with the subsequent refinement step (section 3.2.6) involve many manually set rules and thresholds. Working with different inspection tools to thoroughly analyze the feature space and understand their interrelations as well as optimizing the parameters as described in the evaluation makes us confident that we actually reached quite a good setup. The evaluation on the test data also confirms that. However, doing so required a lot of manual analysis work and we are always left with the feeling that we might have missed an important correlation of features which could improve the results.

The selection of features to be used for classification is a task that in our opinion humans can to best. This is also supported by Witten and Frank [172], who state that "The best way to select relevant attributes is manually, based on a deep understanding of the learning problem and what the attributes actually mean". However, capturing the correlations of those features and setting the right decision thresholds can quickly become overwhelming. For the initial seed selection this was still manageable due to the small number of features and the fact that we were aiming at a high precision, not caring much about sensitivity. Setting up the iterative construction with its many features was very time-consuming and complex.

We were therefore wondering if a machine learning approach would be able to reach similar or better results than our heuristic algorithm. Hence we implemented an alternative approach for the iterative spine reconstruction and the subsequent refinement. The basic

idea of the reconstruction remains the same but the classification is now done with an automatically trained classifier.

In the following sections we will describe and evaluate this alternative approach. Of the overall method we kept all steps before the iterative reconstruction unaltered. The green boxes in the flowchart in Fig. 3.22 indicate the stages that were changed.

### 3.4.1 Iterative Spine Reconstruction

The alternative iterative spine reconstruction starts with the same seed objects $S_0$ as the old approach. $S_k$ represents the set of unprocessed seed objects and $S_k^+$ the set of already processed seed objects at iteration step $k$. At each iteration step we now consider the set of relations $R_k = (S_k, C_k)$ with $C_k$ being the set of all direct and secondary neighbors (see eq. 3.6) of all objects in $S_k$.

Each relation $r = (s, c) \in R_k$ is described by the following features:

- $angleDiff = |c.angle_{MBR} - s.angle_{MBR}|$
- $corner23AbsDistX = |o_{bottom}.corner2x - o_{top}.corner3x|$
- $corner14AbsDistX = |o_{bottom}.corner1x - o_{top}.corner4x|$
- $corner23DistY = o_{bottom}.corner2y - o_{top}.corner3y$
- $corner14DistY = o_{bottom}.corner1y - o_{top}.corner4y$

with $o_{top}$ and $o_{bottom}$ determined as before: if $|c.corner2y - s.corner3y| > |s.corner2y - c.corner3y|$ then $o_{top} = c$ and $o_{bottom} = s$ otherwise $o_{top} = s$ and $o_{bottom} = c$. These features are basically the same as the ones that were used in the conditions for neighbor classification using spatial information (section 3.2.5.1). We just also added the spatial relations between the corner positions 1 and 4 of the minimum bounding rectangle.

We also assign the following features to each relation:

- $upperQuartileDiff = c.upperQuartile - avgUpQuartile$
- $stdDev = c.stdDev$
- $overlapFromSeed = c.overlapFromSeed$
- $overlapToSeed = c.overlapToSeed$
- $size = c.size$
- $eccentricity = c.eccentricity$
- $slopeXY = c.slopeXY$
- $rectFit_1 = c.rectFit_1$
- $squareFit_1 = c.squareFit_1$

However, these do not describe the relation itself but only the candidate object that is part of the relation. We do this, because different from the previous approach we do not want to separate the classification on the basis of spatial relations and the classification on the basis of object features in two subsequent steps, but let the classifier handle them at once.

Also for the features describing the candidate objects we used the same set as for the neighbor classification based on object features (section 3.2.5.2). The only additional feature here is $stdDev$.

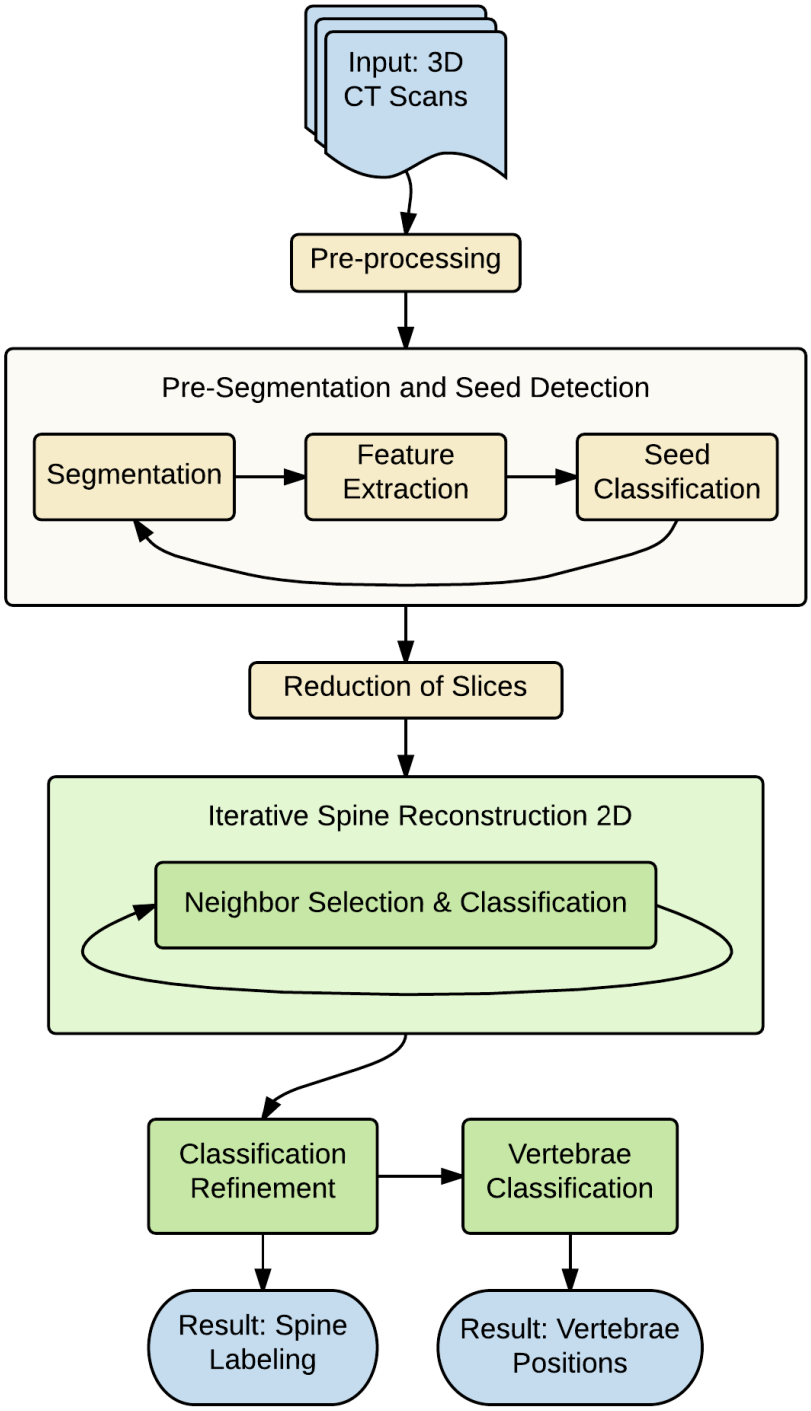



**Figure 3.22:** Overview of the alternative spine detection algorithm. The green boxes indicate the steps that were changed compared to the original approach.

The reason behind adding a few features is that we were thinking to consider them for the initial algorithm already, but it would have increased the complexity of the feature analysis and the manual setup of the rules. Furthermore, similar aspects of those features were already covered by other features. A supervised learning, however, could possibly gain some extra information from those features.

For the classification we used again the Random Forest classifier [14], because it has shown good performances in many applications [35] and is easy to configure. Only two classes are distinguished: "spine" and "non spine". At iteration step $k$ the Random Forest classifies all relations in $R_k$ which results in each $r = (s, c) \in R_k$ getting assigned a class weight $spineWeight$ indicating if the relation connects to another spine objects (given that $s$ is already assumed to be part of the spine). This spine weight is transfered to all $c \in C_k$ such that

$$c.spineWeight = \max\left(c.spineWeight, \max_{r_i=(s_m,c_n)\in R_k | c_n=c} r.spineWeight\right) \tag{3.38}$$

Now the sets for the next iteration are determined as

$$S_{k+1}^{+} = S_k^{+} \cup S_k \tag{3.39}$$

$$S_{k+1} = \{c \in C_k \mid c.spineWeight > spineWeightThresh\} \setminus S_{k+1}^{+} \tag{3.40}$$

The iteration stops when no more candidates are considered to be part of the spine and thus $S_k = \emptyset$ and $S_k^{+}$ contains all spine objects detected so far. For the setup of the $spineWeigthThresh$ and the training procedure of the Random Forest see the evaluation in section 3.4.4.1.

### 3.4.2 Classification Refinement

The alternative classification refinement also follows the same basic idea as before: We want to re-classify objects based on their connection and overlap to other spine/non-spine objects. For this re-classification we consider all objects that have been classified as spine in the previous step plus all of their neighbors. So our candidate set $C_{refine}$ is

$$C_{refine} = S_k^{+} \cup \left\{o_i \in O_1 \mid n_{i,j} \in N_1 \wedge o_j \in S_k^{+}\right\} \tag{3.41}$$

To capture the connection of a candidate to other spine objects we use the $contextPattern$ feature vector based on $spineWeight$. This should give the classifier enough information about the distribution and likelihood of surrounding spine and non-spine objects. Also the $spineWeight$ of the candidate itself is a valuable information. Additionally we also use the same object features we used in the previous step as well as the relative position of the candidate object for the refinement classification. So all in all the following features are considered for the objects in $C_{refine}$:

- $contextPattern(spineWeight)$
- $spineWeight$
- $upperQuartileDiff$
- $stdDev$
- $overlapFromSeed$
- $overlapToSeed$
- $size$
- $eccentricity$
- $slopeXY$
- $rectFit_1$
- $squareFit_1$
- $relPosX$
- $relPosY$
- $relPosZ$

The classification of all objects in $C_{refine}$ is performed by a Random Forest classifier which assigns the class weight $refineWeight$ to each object. The final set of spine objects is determined by

$$S^+_{final} = \{c \in C_{refine} \mid c.refineWeight > refineWeightThesh\} \tag{3.42}$$

For the setup of the $refineWeightThresh$ and the training procedure of the Random Forest see the evaluation in section 3.4.4.1.

### 3.4.3 Vertebrae Detection

The vertebrae detection algorithm is actually the same as described before (see section 3.2.7) except for one minor modification: The feature *spineClass*, which indicates different categories of spine objects is not available anymore. We therefore replaced it by the features $spineWeight$ and $refineWeight$ from the two Random Forest classification phases of the alternative spine detection.

### 3.4.4 Evaluation

In the evaluation of the alternative methods we will first take a look into the setup of the classification threshold weights for the iterative spine reconstruction and refinement. Then we will assess the performance of the spine and vertebrae detection on all cases.

#### 3.4.4.1 Evaluation of Alternative Spine Detection Parameters

Since the alternative approach for the spine detection is using a trained classifier there are almost no parameters that have to be set manually. Two are left however: The thresholds $spineWeightThresh$ and $refineWeightThresh$ on the classification weights to decide if an object should be considered as spine or not. Usually the default threshold would be

**Table 3.11:** Best thresholds for $spineWeightThresh$ and $refineWeightThresh$. Value$^{\mathrm{T}}$ = Values determined on the evaluation only using training samples from the training cases. Value$^{\mathrm{A}}$ = Values determined on the evaluation using training samples from all cases.

| Parameter Name | Value$^{\mathrm{T}}$ | Value$^{\mathrm{A}}$ |
|---|---|---|
| $spineWeightThresh$ | 0.55 | 0.7 |
| $refineWeightThresh$ | 0.45 | 0.25 |

0.5. But our sets of training samples are not guaranteed to be balanced and thus the class weights have to be expected to be biased.

We require two sets of training samples, one for the spine detection and one for the refinement classification. The training set for the spine detection is comprised of all relations between an object that is labeled as spine in the reference and another object (spine or non-spine). The samples for the refinement classifier are all objects which were classified as spine in the first classification step, plus their neighboring objects. The correct class for them is of course given again by the reference label. This means that the actual refinement sample set, that has to be used for training, depends on the setup of $spineWeightThresh$ in the previous step. Therefore we created a specific set for each setting of $spineWeightThresh$ in a range from 0.0–1.0 in 0.05 increments. During training we would then always choose the appropriate training set for the refinement classifier depending on the setup of the spine classifier.

To find the best setup for $spineWeightThresh$ and $refineWeightThresh$ we ran all combinations of them in a range from 0.0–1.0 in 0.05 increments on the seven training cases. As measures for the classification quality we used Precision, Sensitivity and Vertebrae Sensitivity as defined in section 3.3.2. Furthermore we tested two training sets for each classifier: One only containing samples from the training cases and one containing samples from all datasets. By performing all evaluations as leave-one-case-out cross validations we are not in danger of biasing the results.

Figures 3.23 and 3.24 show an excerpt of the evaluation plots for this analysis. Each plot was performed under a specific $spineWeightThresh$ and shows the distributions of Precision, Sensitivity and Vertebrae Sensitivity for the whole $refineWeightThresh$ range.

We determined the best setup for $spineWeightThresh$ and $refineWeightThresh$ by considering

$$\Phi = Precision + \frac{Sensitivity + Vertebrae\ Sensitivity}{2} \tag{3.43}$$

as combined detection quality measure. The thresholds that gave the best $\Phi$ for the two training samples setups are shown in table 3.11. The plots also reveal that those parameters are quite stable. Changing the thresholds a little does not affect the overall precision and sensitivity measures significantly.

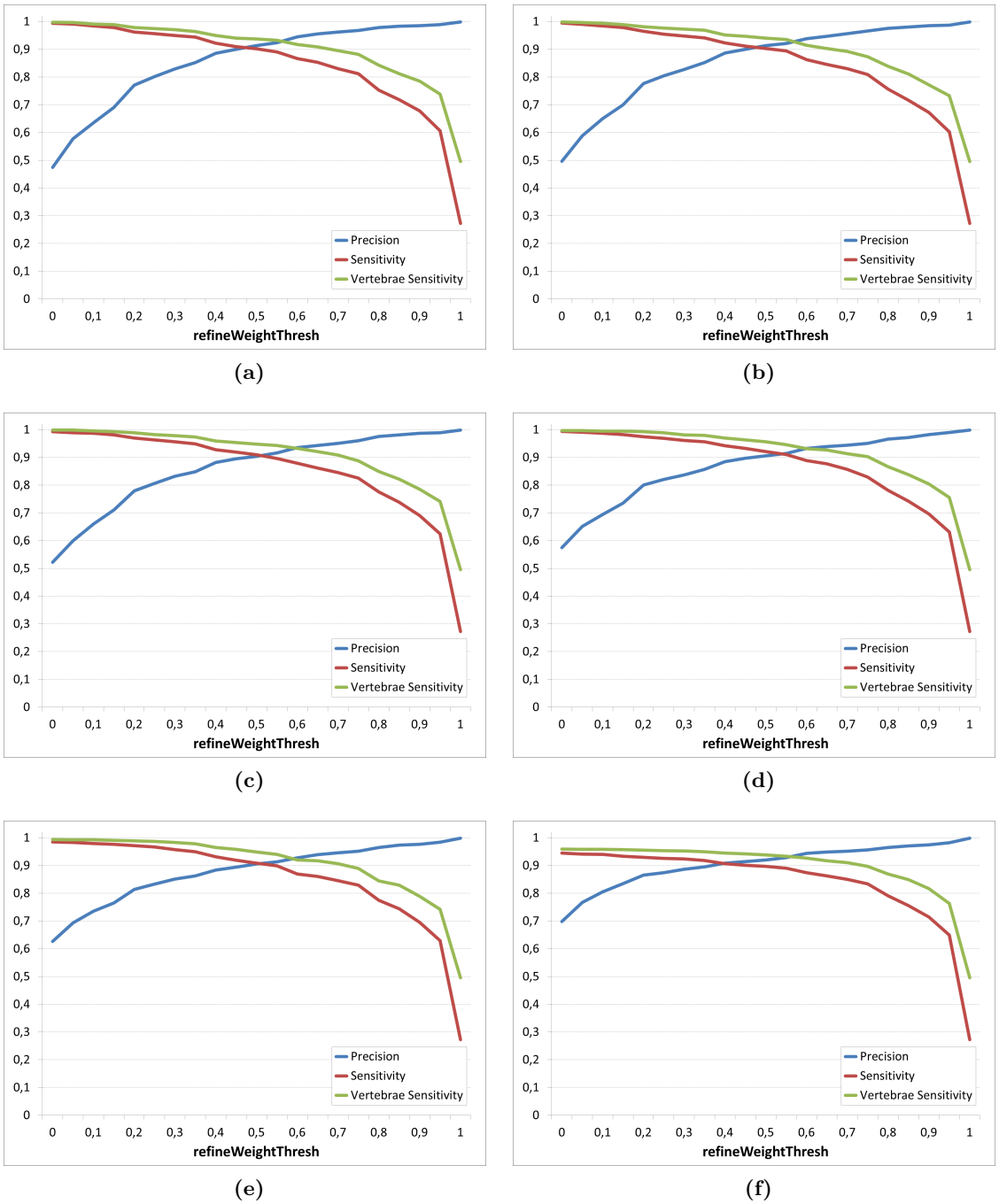



**Figure 3.23:** Excerpt of parameter evaluation for classification weight thresholds $spineWeightThresh$ and $refineWeightThresh$ on the training cases. Only training cases were used to collect samples for the classifier training. Setting of $spineWeightThresh$: (a) 0.1, (b) 0.25, (c) 0.4, (d) 0.55, (e) 0.7, (f) 0.85.

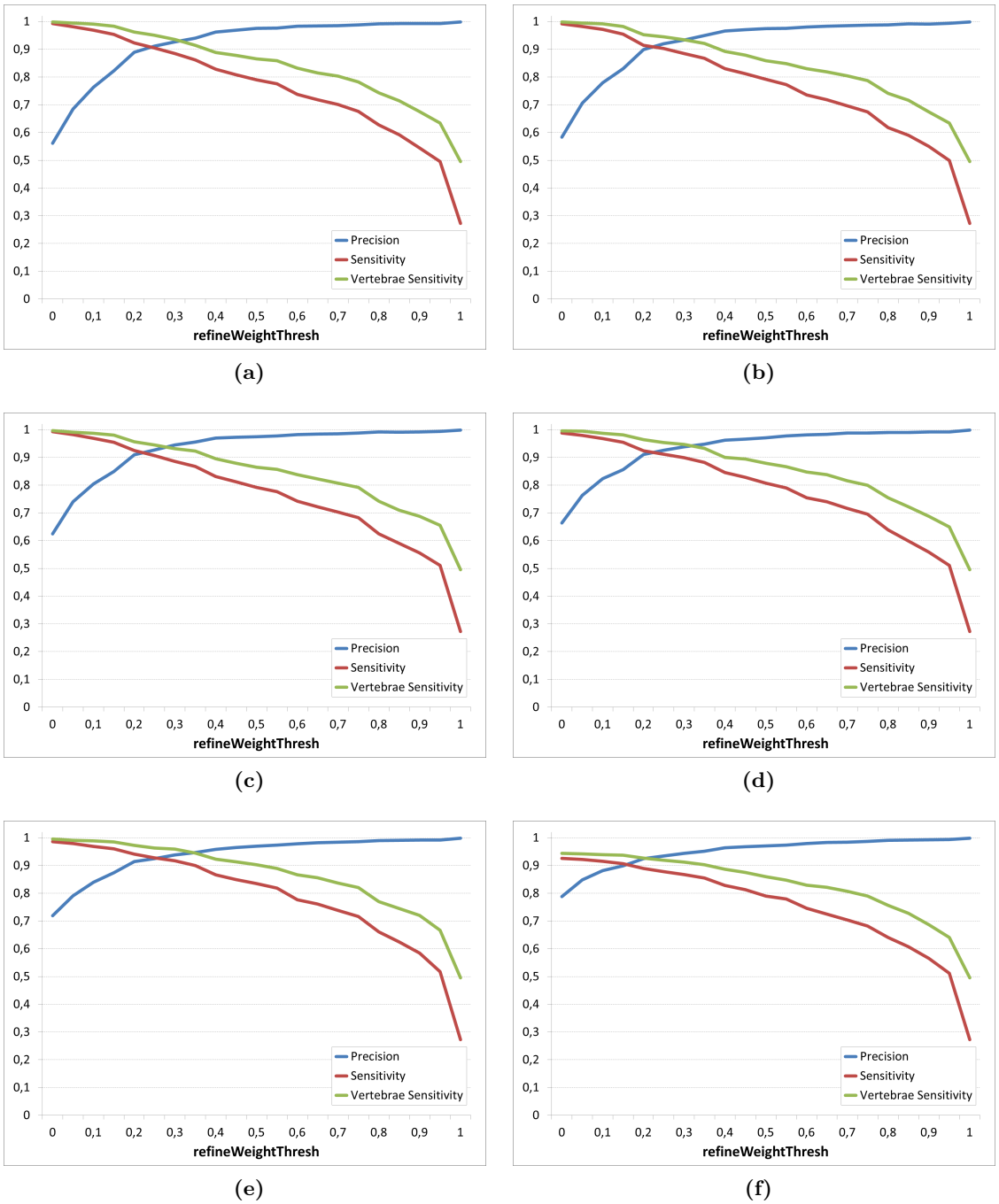



**Figure 3.24:** Excerpt of parameter evaluation for classification weight thresholds $spineWeightThresh$ and $refineWeightThresh$ on the training cases. All cases were used to collect samples for the classifier training. Setting of $spineWeightThresh$: (a) 0.1, (b) 0.25, (c) 0.4, (d) 0.55, (e) 0.7, (f) 0.85.

#### 3.4.4.2 Evaluation of Alternative Spine Detection

For the evaluation of the alternative spine detection we used the same measures for Precision, Sensitivity and Vertebrae Sensitivity as defined in section 3.3.2. Table 3.12 shows the results of the detection on each of the cases. The table compares the results for both types of classifier training sets: the ones only using samples from training cases and the ones using samples from all cases. The computation of the results was performed as a leave-one-case-out cross validation. The class weight thresholds were set according to the results of the parameter evaluation in the previous section.

The results show that using training samples from all cases improves the average precision not only for the test cases but also for the training cases. The average sensitivities remain unchanged. It is interesting that the minimum values for the training cases are stable on both sensitivities, but for the test cases the sensitivities decline when using the classifier that was trained with samples from all cases. However, this is caused by one case which improves on precision but declines in sensitivity, which is not unlikely if the training set changes.

Another interesting observation is that the average precision on the training cases is significantly higher than on the test cases (93% vs. 73%), while both average sensitivities on the training cases are lower than on the test cases. This indicates that the classification training did not generalize very well yet, which is not surprising given the small set of sample data. It also indicates that at least some of the test cases must differ in appearance from the training cases. One obvious difference we know is that among the test cases are several which do not show the whole spine, while all test cases contain the full spine. This can already explain the difference in classification performance. In summary we can see that a more diverse set of training data improves the overall classification performance.

If we compare the results of our heuristic approach with the method based on the Random Forest there are some interesting observations to be made: If we consider only the training cases, the machine learning approach is about on a par with the heuristic approach. And this is not even a fair comparison, because the heuristic approach was laboriously optimized by hand on the training data and thus the results on the training data are extremely biased (which is why we have a deliberate test set to independently assess the quality of the heuristic approach). However, the machine learning approach was tested with a leave-one-case-out cross validation and is thus unbiased. This is a strong indication that the trained classifier is actually performing very well.

If we compare the two approaches based on the test sets, which can be considered similarly independent from the training/development, we can see that the heuristic approach wins in terms of precision, while the machine learning approach wins in terms of sensitivity. As discussed before, the reason for this is that the classifier does not generalize well enough yet, with such little training data. Furthermore, the optimization of the class weight thresholds was performed only on training cases. To see the possible potential of the machine learning approach we also plotted the $refineWeightThresh$ over its whole range again, once considering only the test cases (Fig. 3.25a) and once considering all 20

**Table 3.12:** Evaluation of the alternative spine detection method. The table shows the results achieved with a classifier trained only on samples from training data as well as the results achieved with a classifier trained on samples from all cases. Values for $spineWeightThresh$ and $refineWeightThresh$ were set according to the evaluation in section 3.4.4.1. Evaluation was performed as a leave-one-case-out cross validation.

| | *Classifier trained on training cases* | | | *Classifier trained on all cases* | | |
|---|---|---|---|---|---|---|
| # | Precision | Sensitivity | Vertebrae Sensitivity | Precision | Sensitivity | Vertebrae Sensitivity |
| 1 | 1.00 | 0.86 | 0.93 | 1.00 | 0.85 | 0.93 |
| 2 | 0.96 | 0.97 | 0.98 | 0.97 | 0.97 | 0.96 |
| 3 | 0.68 | 0.97 | 0.97 | 0.78 | 0.96 | 1.00 |
| 4 | 0.90 | 0.88 | 0.96 | 0.92 | 0.87 | 0.94 |
| 5 | 0.87 | 0.96 | 0.97 | 0.90 | 0.95 | 0.96 |
| 6 | 0.99 | 0.95 | 0.97 | 1.00 | 0.95 | 0.97 |
| 7 | 0.89 | 0.93 | 0.96 | 0.91 | 0.94 | 0.98 |
| Avg.: | 0.90 | 0.93 | 0.96 | 0.93 | 0.93 | 0.96 |
| Min.: | 0.68 | 0.86 | 0.93 | 0.78 | 0.85 | 0.93 |
| 8 | 0.91 | 0.97 | 1.00 | 0.94 | 0.99 | 1.00 |
| 9 | 0.79 | 1.00 | 1.00 | 0.85 | 1.00 | 1.00 |
| 10 | 0.29 | 0.99 | 0.99 | 0.42 | 0.99 | 0.99 |
| 11 | 0.57 | 1.00 | 1.00 | 0.64 | 0.97 | 0.99 |
| 12 | 0.63 | 0.99 | 1.00 | 0.71 | 1.00 | 1.00 |
| 13 | 0.44 | 1.00 | 1.00 | 0.53 | 1.00 | 1.00 |
| 14 | 0.67 | 1.00 | 1.00 | 0.73 | 1.00 | 1.00 |
| 15 | 0.38 | 1.00 | 1.00 | 0.66 | 0.99 | 0.99 |
| 16 | 0.56 | 1.00 | 1.00 | 0.75 | 1.00 | 1.00 |
| 17 | 0.70 | 0.98 | 0.99 | 0.88 | 0.98 | 0.99 |
| 18 | 0.80 | 0.99 | 1.00 | 0.89 | 0.99 | 1.00 |
| 19 | 0.76 | 0.90 | 0.94 | 0.86 | 0.86 | 0.87 |
| 20 | 0.51 | 0.99 | 1.00 | 0.58 | 0.99 | 1.00 |
| Avg.: | 0.62 | 0.98 | 0.99 | 0.73 | 0.98 | 0.99 |
| Min.: | 0.29 | 0.90 | 0.94 | 0.42 | 0.86 | 0.87 |

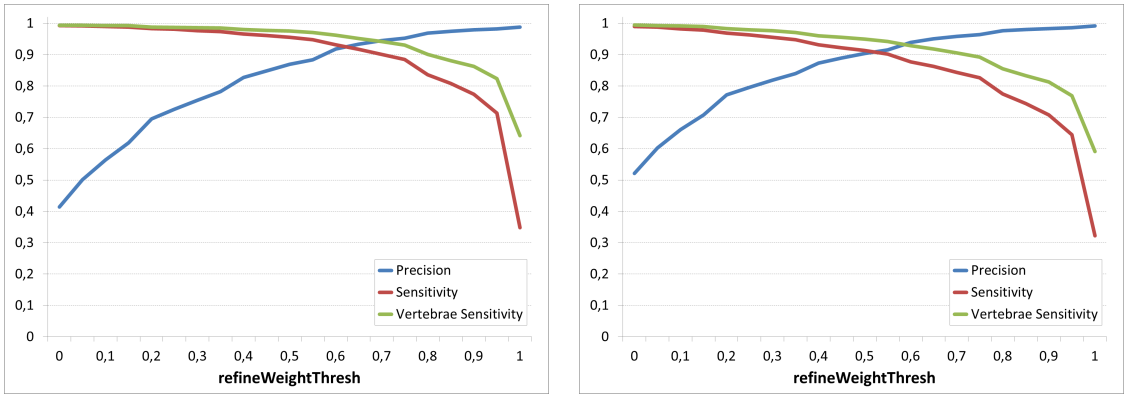



**Figure 3.25:** Plots of the classification performance over the whole range of $refineWeightThresh$. The $spineWeightThresh$ was set to 0.7. (a) Classification measures calculated using only the test cases; (b) Classification measures calculated using all 20 cases.

cases (Fig. 3.25b). If we choose the optimal $refineWeightThresh$ according to Fig. 3.25a we would reach precision, sensitivity, and vertebrae sensitivity values of 93%, 92%, and 95% respectively on the test cases. But this would significantly change the results on the training cases to 98%, 76%, and 86% (see Fig. 3.24e). However, considering Fig. 3.25b we can also find a setting that would result in average rates of 92%, 90%, and 94% on all 20 cases.

All things considered we can conclude that the machine learning approach is at least as good as the heuristic approach.

#### 3.4.4.3 Evaluation of Alternative Vertebrae Detection

The evaluation of the alternative vertebrae detection was conducted in the same way and with the same performance measures as for our initial approach (see section 3.3.4). Basis was the alternative spine detection that used the classifier trained on samples from all cases. The samples for the vertebrae classifier training were also taken from all cases. The evaluation results were obtained with a leave-one-case-out cross validation and are shown in table 3.13.

The average precision and sensitivity on the training cases are a bit higher than on the test cases. Since the training method for the vertebrae classification does not have an inherent bias towards the training cases, this difference can most likely be explained by the fact that the preceding spine detection performed already worse on the test cases. Compared to the initial vertebrae detection method precision and sensitivity are similar. Only the sensitivity on the training cases is a little bit lower for the alternative approach.

The rate of correct assignments reaches only 66% on the training cases, which is worse

**Table 3.13:** Results of the alternative vertebrae detection. NV = number of vertebrae in reference; NA = number of vertebrae index assignments; CA = number of correct vertebrae index assignments; TP = number of true positive detections of vertebrae; FP = number of false positive detections; FN = number of missed vertebrae; MM = multiple markers in the same vertebrae.

| # | NV | NA | CA | TP | FP | FN | MM | Precision | Sensitivity | CA Rate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 23 | 19 | 8 | 18 | 1 | 5 | 0 | 0.95 | 0.78 | 0.44 |
| 2 | 23 | 23 | 22 | 22 | 1 | 1 | 0 | 0.96 | 0.96 | 1.00 |
| 3 | 23 | 23 | 14 | 21 | 2 | 2 | 0 | 0.91 | 0.91 | 0.67 |
| 4 | 23 | 21 | 15 | 20 | 1 | 3 | 0 | 0.95 | 0.87 | 0.75 |
| 5 | 23 | 22 | 10 | 18 | 4 | 5 | 0 | 0.82 | 0.78 | 0.56 |
| 6 | 23 | 21 | 6 | 19 | 2 | 4 | 0 | 0.90 | 0.83 | 0.32 |
| 7 | 23 | 22 | 20 | 22 | 0 | 1 | 0 | 1.00 | 0.96 | 0.91 |
| Avg.: | | | | | | | | 0.93 | 0.87 | 0.66 |
| 8 | 17 | 17 | 17 | 17 | 0 | 0 | 0 | 1.00 | 1.00 | 1.00 |
| 9 | 21 | 15 | 0 | 11 | 4 | 10 | 0 | 0.73 | 0.52 | 0.00 |
| 10 | 23 | 23 | 10 | 19 | 4 | 4 | 0 | 0.83 | 0.83 | 0.53 |
| 11 | 23 | 22 | 9 | 16 | 6 | 7 | 0 | 0.73 | 0.70 | 0.56 |
| 12 | 18 | 23 | 16 | 16 | 7 | 2 | 0 | 0.70 | 0.89 | 1.00 |
| 13 | 17 | 18 | 16 | 16 | 2 | 1 | 0 | 0.89 | 0.94 | 1.00 |
| 14 | 18 | 18 | 17 | 17 | 1 | 1 | 0 | 0.94 | 0.94 | 1.00 |
| 15 | 18 | 19 | 17 | 17 | 2 | 1 | 0 | 0.89 | 0.94 | 1.00 |
| 16 | 17 | 17 | 17 | 17 | 0 | 0 | 0 | 1.00 | 1.00 | 1.00 |
| 17 | 18 | 18 | 18 | 18 | 0 | 0 | 0 | 1.00 | 1.00 | 1.00 |
| 18 | 18 | 17 | 13 | 16 | 1 | 2 | 0 | 0.94 | 0.89 | 0.81 |
| 19 | 18 | 18 | 0 | 17 | 1 | 1 | 0 | 0.94 | 0.94 | 0.00 |
| 20 | 22 | 23 | 14 | 20 | 3 | 2 | 0 | 0.87 | 0.91 | 0.70 |
| Avg.: | | | | | | | | 0.88 | 0.89 | 0.74 |

than in the initial approach. For the test cases the rate of correct assignments is 74%. A deeper investigation into the wrong assignments reveals that 74.5% are shifted by just one index, 24.5% are shifted by 2, and 1% is shifted by 6 indices. These rates are basically the same as in our initial approach.

All in all the results of the alternative vertebrae detection are a bit worse compared to the initial approach. One reason could be that the differentiation into different types of spine objects was actually more helpful than the classification weights.

## 3.5 Discussion

In this chapter we introduced a fully automatic method for detecting the spine in CT images. The method comprises several steps and utilizes the concept of object-based image analysis. This approach allowed us to efficiently incorporate domain knowledge into the analysis method, to benefit from a wealth of object features, and to employ contextual relations.

Our evaluation shows that the method reaches good results for detecting the whole spine as well as detecting and labeling single vertebral bodies. The method is also robust towards minor to medium pathological changes of the spine, such as fractures and lesions.

A current limitation of our method is that it performs a slice-wise 2D detection around the centerline of the spine, which means that vertebral bodies on slices away from the centerline, where only their outer margins are displayed and thus appear small and faint, are usually not detected. For locating and labeling vertebral bodies this does not matter, but we cannot claim to have reached a full 3D segmentation of the spine along the vertebral bodies.

In our main approach we rely on a heuristic classification scheme and a feature selection that was derived manually from an in-depth analysis of the feature space on a set of training cases. We fully agree, as Witten and Frank [172] state, that "The best way to select relevant attributes is manually, based on a deep understanding of the learning problem and what the attributes actually mean". With our heuristic approach we went a step further and tried to also understand the correlations and thresholds of those attributes (features), and map them into rules.

For very specific tasks with little expected variations this worked quite well, as the seed detection shows. The results on the test set are about as good as on the training cases from which we deducted the rules. For the more complex task of detecting the spine this became already much more difficult. The results on the test cases are actually not much worse but the analysis and manual tweaking even on just seven training cases did cost a lot of time and effort. Thus, for the last step, the localization of individual vertebrae, we already decided to go for a machine learning based approach. We still included our domain knowledge in the form of the selected features and the successive localization logic.

To assess how good the idea of following a heuristic approach on spine detection actually is, we also implemented an alternative approach using machine learning methods.

We kept the initial stages until the seed detection, of which we are confident that they are stable, but replaced the later stages in which the complexity for deducing appropriate rules manually was significantly higher.

The evaluation revealed that the machine learning approach for the spine detection is at least as good as the heuristic approach. For the localization of vertebrae the partly-heuristic approach performs overall a bit better than the machine learning approach. However, the advantage of using a trained classifier instead of manual heuristics is that it can be easily re-trained on additional cases or different features. Hence this approach has a lot of potential to be improved in the future with reasonable effort. Adapting the heuristic approach to new cases, which show variations that are not covered yet, would require significantly higher efforts.

In the evaluation we could also see that the heuristic approach generalized better and thus performed better on the test cases. On such a small set of cases this is very likely, because as humans we naturally infer the heuristic rules not only from the data at hand but also from our experience with medical images and domain knowledge (e. g. human anatomy). A machine learning approach only relies on the data given for training. Hence it generalizes better the more training data is available, providing that the training data was well selected to representatively cover the expected variations. Considering this and the fact that our machine learning approach performs already well, even with such little training data, leads us to conclude that our future developments for improving the spine and vertebrae detection should be based on the machine learning approaches.

This conclusion does not mean that the human domain knowledge and capability of capturing general concepts should be neglected. In contrary, the idea would be to focus on capturing the expert/domain knowledge in features, even very task-specific features, and let the classifier training correlate them and find appropriate thresholds.

One future extension we envision for the vertebrae localization is to train specific classifiers for each individual vertebra. Those classifiers should also include features that describe the relation from candidate objects to previously detected vertebrae locations. Like this we expect to improve on the currently very coarse rules regarding the relative location between vertebrae. In this way we also want to overcome the current limitation that we require the image to cover (almost) the whole spine and that the last lumbar must be visible to serve as starting point.

Further future improvement could go into adding other useful features. We believe that more contextual features that describe the relation of candidate objects to other anatomical landmarks outside the spine could be helpful. The lung could be such a landmark which is rather easy to identify on its own.

Most important for future improvements, however, would be to significantly extend the set of cases available for testing and training.

# Chapter 4

# Ultrasound-based Automatic Pregnancy Detection in Pigs

**Acknowledgments** All ultrasound and video camera images used for the work in this chapter were provided by and are used with permission of Big Dutchman Pig Equipment GmbH. I would like to thank Daniel Holling from Big Dutchman for the extraordinary cooperation on this project, it was truly one of the most exciting and fun projects I have worked on. Furthermore I would like to thank Dr. Sebastian Fietze for his consulting.

**Publications** Section 4.1 of this chapter is based on a paper on the detection of embryonic vesicles, which was published in the *Proceedings of the 2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI)* [138] . However, section 4.1 has been significantly extended compared to the paper.

The diagnosis of the pregnancy status of sows and gilts by means of ultrasound imaging is a common procedure in commercial pig farming. Early detection of non-pregnant pigs is an important factor for the efficiency of the breeding process [170]. Currently the diagnosis is done manually by a trained expert. Different studies [170, 38] report sensitivity/specificity rates ranging between 82.7%/33.3% and 96.5%/96.0% depending on the exact day of pregnancy and the quality of the ultrasound probe. However, considering typical farm sizes with several hundreds to several thousands of pigs, manual diagnosis by a specialist means significant effort and costs.

The company Big Dutchman (Vechta, Germany) developed an innovative feeding station, which pigs in a free-range group can enter individually and feed undisturbed by other animals (see Fig. 4.1). This system can be equipped with a remote controllable ultrasound probe (see Fig. 4.2). The probe is installed on a steerable arm which allows to move the probe and engage it at the uterus position of the animal to remotely acquire ultrasound images (see Fig. 4.2b, 4.2c , and 4.2d). The motor unit controlling the arm is attached as an extension on top of the feeding station (see Fig. 4.2a).

A camera installed behind a hatch in the side wall of the feeding station shows the flank of the animal while feeding. The camera delivers frames at a fixed resolution of $727 \times 464$ pixels. The lens of the camera has a fish-eye effect.

Our goal in collaboration with Big Dutchman was to develop an automatic acquisition of ultrasound image series while the animal is feeding, followed by an automatic analysis of the ultrasound images to determine the pregnancy status of the animal. Achieving this would be a world novelty and significantly reduce the manual work and costs required for the pregnancy determination. Our part in this development was to find solutions for all image processing related tasks.

At three stages of the system image processing and analysis is required: First, while the animal is feeding the uterus position has to be tracked on the camera images to navigate the ultrasound probe to the right position. Also the ultrasound arm and probe has to be tracked on the video feed, since neither the construction of the arm nor the motor unit are designed to give feedback about the location of the ultrasound arm. We describe our solution for this stage in section 4.2.

The second stage that requires image processing comes after the acquisition of an ultrasound image series. The images have to be analyzed to determine the pregnancy status by detecting appearances of embryonic vesicles. Section 4.1 covers our methods for the automatic detection of pregnancy on ultrasound images.

The third requirement is a procedure to detect dirt on the camera pane. This is important to ensure that the camera view for the position detection is not occluded. A method for dirt estimation on the video images is described in section 4.3.

The question might arise if there are no other possibilities to check a pig for pregnancy that could be employed? One option would be testing the urine. However, it is virtually impossible to automatically collect urine reliably and uncontaminated in a live stable setting. Another option would be a blood test. Again, collecting a blood sample

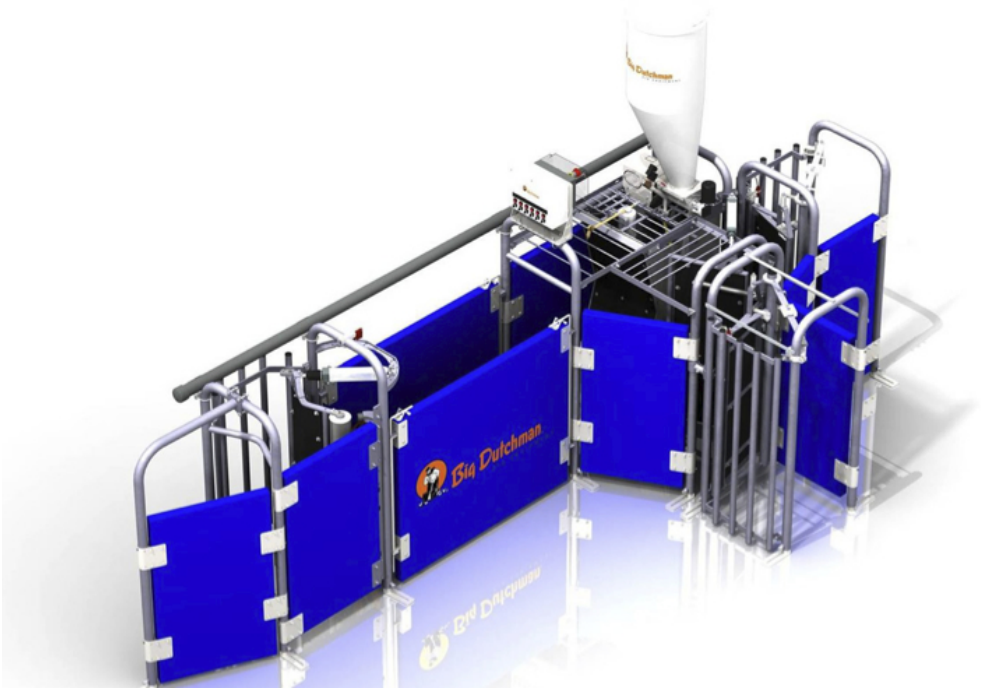



**Figure 4.1:** Illustration of the automatic feeding station without ultrasound extension. (Image courtesy of Big Dutchman Pig Equipment GmbH)

automatically is very difficult and bears the danger of injuring the animal. Furthermore pigs are very sensitive to pain, so such a procedure would induce unnecessary stress. Also both methods of testing urine or blood would require laboratory equipment and expertise causing additional costs [39].

## 4.1 Detection of Embryonic Vesicles

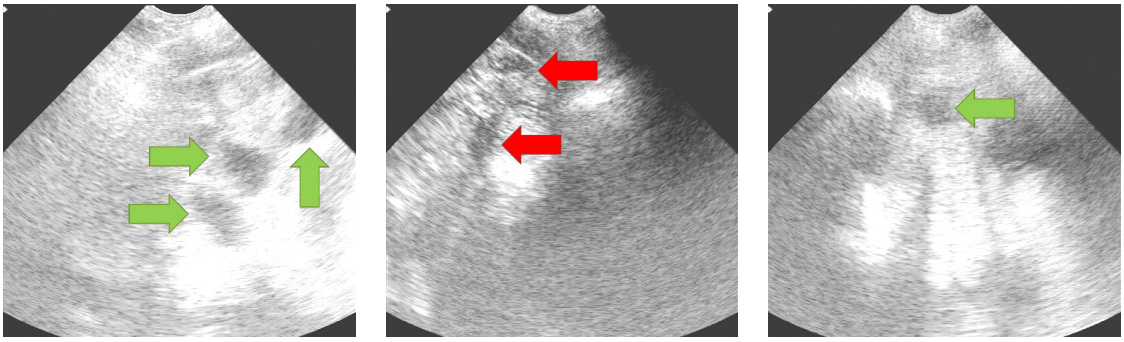
Decisive hints for pregnancy are embryonic vesicles which appear as dark spots on the images. Sometimes they are well-defined as in Fig. 4.3a but they can also be very subtle as Fig. 4.3c shows. However, blood vessels, the bladder, intestinal loops, or imaging artifacts can produce similarly appearing structures which makes the discrimination difficult (see Fig. 4.3b).

For the task at hand it is not required to detect and identify individual embryonic vesicles, since we only want to know if the ultrasound series shows a pregnancy or not. For this one conclusive hint of an embryonic vesicle is enough.

In our approach, we employ a task-specific strategy for building segmentation hierarchies to determine candidates. Within this hierarchy a trained classifier searches for embryonic vesicles. We also introduce a new feature set describing the border of a candidate region, which has a significant impact on the success of the classification.

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

**Figure 4.2:** Photos of the SonoCheck feeding station: (a) Ultrasound extension installed on top of the basic automatic feeding station. (b) Inside of the feeding station with a pig and ultrasound arms on each side. Only the left one is mounted with an ultrasound probe. (c) Ultrasound probe located at the flank of the animal, not engaged. (d) Engaged ultrasound probe at the uterus position. (All photos courtesy of Maike Wübbels)



(a) (b) (c)

**Figure 4.3:** Example ultrasound frames showing suspicious structures: (a) Clearly visible embryonic vesicle. (b) Other structures, no sign of pregnancy. (c) Very subtle embryonic vesicle. (©2014 IEEE)

### 4.1.1 Related Work

Biometric measurements to determine the gestational stage are common in human fetal ultrasound. The goal is usually to estimate the gestational age and to detect possible problems in the development of the fetus. Usual measurements include the head circumference and diameters, abdomen circumference, and femur length. Considerable research has been conducted to automatize these measurements.

Matsopoulos and Marshall [86] presented a method for detecting the fetal head using morphological operations. The same direction is taken by Hanna and Youssef [49], but they additionally combined it with the Hough transform. In the same category of approaches falls the work of Shen et al. [146] who apply morphological thinning to obtain a skeleton image and then perform an iterative randomized Hough transform to detect the fetal head. Nithya and Madheswaran [105] also employ the Hough transform in their approach to extract the abdominal circumference. The initial detection is refined by a gradient vector field snake.

A partly different direction is taken by Ni et al. [104] who use an AdaBoost classifier on Haar-like features to detect the head contour. The Hough transformation is then used for fitting the ellipse into the detected contour parts. Also Carneiro et al. [19] rely on machine learning techniques in their very promising approach to detect and measure basically all typical fetal anatomies. They use a probabilistic boosting tree and employ a large database of ultrasound images with expert annotations of fetal anatomical structures. Another work that employs boosted classifiers was presented by Maraci et al. [83], however limited to fetal head detection. They used a boundary fragment model to feed the classifier and an iterative ellipse fitting for the final estimation.

A rather different approach was proposed by Jardim and Figueiredo [63], who used deformable models for detecting femur and head contours.

For further reading on automatic human fetal ultrasound measurements we refer to Rueda et al. [130] who presented an evaluation of recent methods among participants in a challenge of segmentation methods for biometric measurements from fetal ultrasound images. Their publication also gives an overview of previous work on the topic.

However, ultrasound imaging in pig production is a different matter: Since the cost factor plays an important role, cheaper probes are used that provide a lower image quality than common diagnostic ultrasound devices. This makes the transition of ideas from methods for human fetal ultrasound difficult. As we could see above, most methods rely on the extraction of contours. In the images given for our task though, no pronounced contours delineate the target structures. Also the Hough transform is not easily applicable for our problem, because the shapes of the embryonic vesicles are not regular enough. Furthermore, we might have other similar looking structures in our images (see Fig. 4.3 again). In contrast to that, the head or femur in human fetal ultrasound images appear as rather regular and pronounced structures.

Employing a machine learning approach seems in general a suitable approach on ultrasound images, especially since the ambiguous nature of the images makes it difficult for

a human to grasp the interrelations of different features in the image.

Another thing to bear in mind regarding pregnancy detection in pig production is that the diagnostic question does not regard specific biometric measurements, but rather asks if an image series shows a pregnancy or not.

As mentioned before, ultrasound imaging for pregnancy detection is a common procedure in commercial pig farming. Actually, this topic has been studied for at least 30 years already [162, 12, 4]. However, even recent studies [170, 38] only mention this being done manually by a human expert. To our best knowledge an automatic ultrasound-based pregnancy detection for pigs has not been attempted yet.

The studies by Williams et al. [170] and Flowers et al. [38] report sensitivity/specificity rates ranging between 82.7%/33.3% and 96.5%/96.0% depending on the exact day of pregnancy and the quality of the ultrasound probe. They also determined that the days 21 - 24 after insemination are the best for diagnosis.

### 4.1.2 Methods

The input data for our method are ultrasound image series with 150 - 250 frames, scanned on days 22 - 25 after insemination with a CA20R probe operated at 4.5 MHz and a scanning depth of 150 mm. The image resolution is 640×480, with the relevant image data located in a 480×400 subregion. In the following we will sometimes refer to an image series as a case, since it represents one image acquisition session of an animal.

The basic concept of our approach is a region-based classification of candidate structures in the image series. The advantage of this compared to a pixel-based classification is that we can use more expressive features describing the actual structure we are looking for. This is especially important since the differences between negative and positive structures can be very subtle. However, a region-based classification requires a good initial segmentation that captures the boundaries of the target structures. For this purpose we developed a hierarchical segmentation method which is described in detail in section 4.1.2.1.

Within the segmentation hierarchy a trained classifier finds embryonic vesicles as described in section 4.1.2.2. Several standard features are used for the classification. However, to properly describe the main visually discriminating attribute which is the strength of the border of a dark region, we developed a new set of features as described in section 4.1.2.3.

#### 4.1.2.1 Segmentation hierarchy

The basic concept of segmentation hierarchies is to produce an initial over-segmentation of an image and build one or multiple hierarchies by iteratively merging segments with the goal that target objects are properly covered by a segment at some level in the hierarchy. Considerable research has been conducted in that direction. Paris and Durand [111] as

well as Arbeláez et al. [5] stand as examples for recent advances in the field and also give a good overview of previous work.

However, the published approaches cannot be easily transfered to our specific problem. This is mainly due to the low signal-to-noise ratio in the ultrasound images and the lack of well defined borders. We therefore developed an approach that takes into consideration the specific requirements for the resulting segmentation to serve our classification purpose. This is supported by a statement by Noble et al. [106] who conducted a comprehensive survey on ultrasound image segmentation and conclude that "a good ultrasound image segmentation method needs to make use of all task-specific constraints or priors. This is an explicit or implicit assumption in all successful methods in the literature."

We start with a pre-processing of the image: We cut the region of interest (ROI) of relevant image data (see Fig. 4.4a) and downsample by factor 2 to 240×200. Due to the noise level we do not lose relevant image information by downsampling and gain a speedup for the subsequent steps. The blank corners in the upper left and right are now filled by mirroring the image data along the edge to avoid artifacts in the further pre-processing. Then a Gaussian smoothing with $\sigma = 3.5$px is applied to the image. We denote the result as $I_{smooth}$. This is followed by a reduction of gray levels from 256 to 32. We call te resulting image $I_{iso}$. If we now consider the isolines of $I_{iso}$ (Fig. 4.4b) as segmentation borders we can observe that local minima are surrounded by concentric segmentation layers. The beneficial property of this segmentation is that we can always find an isoline that delineates an embryonic vesicle the way a human would expect it, with one limitation: Where two dark areas are located closer together and the intensity ridge is lower between them, the iso-layer leaks. To overcome this issue, we perform a watershed segmentation on $I_{smooth}$ that separates local minima (Fig. 4.4c). We perform the watershed implementation by Hahn and Peitgen [48] which we initialize with the positions of all local minima found in the smoothed image. The leaking problem is now eliminated by intersecting the isolines with the watershed lines. Additionally the overall segmentation is cut at the mirroring borders to only consider the original image content (Fig. 4.4d). This over-segmentation defines our initial object set $O$.

Starting from each local minimum object $o_m \in O_{min}$ with

$$O_{min} = \left\{ o_i \in O \mid o_i.mean(I_{iso}) < \min_{o_j \in O | n_{i,j} \in N} o_j.mean(I_{iso}) \right\} \tag{4.1}$$

a segmentation hierarchy is built by iteratively merging the current object $o_m$ with all its neighboring objects $o_n$ that have a higher iso-level and share more than 30% of their border with $o_m$. An additional constraint for the merge is that $o_m$ has to share at least 60% of its border with those $o_n$ that fulfill the previous conditions. Furthermore, parts of borders that touch the outside boundary of the valid image area are not considered for the relative border calculation.

Finding embryonic vesicles within those multiple segmentation hierarchies is done in the classification step. All possible segmentations that are captured in the hierarchies are considered candidate regions for the classification.

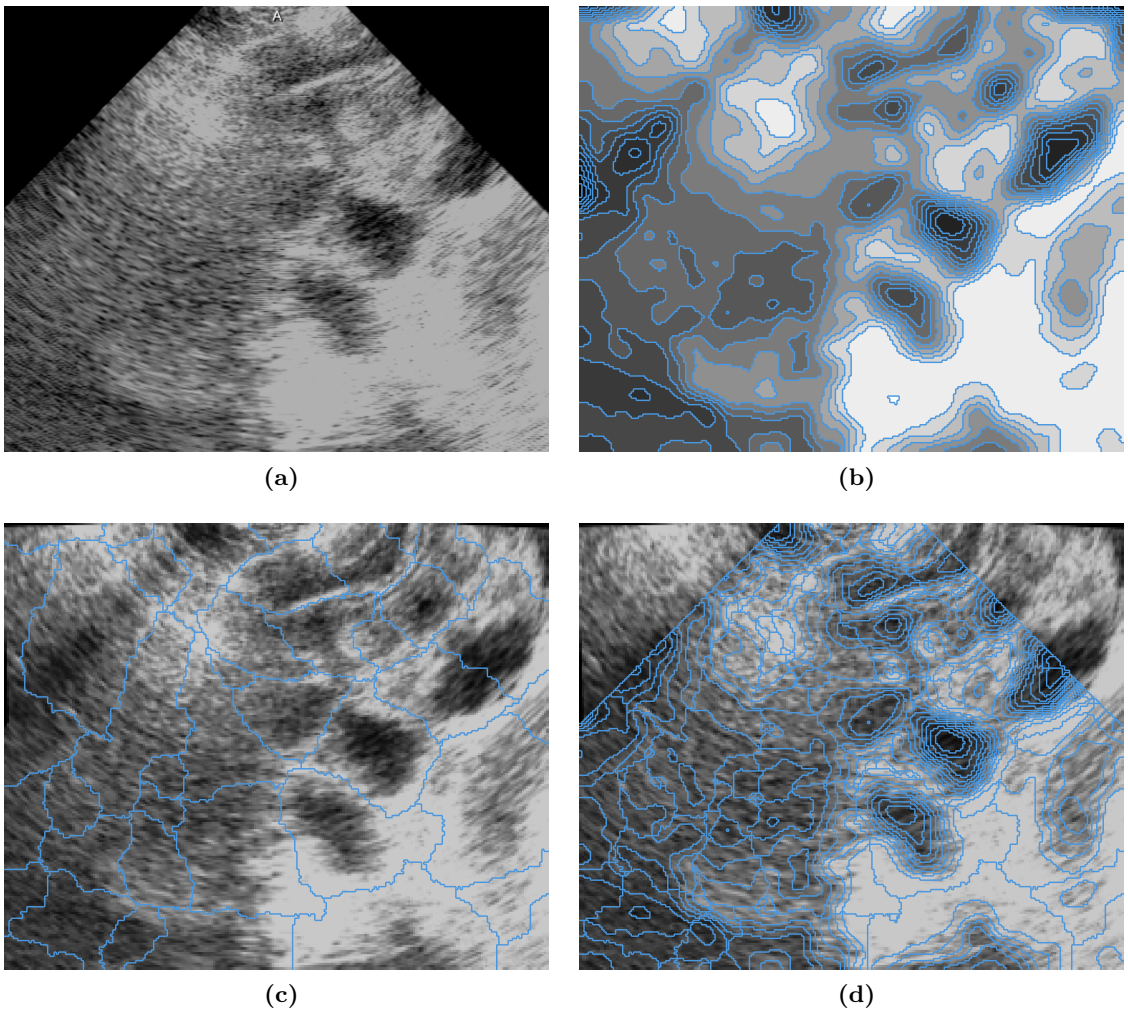


**Figure 4.4:** Building the initial segmentation: (a) ROI. (b) Isoline segmentation. (c) Watershed segmentation seeded by local minima. (d) Final over-segmentation: combination of isoline and watershed segmentation cut at the original image edges. (©2014 IEEE)

#### 4.1.2.2 Classification

For classification of the candidate regions a Random Forest [14] classifier with 25 trees is used. To get the overall result for an image series we classify all candidate regions that were generated in the previous step. However, we are not particularly interested in individual embryonic vesicles but only in the overall result for an image series. It is sufficient that one embryonic vesicle is detected once in one frame to conclude that the image series shows a pregnancy. Therefore, we consider as overall classification result the highest positive classification weight found for a candidate in the image series.

The following basic features were used for classification:

- $mean(I_{iso})$
- $centerX$
- $centerY$
- $circularity$
- $eccentricity$
- $ellipticity$
- $elongation$
- $size$

Furthermore we define the following features describing the connection of a candidate region $o_i \in O$ to the background

$$o_i.borderSize_{\emptyset} = o_i.perimeter - \sum_{n_{i,j} \in N} n_{i,j}.mcBorder2dSize \tag{4.2}$$

$$o_i.borderRelative_{\emptyset} = \frac{o_i.borderSize_{\emptyset}}{o_i.perimeter} \tag{4.3}$$

Another set of features describes the relative location of an object in regard to the origin of the ultrasound rays and the orientation of the object relative to the ray that passes through its center of gravity:

$$o_i.distToOrigin = \sqrt{(o_i.centerX - r.x)^2 + (o_i.centerY - r.y)^2} \tag{4.4}$$

$$o_i.angleDiff = \left| o_i.slopeXY - atan2(o_i.centerY - r.y, o_i.centerX - r.x) \cdot \frac{180.0}{\pi} \right| \tag{4.5}$$

with $r$ representing the coordinates of the ultrasound ray origin.

In addition to those, a set of novel "borderness" features, describing the border area of a region, was provided for the classifier and is described in detail in section 4.1.2.3.

#### 4.1.2.3 Borderness features

For the human observer the most discriminate attribute to identify an embryonic vesicle is the strength of the border around dark regions. However, those borders can appear more

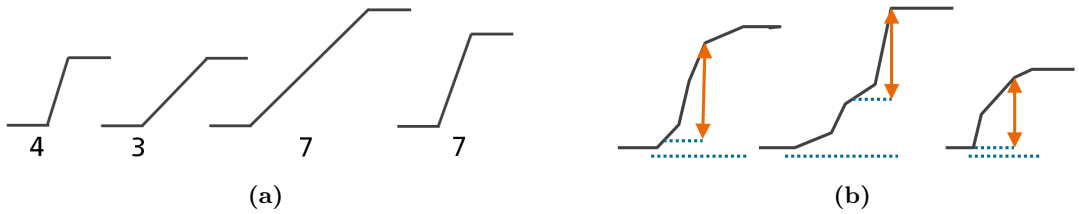



**Figure 4.5:** Simplified illustrations of borderness features: (a) Intensity profiles and exemplary values for *absBorderness*. (b) Finding the 7 unit long segment with the maximum gradient along the intensity profile. (© 2014 IEEE)

or less blurry, with different contrast and are not always fully closed around the vesicle. Nevertheless a human observer is somehow able to interpret the correct delineation even into subtle, incomplete and diffuse border appearances. Therefore we developed a novel feature set, which we call "borderness features", which capture the essential characteristics of the border area of a region.

The calculation is based on the gray-level reduced image $I_{iso}$. 12 rays are cast from the center of gravity of an object in a clock-like pattern. For a segment of each ray an intensity profile is created. This segment starts at the isoline just below the average intensity of the region and goes until the first maximum isoline (outside the region).

We define the absolute borderness on a segment as

$$absBorderness = (max - min)\sqrt{\frac{max - min}{length_{subray}}}, \tag{4.6}$$

with $min$ and $max$ representing the minimum and maximum iso value of the intensity profile, and $length_{subray}$ being the euclidean distance between the start and endpoint of the ray segment. This definition combines two important aspects that define how strong a border appears overall: the absolute intensity difference and the gradient strength. A border with a high gradient but a small absolute intensity difference might appear as strong as a border with a medium gradient but a higher absolute intensity difference (see also Fig. 4.5a). We use the square root over the gradient to attenuate its otherwise too strong influence in the formula.

The second feature defined per ray is the maximum gradient that can be found on a 7 units long (1 unit = 1 pixel side length) segment along the intensity profile. Fig. 4.5b illustrates this feature.

With those two features we already have a set of 24 feature values per candidate region. Additionally the min, max and quartiles of the 12 feature values per feature type are also provided for the classification.

### 4.1.3 Evaluation and Results

To evaluate the classification performance of our method we performed two tests: First a cross validation over the set of training samples to assess the detection performance for individual embryonic vesicles. Second a validation of the overall classification quality per case (image series) on a dedicated test set.

For the cross validation we used a set of 220 positive and 484 negative samples from 74 different image series. The samples were selected manually by a human expert from the generated segmentation hierarchies. Within a hierarchy covering an embryonic vesicle, the segmentation considered to be best delineating the vesicle was selected as positive sample. All other segmentations in this hierarchy were ignored. Negative samples were selected from segmentation hierarchies which did not cover embryonic vesicles. Due to the large amount of frames and thus the large number of positive and negative instances in an image series the selection of samples was done sparsely. To facilitate the selection of strong examples for the training of the classifier, first a set of positive and negative samples was selected on a few cases. Then the Random Forest was trained with those samples and executed on new cases. On these new cases additional samples were selected from those candidates, which were currently not classified correctly. This procedure was repeated several times until samples from 74 cases were selected.

The cross validation was performed in 74 folds, which we defined in such a way that for each fold all samples from one image series were considered validation data and the rest training data. Furthermore, to evaluate the impact of the newly developed borderness features, we performed this cross validation three times: Once using all features, once using only the borderness features, and once using all but the borderness features. The ROC analysis of the results (Fig. 4.6) shows that using only borderness or no borderness features leads to similarly good results. Using all features, however, improves the overall performance. Especially the boost in specificity we gain on the left side of the ROC is important in our scenario, since having one false positive finding in an overall negative case would already lead to a misclassification. However, one false negative in an overall positive case would most likely be compensated by other positive findings.

Fig. 4.7 shows some typical examples of false negative and false positive detections of embryonic vesicles. The two false negative cases among these examples show a weak contrast and their borders are very faint. This causes the classification weight to lie just below the positive threshold. The two false positive examples have different causes. The first (Fig. 4.7c) is caused by an over-segmentation of the faint darker area, which is split up into three parts. Hence the right candidate region is too small and in combination with the strong borders to the right receives a classification weight slightly above the positive threshold. The second false positive detection (Fig. 4.7d) actually does have some comprehensible similarity to a small embryonic vesicle. However, the fading vessel-like dark structure that arches away to the left of it indicates that this is a different structure (likely a vessel or intestines). This larger contextual information is not captured by the current features. The classification weight is again just a bit above the positive threshold.

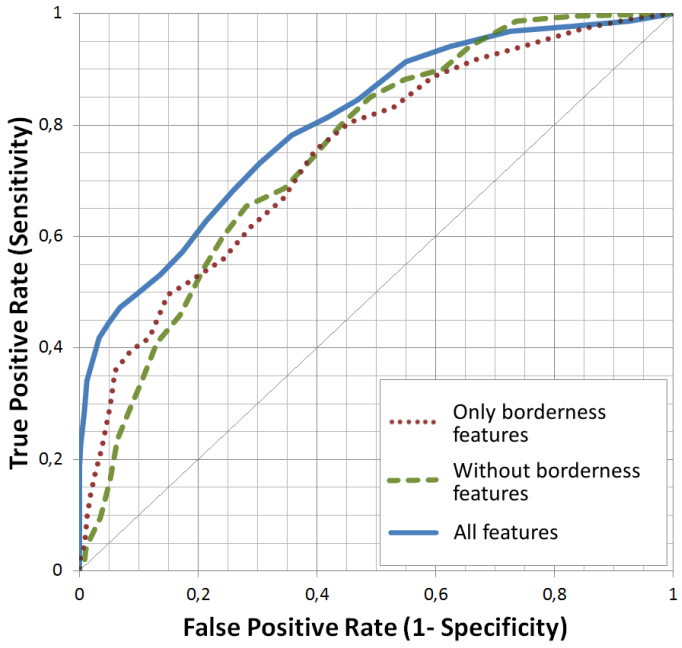
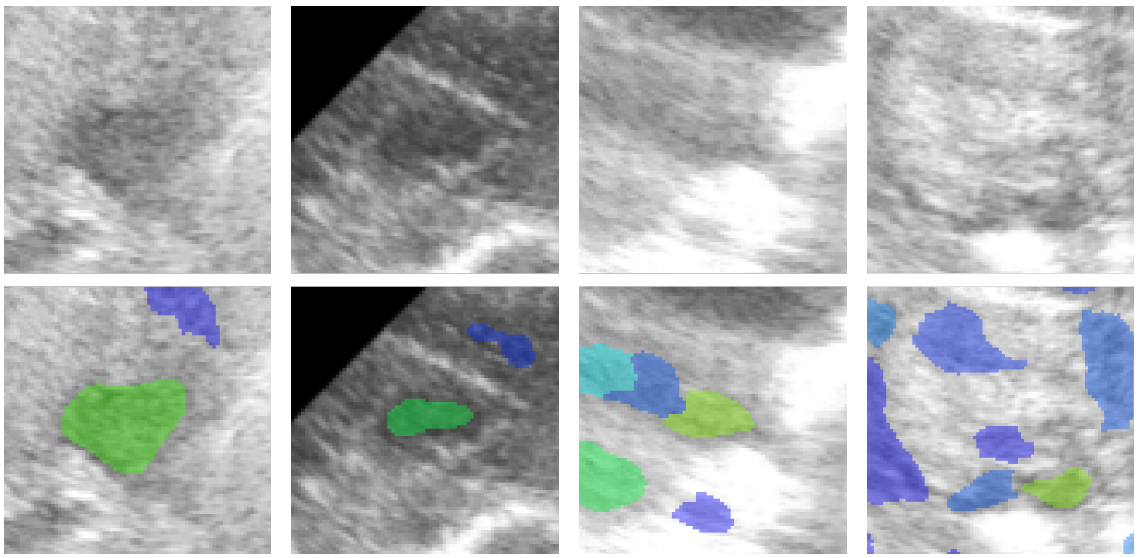



**Figure 4.6:** ROC analysis of a cross validation of the performance of the embryonic vesicle detection on the training sample set. (©2014 IEEE)



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

**Figure 4.7:** Examples of (a), (b) false negative and (c), (d) false positive detections of embryonic vesicles. The top row shows only the image data and the lower row shows the candidate regions that were evaluated as overlays. Colors indicate the classification weight (blue = 0%, green = 50%, red = 100%).

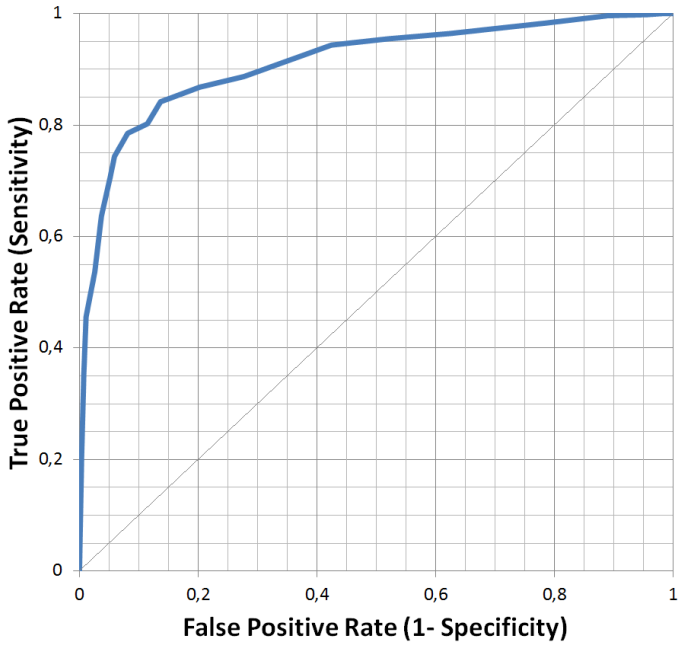



**Figure 4.8:** ROC analysis of the overall classification performance (image series showing indication for pregnancy or not) on the set of 802 dedicated test cases. (©2014 IEEE)

For the evaluation of the overall classification quality we used a dedicated set of 802 cases, which had not been used to select training samples from. Each of these 802 image series was labeled by an expert as either showing indications for a pregnancy or not showing such indications. The ROC analysis in Fig. 4.8 shows a very good detection performance of our algorithm. The best result is gained at a classification weight threshold of 0.65 with a sensitivity/specificity of 84.2%/86.4%.

For 417 of the 802 dedicated test cases an additional label was provided, which indicated if the animal was actually pregnant. This label was determined independently from the automatically acquired ultrasound image series during manual routine examinations and observations. Since non-pregnancy is the exception in commercial pig breeding, most of the negative image series were actually from pregnant animals, but supposedly did not capture embryonic vesicles.

Table 4.1 gives an overview of the three groups among those 417 cases and the number of positive and negative overall classification results from our algorithm. By considering the negatively labeled image series separated by their actual pregnancy status, we can make an interesting observation: In only 6% (2 out of 33) of the the non-pregnant cases our method yields a false positive classification, while on the pregnant cases the false positive rate of our algorithm reaches 13% (15 out of 112). This large difference raises the question, if all those latter results are indeed false positives or if some of the negative labels on the image series are wrong. Unfortunately within the project it was not possible

**Table 4.1:** Evaluation of the overall classification results for the 417 test cases, for which the actual pregnancy status was available.

| Label Image Series | Actual Pregnancy Status | No. Cases | No. Positive Classifications | No. Negative Classifications |
|---|---|---|---|---|
| positive | positive | 272 | 247 | 25 |
| negative | positive | 112 | 15 | 97 |
| negative | negative | 33 | 2 | 31 |

to re-visit the image series in question again with an expert. Our own inspection of the images lead us to believe that some of them are in fact mislabeled, which would mean that our method actually performs a bit better than the numbers indicate.

The single-core computing time on an i7 Q820 1.73 GHz CPU lies between 1 and 2 seconds per image frame. Since the classification in this scenario is not required in real-time this is an acceptable speed.

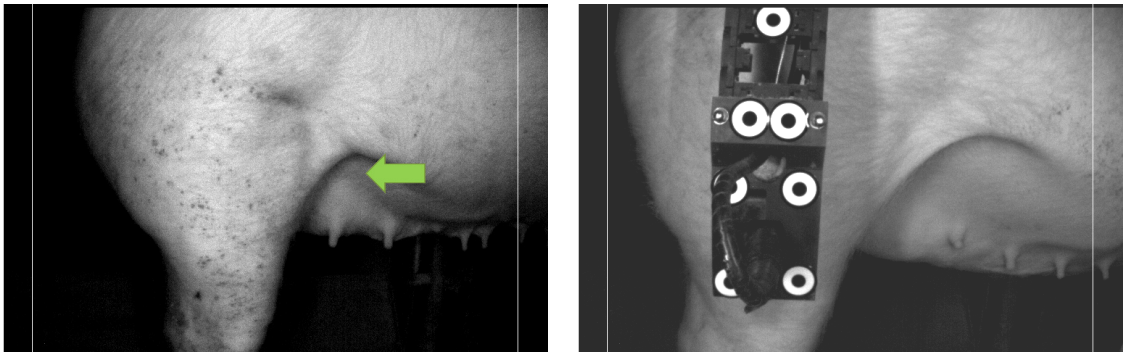
### 4.1.4 Independent External Evaluation

An independent evaluation of our ultrasound-based pregnancy detection was performed by M. Wübbels [173]. The basis for this evaluation was a set of 917 ultrasound series acquired from 120 different pigs. The acquisition was performed fully automatically with BigDutchman's SonoCheck system, which also includes our algorithm for detecting the uterus position and ultrasound arm (see section 4.2).

Of the 917 cases, 533 were labeled as positive (showing pregnancy) and 384 as negative (not showing pregnancy) during a manual inspection of the ultrasound videos. Based on this reference, M. Wübbels reports a sensitivity of 74.22% and a specificity of 91.37% for our algorithm. She does not report, which classification weight threshold was used to determine positive and negative detections, but considering our ROC analysis in Fig. 4.8, her results conform to our own evaluation.

### 4.1.5 Discussion

To our best knowledge, our method is the first solution presented for the fully automatic detection of pregnancy on ultrasound images of pigs. Our approach achieves a high detection quality, sufficient to be implemented in practice. Even compared to manual ultrasound examinations done by human experts, who reach sensitivity/specificity rates of up to 96.5%/96.0%, our results are particularly good, considering that, in contrast to the automatic image acquisition, a human is able to reposition the probe as necessary to get a better view. An independent evaluation of our method integrated into the full BigDutchman SonoCheck system and performed under real-world conditions confirmed the results of our own evaluation.



(a) (b)

**Figure 4.9:** Example camera images of a pig standing in the right position in the feeding station. (a) The uterus position is marked with a green arrow. (b) The ultrasound arm with attached markers is visible.

This section presents two main methodological contributions. First, the strategy of using a classifier to find the correct segmentation within multiple segmentation hierarchies, which are designed in such a way that the correct segmentation is almost always contained. This strategy can be transferred to many region-based classification tasks. The second contribution is the set of borderness features, which are able to capture the perceived strength of a border, even though the borders are particularly fuzzy in this image domain.

Future work would be to improve the computing time, which is acceptable in the context of the application, but high considering the small size of an image frame. The critical factor is that with the segmentation hierarchies a large number of candidate regions is generated, for which rather expensive region-based features have to be computed. A promising approach would be to implement a pre-classification which uses a smaller set of features to limit the candidate regions to the most likely positive ones.

## 4.2 Detection of Uterus Position and Ultrasound Arm on Video Images

To fully automatize the pregnancy detection it is not enough to only automatically analyze the ultrasound images. Also the acquisition of the ultrasound images has to be performed automatically. For this purpose the feeding station has a camera mounted on the side behind a hatch. The position is selected in such a way that it captures the flank of the animal around the uterus position, if the animal stands in a position where it can feed. The images of the camera are taken under infrared light to eliminate the influence of normal light sources. Figure 4.9 shows example camera images.

The image analysis task that arises in this setup is to detect the uterus position in the video image so that the ultrasound arm can be directed correctly. Furthermore, the

construction of the movable ultrasound arm does not allow to track the position of the arm and ultrasound probe precisely enough. Hence to know where the arm is and consequently how to move it to the correct position, also the ultrasound arm must be detected on the video image. To facilitate the detection of the arm and to be able to distinguish the location of the mounted ultrasound probe on the arm, several markers in a specific configuration are attached on the arm. On the probe mount three rows of double markers (left, right) are applied while on the arm only single markers in a regular distance are applied. See Fig. 4.9b for an example of the marker configuration in the ultrasound arm.

We developed a method that automatically detects the uterus position as well as the ultrasound arm marker positions in such video images.

### 4.2.1 Related Work

Regarding related work, the general field the task at hand is dealing with is computer vision with the focus on detecting and recognizing objects in real world images and videos. Szeliski [158] gives an introduction into the topic. He shows that it is a quite popular and big field with various approaches. He also recognizes it as a particularly hard task and mentions that most success has been made in face detection and recognition. Szeliski indicates that Yang et al. [175] give a comprehensive overview of earlier work on face detection, to which we would like to refer for further reading on this particular topic. They differentiate four categories of approaches for face detection: knowledge-based, feature invariant, template matching, and appearance-based. The general advantages and disadvantages of these different types of methods are discussed in their paper. However, they are not able to deduct a final conclusion regarding which type of approach seems most promising.

One of the most famous works worth mentioning on face detection is the one of Viola and Jones [167] who propose a classifier boosting algorithm and Haar-like features. They also successfully extended their approach to pedestrian detection in video images [168], which is another hot topic according to Szeliski [158]. Extensive surveys on the topic by Enzweiler and Gavrila [34], Geronimo et al. [41], and an older one by Moeslund and Granum [92] show the dimensions and different directions of the research. Enzweiler and Gavrila [34] conclude that the best approaches are those using as features the histogram of oriented gradients (HOG) [30] or for faster computation the scale-invariant feature transform (SIFT) [78] in combination with machine learning. However, for real-world applications they as well as Geronimo et al. [41] argue that more improvement is still required.

Hu et al. [59] as well as Yilmaz et al. [177] conducted surveys with a more widened focus on object detection in general, while human detection still seems to be the most prevalent. In general they do not introduce significantly different approaches than the other mentioned surveys. Yilmaz et al. [177] mention that “Several robust trackers have been developed which can track objects in real time in simple scenarios”, but “the assumptions used to make the tracking problem tractable [...] are violated in many realistic scenarios and therefore limit a tracker’s usefulness”.

We can see there is a large variety of approaches when it comes to tracking of structures in video images. It seems the more successful ones tend to use machine learning algorithms and sophisticated features. However, in a more constrained setting other approaches might work just as well. Yang et al. [175] argue that template matching approaches are easy to implement but "cannot effectively deal with variation in scale, pose, and shape". Furthermore they claim that knowledge-based approaches, in which expert knowledge is captured in explicit rules, are difficult to establish in complex scenarios. However, in our setting we are dealing with very controlled conditions and the above mentioned variations are limited. A combined approach of template matching and knowledge-based processing seems feasible and rather easy to implement.

We did not encounter specific literature on the automatic detection of body positions on animals under similar conditions as the task at hand. Hence, to our best knowledge, this is the first method presented for detecting the uterus position on pigs in video images.

### 4.2.2 Methods

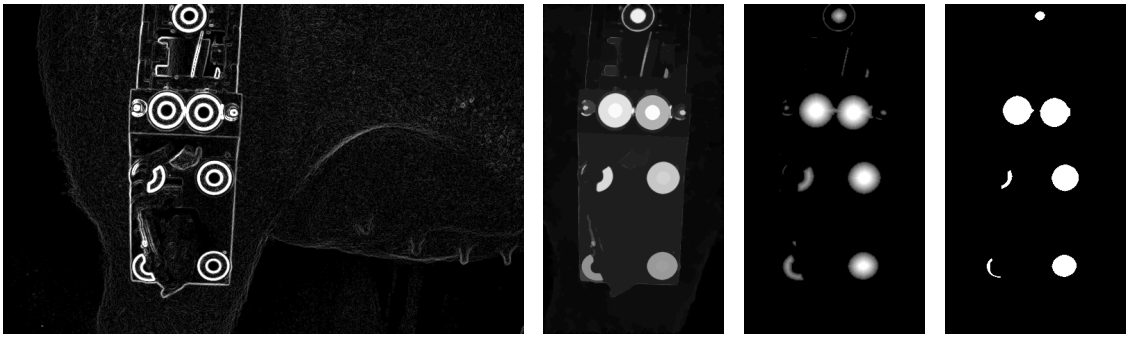
The first stage of processing is the detection of ultrasound arm markers, followed by the uterus detection, which uses the results of the ultrasound marker detection, if available.

#### 4.2.2.1 Ultrasound Arm Marker Detection

The markers on the ultrasound arm are designed as a black circle/dot surrounded by a white circle. The ultrasound arm and attached ultrasound probe and cables are designed to be all black. This gives the markers a good contrast.

The first step towards detecting the markers is to calculate the Gauss gradient on the input image with $\sigma = 0.6$ pixel. Due to the strong contrast and clear edges of the markers we get a strong signal along their borders which is significantly stronger than almost any other gradient signal in the image (see Fig. 4.10a). This helps separating them. Further processing is limited to a region of interest (ROI) around the highest gradients. The threshold for this is set to a fixed value of 10, since the gradients for anything else but the ultrasound arm are usually below 5 while the ones around the markers reach levels above 20.

In the ROI on the gradient image a constrained connection cost filter [154, 118] followed by a morphological erosion with a $3 \times 3$ pixel kernel is applied to create segmentations of the markers (see Fig. 4.10b). To refine these and cut possible connections we threshold the current image at 5 into a binary image and apply a euclidean distance transformation to the result (see Fig. 4.10c). Cutting out everything below a threshold of 5 removes small and thin objects, and cuts small connections. What remains are a few individual components that represent (most of) the markers and possibly a few other structures. To sort out the markers among these, we transfer the remaining components into an OBIA representation and classify them by shape and size into markers and non-markers using



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

**Figure 4.10:** Image processing stages of the ultrasound arm marker detection: (a) Gauss gradient; (b) Constrained connection cost + morphological erosion; (c) Euclidean distance transformation; (d) Resulting objects for OBIA analysis.

the following rule for objects $o \in O$:

$$o.circularity > 0.6 \ \wedge \ o.ellipticity > 0.9 \ \wedge \ o.size > 100\text{px} \tag{4.7}$$

We include the ellipticity additionally since due to the fish-eye effect of the camera lens the circular markers might actually appear elliptic on the image.

#### 4.2.2.2 Uterus Detection

According to Knox et al. [65] the correct placement of the ultrasound probe should be done so that "The surface of the transducer is [...] applied to contact the abdomen just lateral to nipple line and ahead of the rear leg". The green arrow in Fig. 4.9a indicates the correct position. The most stable landmark for that position is the skin fold of the leg, which under the direct camera light also casts a prominent shadow along its border. Our target point lies just below the maximum of the curve that is defined by that skin fold.

In order to detect the skin fold we first go for another landmark: the rear leg, because relative to that we can define a search area for the skin fold, eliminating large parts of the image from being processed at all. To find the leg we take the lower half of the image and apply a fixed threshold of 18 which is sufficient to coarsely separate the pig from the black background. We then perform a connected components analysis [153] on the background pixels and use OBIA to select the leftmost segment, which touches the bottom image border, but does not touch the left image border. The leftmost pixel position of the selected object, shifted by 50 to the left, is considered as leg position. The ROI for further processing is defined as a $500 \times 400$ pixel box, spanning from the right of the leg position and the bottom of the image (see Fig. 4.11a). For all further processing the ROI is down-sampled to $250 \times 200$ pixel.

In the ROI we first apply the line detection filter of Sato et al. [131], which we set to detect dark curvilinear structures at $\sigma = 5.0$px. In Fig. 4.11b we can see that among other

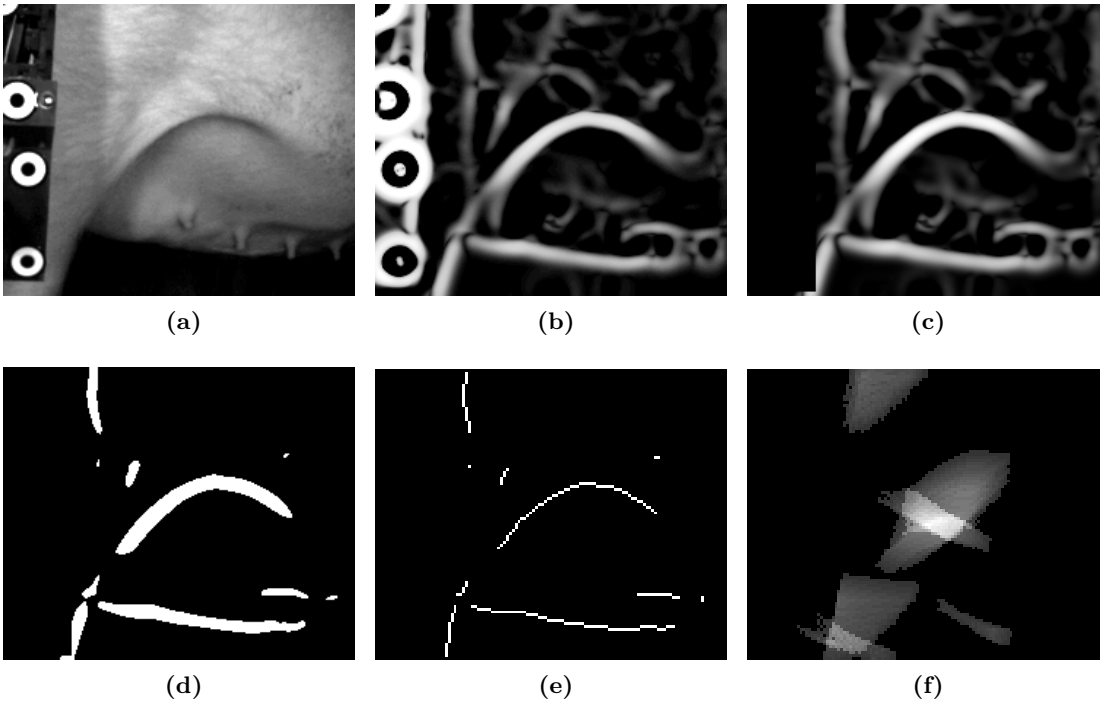



**Figure 4.11:** Image processing stages of the skin fold/uterus detection: (a) ROI; (b) Line detection; (c) Removed ultrasound arm area; (d) Binary image after thresholding; (e) Skeletonized connected components; (f) Curve Shape Filter output.

structures this highlights the skin fold very well. In case the ultrasound arm is present in the image, we use its ROI coordinates to remove those areas from the current image stage (see Fig. 4.11c). A low threshold is now applied to generate a binary image with weak borders removed (Fig. 4.11d), which is then skeletonized using the method of Selle et al. [142] (see Fig. 4.11e).

What is left to do now, is to decide, which of the remaining skeletonized components represents the skin fold. The main distinguishing property is its particular shape that looks like a smooth curve. Depending on the standing position of the pig this curve might be more or less stretched. We therefore have to account for these variations. For this we developed a dedicated kernel-based filter.

Based on hundreds of sample images we created a model that covers all typical variations of skin fold curves. As illustrated in Fig. 4.12, the filter kernel is designed as a thick arch which is sectioned into several bins (different from the illustration the actual kernel we used is sectioned into 58 very thin bins). For each filter position in the image we determine which connected component covers most of the bins. The output of the filter at the current position is then the number of bins covered by this connected component. The center position of this Curve Shape Filter is placed so that it matches the target position for the ultrasound probe below the skin fold.

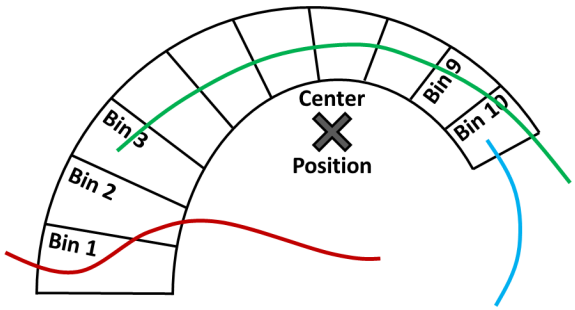



**Figure 4.12:** Schematic illustration of the Curve Shape Filter kernel. The colored curvilinear structures represent skeletonized connected components. The structure that covers most bins is the green one, hence the output of the filer in this example would be 8.

The result of applying this filter to the skeletonized connected component image is shown in Fig. 4.11f. The uterus position is determined by considering the position of the highest Curve Shape Filter output value. If several pixels share this maximum value, the center of gravity of those pixel positions is considered as the uterus position.

### 4.2.3 Evaluation and Results

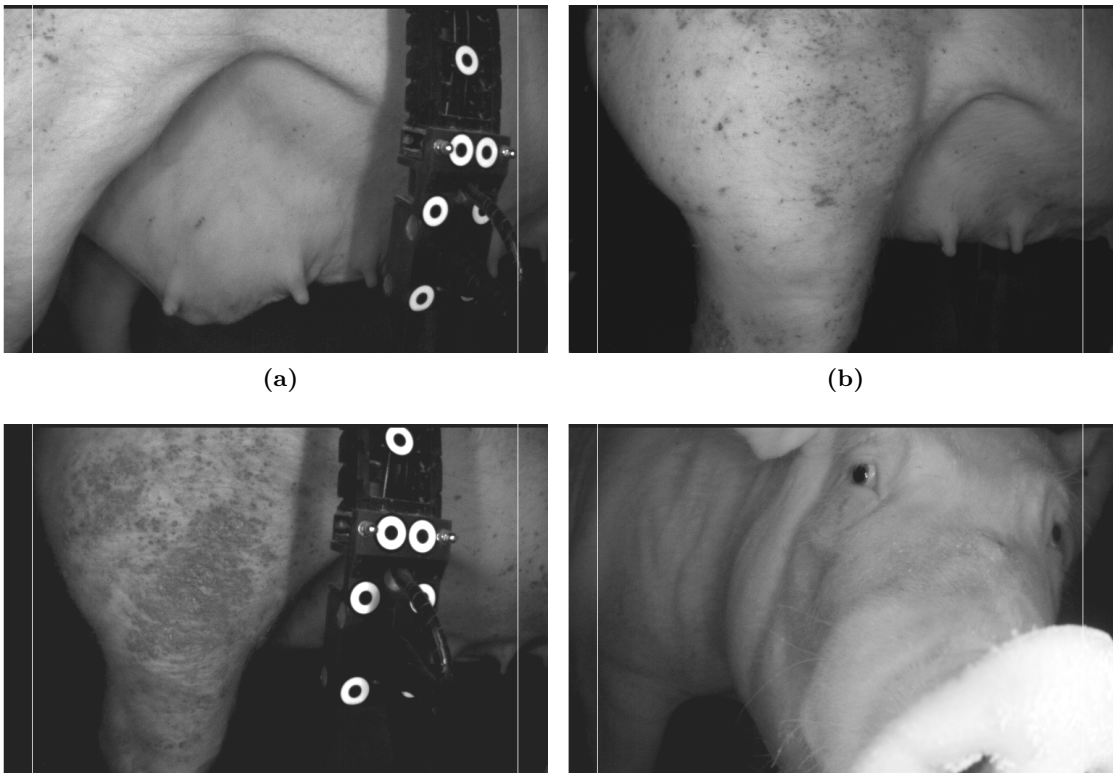
For the evaluation we used 200 dedicated test images, which were selected from more than 30,000 sample images from more than 60 different pigs. The 200 test images were grouped into four categories:

- 50 images showing the ultrasound arm and the uterus
- 50 images showing only the uterus
- 50 images showing only the ultrasound arm
- 50 images showing neither the ultrasound arm nor the uterus

Fig. 4.13 shows a sample image from each of these categories. All 200 test images were annotated with reference labels to compare the detection results against. The reference labels mark a circular area, which defines the detected positions that would be accepted as a hit. This circular reference area is subdivided into an inner circle defining an "optimal hit" area and an outer circle defining a "good hit" area. Any detected position outside a reference circle would be considered a miss/fail. The reference circles for the ultrasound arm markers have a radius of 5 pixels for the "optimal" area and an additional 5 pixels radius for the "good" area. For the uterus position the "optimal" reference circle has a radius of 20 pixels and an additional radius of 20 pixels for the "good" area. See also Fig. 4.14 for in illustration of the reference label areas.

#### 4.2.3.1 Evaluation of Ultrasound Arm Marker Detection

On the 100 test images showing ultrasound arm markers a total of 418 reference labels for the ultrasound arm markers were annotated. The number of annotated visible markers per image ranged from 2–6. We evaluated the detection algorithm in terms of sensitivity



**(a)** **(b)**

**(c)** **(d)**

**Figure 4.13:** Samples from the four categories of test images: (a) Uterus and ultrasound arm visible; (b) Only uterus visible; (c) Only ultrasound arm visible; (d) Neither ultrasound arm nor uterus visible.

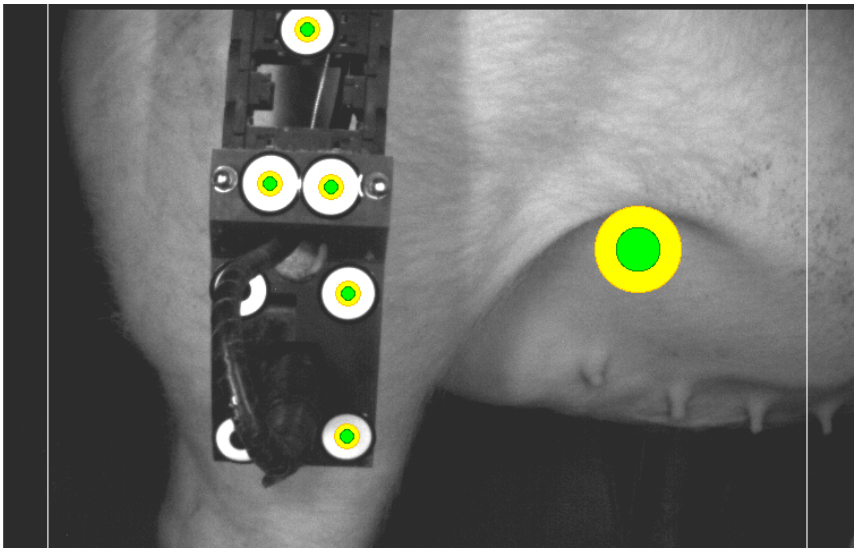


**Figure 4.14:** Illustration of the reference labels for evaluation. Small circle reference labels indicate correct ultrasound arm markers. The large circle reference label indicates the correct uterus position. The colors indicate areas for an optimal (green) and good (yellow) hit. Everything outside the reference is considered a miss/fail.

(true positive rate) and precision. Furthermore we calculated the rate of images in which no marker was found at all as well as the rate of images in which false markers were found. The results of the evaluation are summarized in table 4.2.

Our algorithm reaches an overall sensitivity of 94% and an overall precision of 99%. Hence in most cases all markers are being detected correctly. Additionally, almost all correct detections hit the optimal reference area. Furthermore, considering the NMI Rate we can see that no case occurred in which the ultrasound arm was present but was not detected at all. Even if some markers remained undetected, at least one was always found. False positive detections occurred in only three cases. One was caused by a reflection of the camera light on the ultrasound arm, the other two detected the eye of the pig as a marker, when the animal was curiously looking through the opened hatch at the camera (see Fig. 4.13d for an example of this situation).

#### 4.2.3.2 Evaluation of Uterus Detection

We evaluated the uterus position detection algorithm in terms of precision and sensitivity (true positive rate). We also determined the rate of images in which no uterus was detected as well as the rate of images in which a false uterus position was detected. The results of the evaluation are shown in table 4.3.

Considering hits in the good and optimal reference areas together over all images, our algorithm reaches a sensitivity of 89% as well as a precision of 89%. If we consider only those images in which the uterus is actually visible, the precision even reaches 99%, since

**Table 4.2:** Results of the ultrasound arm marker detection evaluation. True positive (TP) rates and precision are calculated with respect to the overall number of reference markers/detections per image category. NMI and FMI rates are calculated with respect to the number of images per category: NMI Rate = Rate of images with no ultrasound marker detected; FMI Rate = Rate of Images with one or more false ultrasound marker detections.

| Image Category | Optimal TP Rate | Good TP Rate | Combined TP Rate | Combined Precision | NMI Rate | FMI Rate |
|---|---|---|---|---|---|---|
| Uterus & arm markers | 0.92 | 0.01 | 0.92 | 0.99 | 0.00 | 0.02 |
| Only arm markers | 0.95 | 0.00 | 0.95 | 1.00 | 0.00 | 0.00 |
| Only uterus | — | — | — | — | 1.00 | 0.00 |
| Nothing | — | — | — | 0.00 | 0.96 | 0.04 |
| Overall | 0.94 | 0.00 | 0.94 | 0.99 | — | 0.02 |

**Table 4.3:** Results of the uterus detection evaluation. NUP Rate = Rate of images in which no uterus was detected; FUP Rate = Rate of images in which a false uterus position was detected.

| Image Category | Optimal TP Rate | Good TP Rate | Combined TP Rate | Combined Precision | NUP Rate | FUP Rate |
|---|---|---|---|---|---|---|
| Uterus & arm markers | 0.66 | 0.14 | 0.80 | 1.00 | 0.20 | 0.00 |
| Only uterus | 0.76 | 0.22 | 0.98 | 0.98 | 0.00 | 0.02 |
| Only arm markers | — | — | — | 0.00 | 0.98 | 0.02 |
| Nothing | — | — | — | 0.00 | 0.82 | 0.18 |
| Overall | 0.71 | 0.18 | 0.89 | 0.89 | — | 0.06 |

almost no false detections occur. However, especially in those images in which the pig is not standing in the right position and neither the ultrasound arm markers nor the uterus is visible, the number of false uterus position detections goes up to 18%. The most typical error is caused by a skin fold at the neck of the animal, when it is standing back in the feeding station. This skin fold looks very similar and is placed right next to the front leg. Since the rear leg is not visible in those cases, the algorithm mistakes the front leg for the rear. Then the derived ROI captures the neck skin fold which is detected as the uterus due to its similarity in shape.

The sensitivity on images showing only the uterus is very high with 98%. Difficulties occur when the ultrasound arm is present as well. Especially when it is quite close to the uterus position, the shadow of the arm — or worse the arm itself — interferes with curvilinear shape of the skin fold. Nevertheless, we still reach a sensitivity of 80%.

#### 4.2.3.3 Evaluation of Computation Time Performance

As hardware for the evaluation of the computation time performance we used a Dell Precision M3800 laptop computer with an SSD hard drive, 16GB memory and an Intel Core i7-4712HQ CPU at 2.30GHz. All computations were executed on a single core.

The average computation time per frame is 0.36 seconds. In this particular setup this was sufficient, because the camera feed supplies the input images with 2.5 frames per second. We therefore did not put extra effort into optimizing the computational performance. Hence the processing pipeline can most likely be optimized. Further improvement could be reached by parallelizing computationally expensive operations like the Curve Shape Filter.

### 4.2.4 Independent External Evaluation

Part of the evaluation of M. Wübbels [173] (see also section 4.1.4) also covered the automatic acquisition of the ultrasound video images. Our uterus and ultrasound arm marker detection software was linked to a control system that automatically interprets the detected positions and steers the ultrasound arm towards the uterus position and then engages the probe. Details about the implementation of the control system are not known to us.

M. Wübbels reports that among 120 different pigs 984 acquisition attempts were performed. 772 (78.5%) of those resulted in a successful recording of an ultrasound video at the correct position. For 100 (10.2%) cases the ultrasound probe did not attach properly to the skin and recored only air. For another 45 (4.6%) of the cases the probe was attached to the wrong position on the animal. The reasons for the failures are not reported. It could of course be caused by a wrong output of our detection algorithm but could also be a mistake of the control system. Furthermore it is also likely that in some cases the animal moved upon contact and displaced the probe, or stood too far on the opposite side of the feeding station such that the probe could not properly attach to the skin.

In the remaining 67 (6.8%) cases the ultrasound probe was not engaged and thus no

image data was acquired. It was reported that in 4 of those cases the animal left the feeding station. For the remaining 63 the control system ran into a time-out before it would engage the ultrasound probe.

Considering that several factors besides an erroneous output of our position detection algorithm could lead to a failed acquisition attempt, a rate of 78.5% successful acquisitions indicates that our methods work robustly in a real world setup.

### 4.2.5 Discussion

The goal of our efforts was to detect the uterus position and markers on the ultrasound arm well enough, to facilitate automatic steering of the ultrasound arm by an external control system, linked to our software. As the evaluation shows, the markers on the ultrasound arm are detected with a high sensitivity and an almost perfect precision. We can therefore conclude that our algorithm will give sufficient feedback to a control system to facilitate steering the ultrasound arm to any location captured in the video feed. This conclusion is also supported by an independent evaluation of the whole system.

Sensitivity and precision for the uterus detection is reasonably high with 89% each. Considering that the system analyzes a constant video feed, a missed detection in one frame will most likely be compensated by correct detections in the following frames. The same holds for false detections. Especially if the detected uterus position "jumps" around from frame to frame it is not safe to engage the ultrasound probe anyway, because the instability in the uterus position is either caused by false detections or by the animal moving around too much. Since the steering control system would need to check the stability of the uterus position over many frames anyway, the detection quality for the uterus can be considered sufficient for practical purposes.

It should also be noted that the most problematic cases that produce false positive detections are the least common to actually happen in practice. This is when the animal moves back or even starts looking through the camera hatch. However, pigs get used to the new system and the touch of the ultrasound probe in particular, lose their fear and curiosity, and just keep feeding and thus staying in the correct position. Nevertheless, an idea for future work would be to detect these wrong positions by detecting the head of the pig or checking if the nipple line reaches until the left side of the image, which could not happen if the pig stands in the correct position.

Considering the success and attention that detection algorithms using HOG [30] or SIFT [78] features (and enhancements of them) receive, it could be a worthwhile future work to test such an approach for the uterus detection, even though our current solution reaches already results that make it applicable in the designated setting.

## 4.3 Detection of Dirt on Camera Pane

Another small image processing task on the video image was to automatically detect dirt on the camera pane. Since the system will be permanently installed in animal stables, it is likely that the camera's field of view will be occluded by dirt with time. The camera lens itself is protected behind a plexiglass pane but dirt on this pane has to be removed from time to time to get a clear image of the scene in the feeding station. Therefore, a special detection mode for estimating the amount of dirt on the plexiglass pane in front of the camera was supposed to be developed as well.

To be able to detect if something in the camera image is actually dirt, we require some kind of reference to compare to. The inside of the feeding station is too dark and irregular. Furthermore the inside will get dirty much faster than the outside. However, the hole through which the camera observes the inside of the feeding station can be covered with a hatch, which is remotely controlled. The metal of the hatch itself creates too many refractions to serve as reference, therefore it was covered with a checkerboard pattern (see Fig. 4.15a)

With this well defined pattern as reference, the goal was to develop a detection algorithm that can estimate the level of dirt in the camera's field of view.

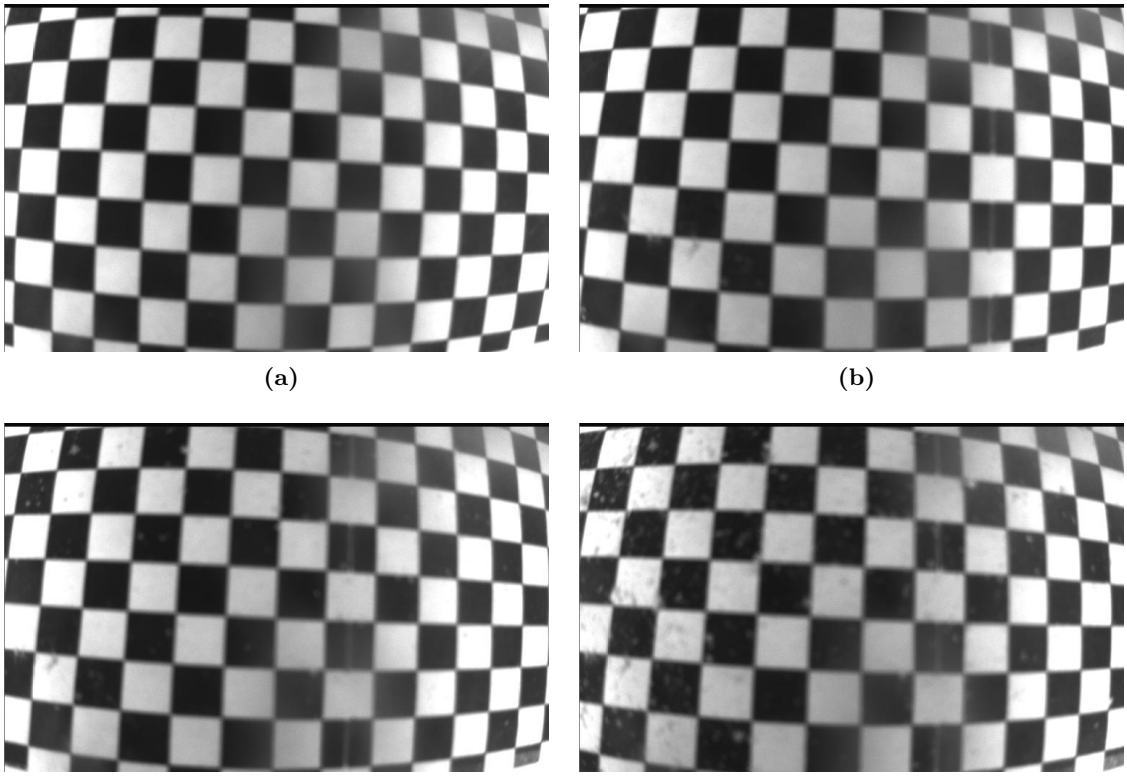
### 4.3.1 Related Work

The task of detecting dirt on a surface falls roughly into the category of material property analysis and detection of defects and irregularities. Quite some work has been done on computer-assisted visual analysis of textile fabrics. Abouelela et al. [1] present a method for localizing structural defects in fabrics. Like in our setting they also use infra-red lighting and an infra-red camera to control the lighting environment. Their image processing is rather simple, using a zero-mean image, median and variance filtering as well as fixed thresholds to detect areas of high variance.

Kumar [68] performed a survey on fabric defect detection. They report that the majority of methods follow statistical approaches. The underlying assumption for these approaches is that defect-free regions cover large areas and are rather invariant in their visual appearance as oppsoed to the areas with defects. Another group of approaches are spectral methods, utilizing for example the Fourier transform or a wavelet transform. Furthermore Kumar describes model-based approaches which attempt to model the expected texture as a stochastic or deterministic model.

A review on inspection systems for agricultural and food products was published by Brosnan and Sun [15]. However, they do not really compare different types of methodological approaches but simply report on existing ones for different types of products. They conclude that more complex systems are required for fresh products due to the higher variability in their appearance. Simpler approaches can be successful for processed foods due to the standardized production procedures.

A broader perspective on detection of irregularities is taken by Boiman and Irani [11].



**(a)** **(b)**

**(c)** **(d)**

**Figure 4.15:** Examples images of the checkerboard test panel with (a) no dirt, (b) light dirt level, (c) medium dirt level, (d) strong dirt level.

Their approach is not limited to a specific domain but also extends to detecting suspicious (human) behavior in images as well as salient pattern detection in general. They present a very interesting approach which relies on a database of "approved" pictures. A new image is now analyzed by trying to match patches from the images in the database to the test image. If some area in the test image can not be composed of matching patches from the database it is considered an irregularity. Our approach is a little related to this with respect to the idea of a patch-based processing. However, in our approach the patches are explicitly defined by the checkerboard pattern. Furthermore our method is not comparing against a pre-established database but allows an automatic configuration to establish a reference for the individual system.

Xie [174] performed a review of surface defect detection methods that rely on texture analysis. His survey is not limited to a specific type of material. He differentiates the approaches into statistical, structural, filter-based and model-based methods. He also concludes that statistical approaches are among the most widely used ones.

A work on actual dirt detection was published by Mertens et al. [88]. They presented a method for automatically detecting dirt on eggs. They used a rather simple approach

that performs a thresholding based on a histogram analysis on different color channels.

With respect to the findings in the related work, especially the different surveys, complex and advanced methods like structural, spectral or model-based seem over the top for our purposes. First, our reference checkerboard pattern does not possess any particular texture within the single checkerboard cells. Second, we are not interested in locating dirt but only in assessing if dirt is present or not. A statistical approach as the basic idea seems appropriate for our purposes.

### 4.3.2 Methods

The idea for our algorithm is simple: Count the number of clean squares on the checkerboard. If the number is significantly lower than for a defined clean checkerboard, then the dirt level is considered too high. The algorithm will be initialized with a clean checkerboard for reference.

The exact alignment of the checkerboard pattern is not 100% stable. It may vary a bit after opening and closing the hatch. Therefore we cannot base our detection on the assumption that we have exactly the same alignment of the checkerboard pattern, but have to account for those variations.

Our approach is therefore to first detect black and white square elements in the current image. Then estimate the amount of dirt in each individual square element and consider all squares which have a dirt level below a certain threshold as clean. Now use the difference between the number of clean squares in the reference and the number of clean squares in the tested image as indicator for the overall dirt level. The reference number of clean squares as well as the dirt threshold are determined in a calibration step.

#### 4.3.2.1 Square Detection

We start the square detection by running a Gauss gradient magnitude filter on the input image with $\sigma = 1$ pixel. This is followed by a morphological closing operation with a $3 \times 3$ kernel to close holes which appear at junctions of 4 squares (see Fig. 4.16a).

Then again a Gauss gradient filter is applied ($\sigma = 1$ pixel), which creates a double border along the contact lines of the squares (see Fig. 4.16b). This is required to prevent the subsequent watershed segmentation from leaking along the single border of the first processing step. Performing a watershed segmentation on the double border image results in an over-segmentation that captures the squares nicely while keeping them separate from the connection areas (see Fig. 4.16c). This over-segmentation defines the objects $O$ for an OBIA analysis. Deciding which objects are considered to be square elements for the subsequent analysis is done by selecting all $o \in O$ for wich

$$o.rectFit_1 > 0.7 \tag{4.8}$$

$$o.size > 400 \text{ pixel} \tag{4.9}$$

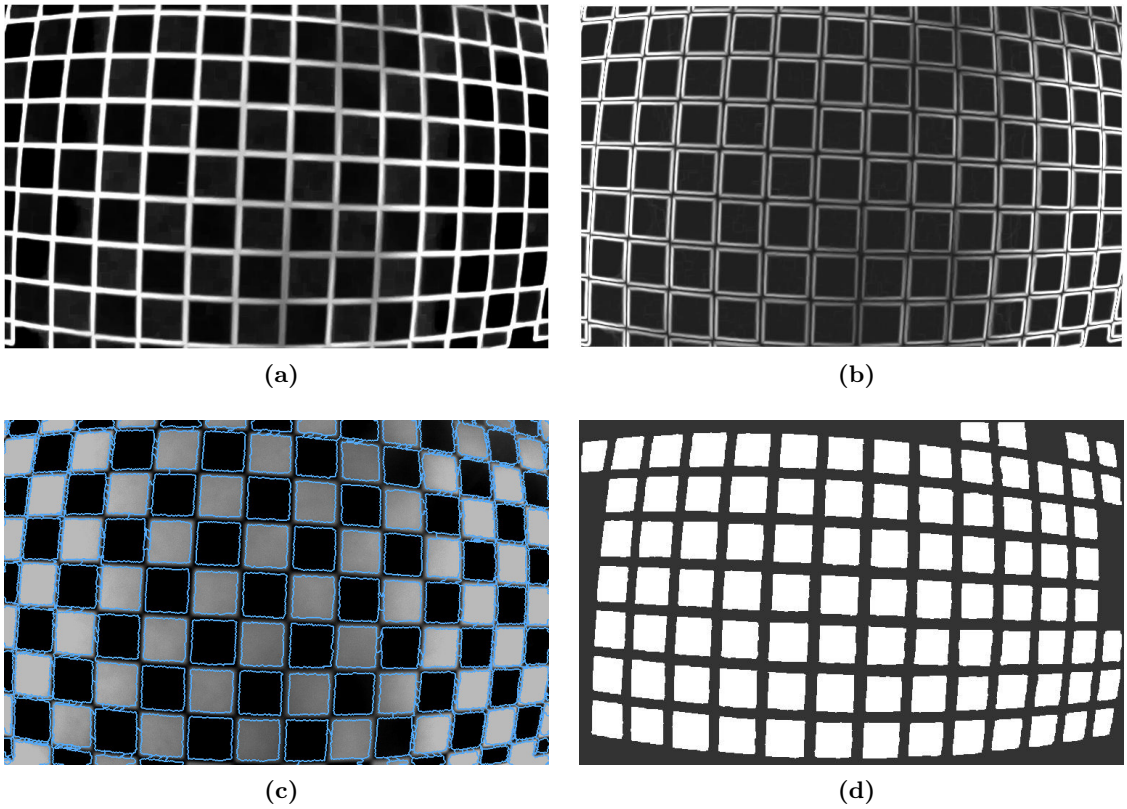


**Figure 4.16:** Exemplary intermediate results of the essential steps of the square detection on a clean input image: (a) Step 1: detect borders between squares; (b) Step 2: create double borders; (c) Step 3: perform watershed segmentation; (d) Step 4: select proper rectangles and shrink them.

We use the $rectFit_1$ instead of $squareFit_1$ or $squareFit_2$ because the fish-eye effect causes some squares on the checkerboard to appear rather rectangular on the captured image. For the purpose of the dirt analysis within the square elements, all detected objects are slightly shrunk by a morphological erosion (see Fig. 4.16d).

In case of strong dirt in an image the detection of some squares might of course fail, because the gradients along the square element's borders would be disturbed. However, this is not a problem at all: If they are not even considered as square objects in the first place, they will also not be counted as clean squares.

#### 4.3.2.2 Dirt Indicator per Square

The dirt estimation is based on the gradient image from step 1 of the square detection. Under clean conditions, the inside of all square elements should be quite homogeneous. If dirt is present, this homogeneity would be disturbed, which would be reflected as peaks in the gradient image (see Fig. 4.15 for examples of different dirt levels). Hence for each

square element detected in the previous step we consider the maximum gradient inside of it as indicator for the dirt level of the particular element. To avoid catching the expected strong gradients along the border between squares a margin around the analysis areas is required. Hence the shrinking of the final objects at the end of the square detection (see section 4.3.2.1). Determining the maximum gradient for each square object is a simple feature extraction in OBIA. To be able to decide until which threshold of the maximum gradient a square element is considered to be clean we require a reference calibration.

#### 4.3.2.3 Reference Calibration

The reference calibration is performed under a manually confirmed clean setup. We run the square detection on 3 frames and save the median number of white ($n^R_{white}$) and black reference squares ($n^R_{black}$). As dirt indicator threshold we consider the 90th percentile of the maximum gradients for white ($t^R_{white}$) and black ($t^R_{black}$) squares. As before, $t^R_{white}$ and $t^R_{black}$ are saved as the median of the three available 90th percentiles.

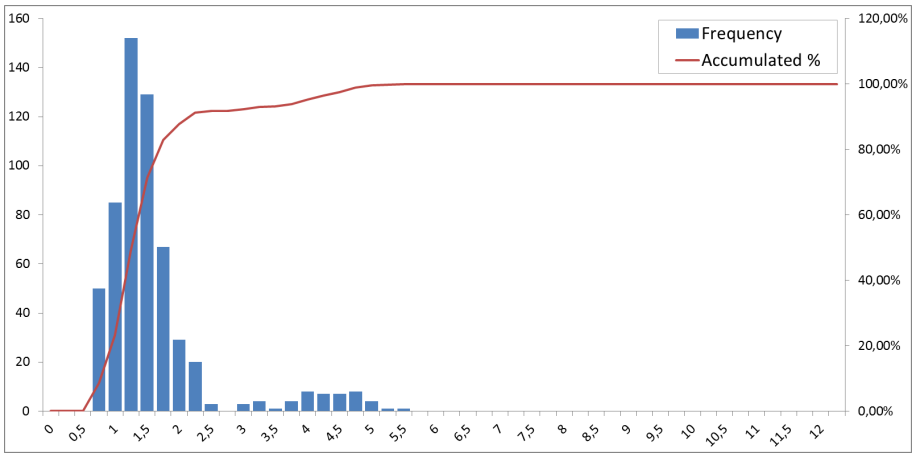
The reasoning behind using the 90th percentile is based on an analysis of the available test images (for details on the image data see section 4.3.3). We analyzed the distribution of maximum gradients among the square objects for three groups of images: One group considered clean or showing a light dirt level, one group considered showing a medium dirt level, and one group considered showing a strong dirt level. Fig. 4.17 shows the analysis for black squares and Fig. 4.18 for white squares.

For the black squares we can see that the distribution shifts more and more towards the higher gradients for higher dirt levels. For the white squares the distributions of clean/light and medium dirt levels are very similar with just slight tendency towards the higher gradients in the medium cases. We can also see in both histograms of clean/light samples that a few outliers with high gradients appear. Using the maximum gradient as reference threshold is therefore no reliable option. The 90th percentile however captures the fall-off at the right side of the main accumulation very well.

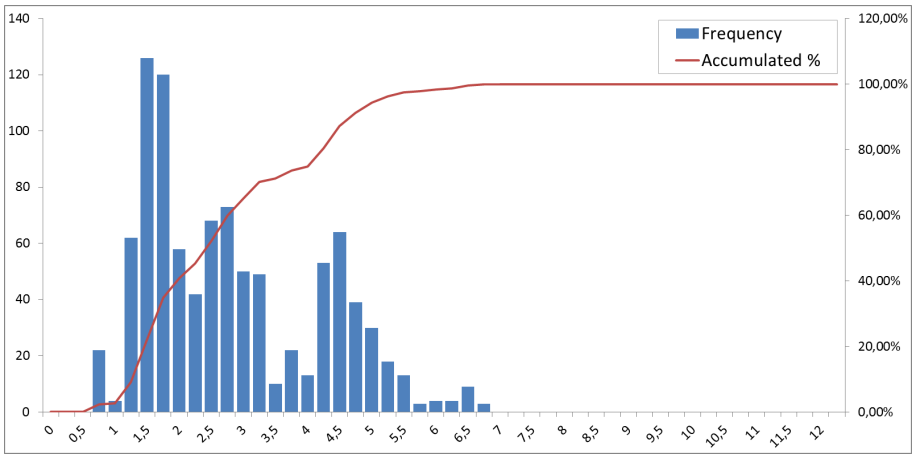
#### 4.3.2.4 Overall Dirt Estimation

The dirt estimation is performed by running the square detection on 3 frames and considering the median number of detected white ($n_{white}$) and black ($n_{black}$) squares. Furthermore for each frame the number of white and black squares with a maximum gradient above the respective thresholds $t^R_{white}$ and $t^R_{black}$ are counted and the median count is considered as $n^{>t}_{white}$ for the white median and $n^{>t}_{black}$ for the black median count. The overall indicator for the dirt level is calculated as

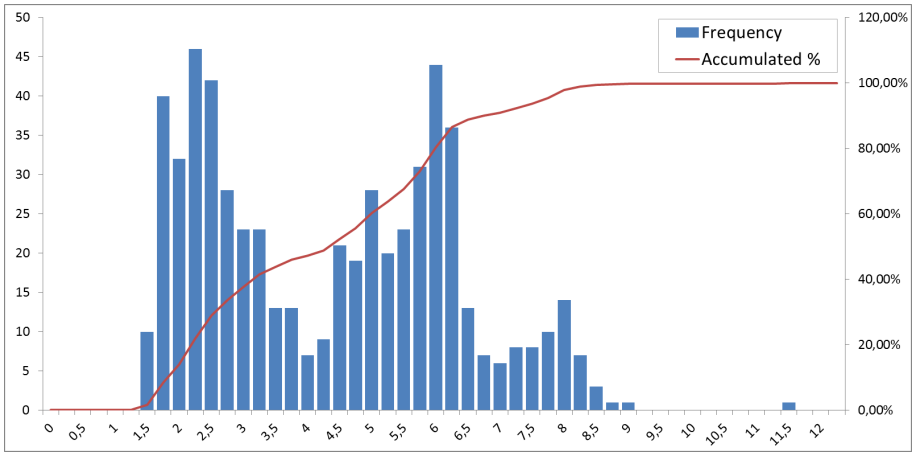
$$\Delta = \Delta_{white} + \Delta_{black} + \Delta^{>t}_{white} + \Delta^{>t}_{black} \tag{4.10}$$




**(a)**

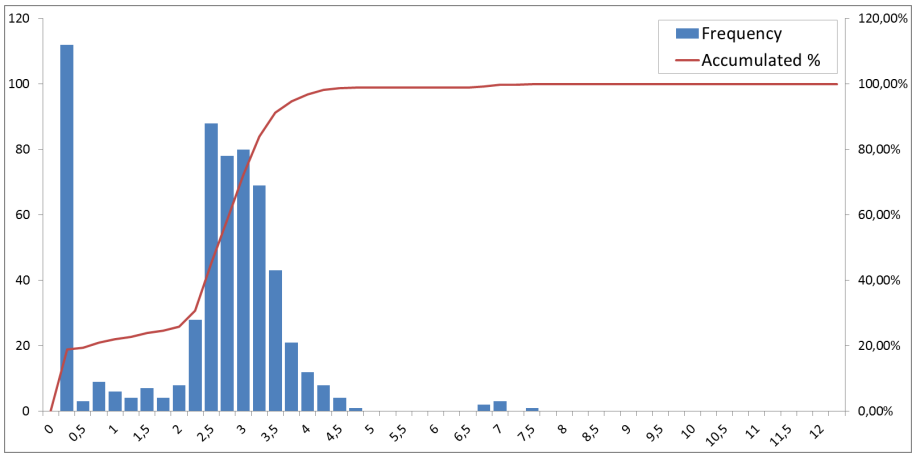



**(b)**

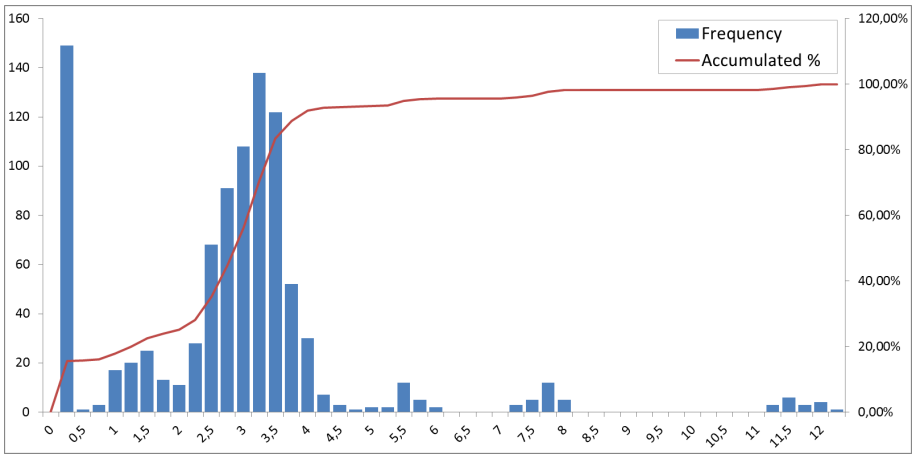



**(c)**

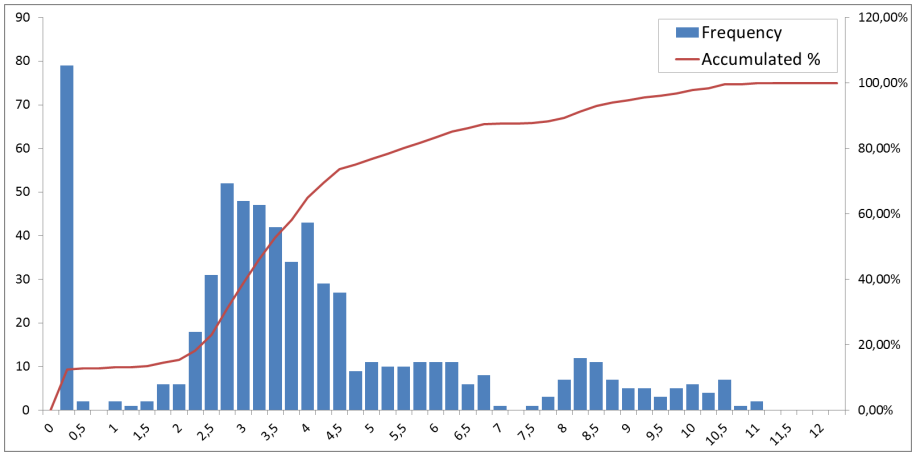
**Figure 4.17:** Distribution of maximum gradients among black test squares on images with (a) none or light dirt level, (b) medium dirt level, (c) strong dirt level.




**(a)**




**(b)**




**(c)**

**Figure 4.18:** Distribution of maximum gradients among white test squares on images with (a) none or light dirt level, (b) medium dirt level, (c) strong dirt level.

with

$$\Delta_{white} = max(0, n^R_{white} - n_{white}) \tag{4.11}$$

$$\Delta_{black} = max(0, n^R_{black} - n_{black}) \tag{4.12}$$

$$\Delta^{>t}_{white} = max(0, n^{>t}_{white} - 10) \tag{4.13}$$

$$\Delta^{>t}_{black} = max(0, n^{>t}_{black} - 10) \tag{4.14}$$

The $-10$ for $\Delta^{>t}_{white}$ and $\Delta^{>t}_{black}$ are introduced to level out positive detections on actual clean or light dirty samples. Since we chose the maximum gradient threshold at the 90th percentile of clean samples, there will be detections of dirty squares on clean images as well. This buffer is supposed to level that out a bit.

### 4.3.3 Evaluation and Results

Big Dutchman provided 53 sample images for the dirt detection. They were labeled into three categories: clean/light dirt, medium dirt, and strong dirt. Clean and light dirt are summarized into one category, since in practice a pristine environment cannot always be guaranteed for the reference calibration—the system is being installed in an animal stable after all. Hence, the dirt detection should also be able to work with light dirt levels in the calibration image.

Of the 53 sample images 13 were considered as clean/light dirt, 23 were considered medium dirt, and 17 were considered strong dirt. To assess the performance of our method we took 3 random images from the clean/light dirt samples for calibration and tested the dirt detection on 10 sets of 3 random samples from clean/light, medium , and strong dirt samples each. We repeated this procedure 100 times, which amounts to 100 random calibration setups tested on 3000 random test settings.

The results of this evaluation are summarized in table 4.4. Considering the average $\Delta$ as well as the $\sigma_\Delta$ indicates that the three categories are actually separated pretty well. However, the minimum and maximum $\Delta$ values show that it is not perfect. This is not surprising since it is a smooth transition between the categories.

If the system should warn already at medium dirt levels, a warning threshold placed right between Clean/Light Avg. $\Delta + \sigma_\Delta$ and Medium Avg. $\Delta - \sigma_\Delta$ (which is about 4.5) yields the best results. In this case 91% of the clean/light dirt cases and 93% of the medium dirt cases would be detected correctly. The strong dirt cases would be detected at 100%.

### 4.3.4 Discussion

An algorithm for the detection of dirt that could occlude the camera has been developed. The algorithm requires a calibration phase with a clean pane and a clean reference checkerboard background that is applied on the opening hatch on the side of the feeding station. The severity of the dirt level is indicated by the difference of clean squares found on the current image compared to the configuration image.

**Table 4.4:** Results of the evaluation of the dirt detection algorithm.

| | Clean/Light Dirt | Medium Dirt | Strong Dirt |
|---|---|---|---|
| Avg. $\Delta$ | 1.14 | 16.68 | 41.59 |
| Min. $\Delta$ | 0 | 4 | 28 |
| Max. $\Delta$ | 7 | 31 | 54 |
| $\sigma_\Delta$ | 1.87 | 10.31 | 9.14 |

The algorithm employs the gradient information on the image as indicator for dirt. By grouping the gradient information by means of the checkerboard pattern, we reach a controlled and even distribution of the information, which makes the detection robust towards punctual peaks.

The evaluation shows that the algorithm can distinguish three categories of dirt levels very well. However, due to the small number of test images the evaluation status is yet limited. An evaluation on more data and possibly long-term data captured in the field would be future work. Furthermore the actual impact of dirt on the uterus detection would be a valuable investigation to be conducted in the future.

Considering this rather simple image processing task one might argue that it does not really require the OBIA framework for its implementation. And it is true in the sense that we only used one real object feature ($rectFit_1$) and did not make any use of interrelations between objects. All objects were inspected individually. But even so, using the OBIA framework made exactly this processing of individual regions very easy and efficient. Hence using OBIA was still advantageous for the sake of development efficiency.

# Chapter 5

# Histological Vessel Reconstruction on Murine Liver Samples

**Acknowledgments** All histological images used for the work in this chapter were provided by and are used with permission of the Department of General, Visceral and Vascular Surgery of Prof. Dr. Dahmen at the Friedrich-Schiller-University Jena. I would like to thank Prof. Dr. Dahmen, Dr. Dirsch and Stephanie Lange for their consulting regarding histopathology, as well as Christiane Engel and Andrea Koller for creating reference labels for my evaluations.

Furthermore I want to thank my colleague Johannes Lotz for helping me to implement and set up the elastic registration step.

The work in this chapter was partly conducted within the Virtual Liver Network initiative, funded by the German Federal Ministry for Education and Research.

**Publications** We have previously published an approach on vessel reconstruction from serial sections based on intensity thresholding in the *Proceedings of SPIE Medical Imaging 2013* [139] . However, we found that this was not robust enough for all stainings (e. g. GS or BrdU) and had difficulties dealing with small vessels and vessels in which residue remained. We therefore present a completely new approach in section 5.2 of this chapter, which is based on the paper published in the *Proceedings of SPIE Medical Imaging 2016* [140] .

On the registration of histological slides we published a paper in the *Journal of Pathology Informatics* [135]. Section 5.1 is based on this, but has been revised and extended compared to the paper.

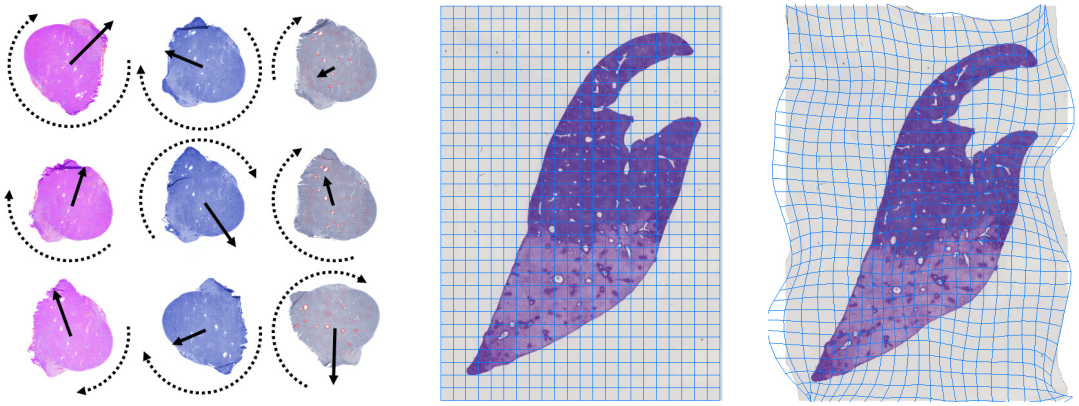
The challenge of understanding liver functionality, physiology and morphology is addressed all over the world by many research groups and scientists from several disciplines concentrating on different aspects from the whole organ to single cells. Understanding the vascular system of the liver is a crucial part of understanding the liver as a whole, since it is the structural system connecting all scales withing the organ. If it would be possible to build and simulate a reasonably realistic model of the liver vessel structures, this would have a great impact in different areas: For the development of drugs, drug delivery through the vascular system could be simulated or regeneration of vessels after hepatectomy could be predicted. However, to be able to create such a model we would first need to analyze real vascular systems in different conditions (healthy, pathological, different stages of liver regeneration, etc.) to have something to base the models on. For a thorough analysis it is important to assess the vessel tree all the way from the large-scale vessels until the fine vessel structures, if possible until the sinusoids.

For this type of research typically mouse and rat livers are used and the analysis is done based on different imaging modalities. To assess the vascular system of a murine liver in 3D and on a whole organ scale MicroCT images are typically used. However, the resolution is not high enough to exhibit the finer vessels. Alternatively employing confocal laser scanning microscopy (CLSM) delivers very high details such that even sinusoids are visible, but this technology has a very limited field of view, especially in z-direction. Thus there is a gap between the scales delivered by those modalities: On the one hand, MicroCT can deliver an overview of the whole main vessel structure, while on the other hand, CLSM can deliver very fine-scale details but only in a very limited region, thus missing the broader context.

The recent introduction of histological whole slide scanners was a significant step forward in histology and histopathology, since they enable a digitized workflow and the broader application of software tools for analysis. Furthermore, this technology is also able to fill the above mention gap between the scales of other imaging modalities. It provides a fine-scale resolution on a whole organ scale (for small mammals). However, the inherent disadvantage of histological whole slide imaging is the fact that the acquired images are only two-dimensional. To acquire a 3D volume means to prepare serial sections of the whole organ sample, which is a tedious manual task for the medical technical assistant. Then, after scanning all slides, we can attempt the digital reconstruction.

Reconstructing vessels from a stack of serial whole slide sections can be roughly separated into two steps: The first step is the registration of the serial sections to regain a 3D volume. The second step is the segmentation and reconstruction of the vessel structures from the 3D volume.

The registration is required, because the samples undergo different deformations during the acquisition process: Linear deformations are introduced when placing the sample onto the glass slide (see Figure 5.1a) while small elastic deformations are introduced during the cutting process with the microtome (see Figure 5.1b and 5.1c). Severe non-linear deformations like tears and overlaps can also occur due to the fragile nature of the samples,



(a) (b) (c)

**Figure 5.1:** Example for typical deformations occurring during acquisition of histological whole slide images: (a) Linear deformations (rotation and translation) introduced when placing the sample on the glass slide. (b) (c) Illustration of elastic deformation introduced during the microtome cutting process.

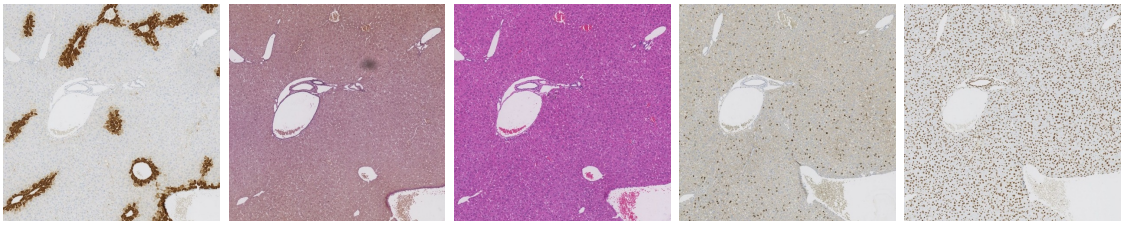
but are impossible to be compensated by a registration process.

The reconstruction of murine liver vessel structures is currently more of interest in a research context rather than for clinical questions. However, a general registration method for histological serial sections would have potential applications in the diagnostic routine as well. For example Onozato et al. [108] report that a 3D reconstruction of histological samples improved the classification of lung adenocarcinoma. Another opportunity is the so called "virtual double staining" or "colocalization analysis". The idea behind this is to apply different stainings to consecutive samples, register those samples and correlate the staining. Since a single staining only exposes very specific tissue parameters the virtual combination of those would open unprecedented possibilities for the assessment of histological samples.

## 5.1 Registration of Histological Whole Slide Images

As mentioned in the introduction of this chapter, a general registration method for histological serial sections has many potential applications. Regarding the development of a registration method we therefore aimed to expand the scope beyond reconstruction of murine liver vasculature to reach a solution that may be applied to various tissue types.

One important requirement to consider is that histological sections need to be stained to expose certain tissue properties. Many different staining types exist, which manifest in various visual appearances. Figure 5.2 shows a few examples of different stainings. Furthermore, not only can the staining types vary but also using the same staining does not guarantee to result in the exact same visual appearance. This depends on many factors



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

**Figure 5.2:** Examples of different stainings on consecutive histological sections (already registered): (a) GS, (b) EvG, (c) H&E; (d) BrdU, (e) HMGB1.

like the concentration, age and manufacturer of the stain, slice thickness or the scanner. Hence for the registration we cannot rely on the color information.

We choose a two-step registration approach to address the different types of deformations. First a rigid registration will compensate for the large linear deformations that are mainly introduced during placing a sample on the slide. Subsequently an elastic registration is applied to compensate for the smaller non-linear deformations. The rigid registration is based on aligning interest points (IPs) to be invariant towards different stainings.

### 5.1.1 Related Work

Several works have been published regarding the registration of histological slides. Many of them focus on elastic registration methods as general solution but give little consideration to the rigid deformations which can be very strong due to the manual placing of the tissue samples on the slides. Wirtz et al. [171] presented a fast registration method to reconstruct a whole rat brain from serial sections, but for pre-alignment performed a principal component analysis, which is prone to errors especially if the tissue sample is of rather circular shape. Later Schmitt et al. [133] continued the work and published a detailed description of their intensity-based registration method for reconstructing a whole rat brain. Modersitzki et al. [91] published a work on the reconstruction of a human brain form serial sections. As a preprocessing step they also pre-aligned the sections using a principal axis transformation, which will fail for tissue sections with less stable principal axes orientations.

Dauguet et al. [32] present a method for reconstructing a baboon brain from histological images, using block face images as external reference. However, such images are in most cases not available since they require additional effort and hardware during the slide acquisition. Ourselin et al. [109] and Cifor et al. [23] utilize a block matching algorithm which relies on an intensity-based similarity matching, which they also use for the registration of histological brain images. Also Bağci and Bai [6] address the topic of reconstructing a 3D brain from serial sections and rely on an intensity-based similarity matching which can

be unreliable if consecutive slides are stained differently. Another paper on mouse brain reconstruction is presented by Yushkevich et al. [178]. They perform a combination of slice-to-slice registration and histology-to-MRI registration. The slice-to-slice registration is aiming at reconstructing details, while the histology-to-MRI registration assures the correct reconstruction of the overall 3D shape of the brain. However such an additional MRI image of the same tissue sample can in general not be assumed to be available.

As we can see there seems to be some focus on brain reconstruction. However, several works have also been published regarding registration and reconstruction of different kinds of organs and tissue. Feuerstein et al. [36] present a general method for registering histological sections which they evaluated on a synthetic dataset as well as on serial sections of a rat kidney. For the registration of the serial sections they additionally use block-face images as reference, just like Dauguet et al. [32].

Sharma et al. [145] conducted a feasibility study of current registration techniques for high resolution tissue imaging. Their focus, however, was not registration of whole slide images but of small histological entities like cells and nuclei in a small region of interest. Furthermore they only conducted the feasibility study for synthetic images. Tan and Dong [159] presented a method for the affine registration of histological slides employing the sharpest contour curvatures as features for a NURBS spline guided alignment refinement. Again for the initial alignment they employ an intensity-based registration. Mosaliganti et al. [96] presented a rigid registration scheme for reconstructing serial sections of mouse placenta. Their initial registration is based on a principal component analysis and for refinement they employ mutual-information-based registration technology from ITK but introduce random perturbations to the process to avoid local minima.

In a short paper Huang et al. [60] present a two step approach that first performs a rigid pre-registration and then a non-rigid refinement. For the rigid registration they attempt to cluster red blood cells by detecting red pixels and then match similar candidates by their size and distance to find the most likely transformation. The non-rigid registration is established by finding corresponding points by matching $100 \times 100$ pixel-sized windows. Details on how the matching is actually performed are not given. Also besides example images of a few results no evaluation was performed. It seems doubtful that this method could actually show reasonable performance in practice. Alone the clustering of blood cells by thresholding red pixels makes this method instable for different stainings.

A method that relates more closely to our approach was published by Lee and Bajcsy [71]. They use a sphere-based region growing to segment vessel structures in blocks of confocal laser scanning microscopy (CLSM) images, then estimate the trajectories of the vessels and align images by fusing the trajectories. However, this is only possible in CLSM images, because the technology allows for acquiring 3D images, even though with a very limited depth field, while serial whole slide imaging creates strictly 2D images.

Braumann et al. [13] also presented a method in which they do not rely on the image data itself alone, but first perform a classification of each pixel so that even under different stainings pixels belonging to the same tissue type are recognized as such. Those class label

images are then used for an elastic registration based on optical flow.

Lotz et al. [76] published a method to register differently stained histological whole slide images. They employ a tile-based approach and perform an affine registration of tiles while consecutively increasing the zoom level. At the highest resolution an elastic registration is performed on the tile. As distance measure they use normalized gradient fields [47]. They also perform a pre-alignment which requires a segmentation mask of the tissue.

Another approach that also follows the idea of registering overlapping patches is presented by Magee et al. [80]. However, in contrast to Lotz et al. [76] the registration of the patches itself is always strictly rigid. The elastic registration is then implicitly created by merging the rigid transformations of the patches. To deal with different stainings they classify pixels into tissue classes and use mutual information as distance measure on the classification feature images.

Song et al. [155] also perform a classification to identify the same tissue classes in different stainings based on clustering and mutual information. For the registration they employ a phase correlation based method. For registering slides of the same staining they directly use a grayscale version of the slides, otherwise the probability maps of the tissue classification are used. The non-rigid registration is established by a multi-scale rigid registration of small patches.

Another approach that uses phase correlation for rigid registration is presented by Roberts et al. [127]. They actually present a complete tool for registration and visualization of the 3D reconstruction of histological whole slide images. As the methods above the registration is performed patch-based on multiple scales. From the patch registrations a global B-spline-based non-rigid transformation is estimated.

Tasdizen et al. [161] also follow the idea of registering patches to reconstruct electron microscopy serial sections. Different from the light-microscopy-based imaging we are dealing with, electron microscopy imaging produces gray scale images which are also rather stable from slide to slide. To evaluate the displacement they use the the Fourier shift property.

The above mentioned registration methods use the image information directly for registration and as distance measure. We follow an approach that is based on first detecting interest points (IPs). Such IPs can describe any structure or feature in an image that can be recognized on consecutive slides. The goal of the registration is then to only match those IPs, thus making the approach independent of the type and appearance of the image—in our case independent of different stainings.

The classic approach to IP matching is the iterative closest point (ICP) algorithm by Besl and McKay [9]. However, their method is very limited in dealing with cases in which several IPs from one image do not have corresponding IPs in the other image (outliers).

Rangarajan et al. [120] presented an algorithm that extends the Procrustes method by Goodall [44] so that it can deal with unknown correspondences. Their method also handles the outlier problem. However, the test samples in their evaluation contain a significantly

lower number of outliers than corresponding points. It is not clear if their algorithm would still perform well if the number of outliers reached 30% or more.

In a later work Chui and Rangarajan [22] present a sophisticated point matching algorithm that even allows for non-rigid registration. Their results seem very promising and stable under high outlier-to-data ratios. Their method relies on two free parameters though. They also note that the computation time can reach $O(N^3)$ in the worst case. Furthermore, all cases in their evaluation examples remain with the full set of original corresponding points. Outliers are added as additional random points. How the algorithm performs if correspondences in the actual target structures are missing is not assessed.

Cross and Hancock [27] as well as Wells III [169] presented probabilistic approaches which use the expectation maximization algorithm to find an appropriate point matching. However, Chui and Rangarajan [22] note that "[...] they do not enforce one-to-one correspondence". Also Hinton et al. [56] followed a probabilistic approach to match point sets. Their work is aimed at recognizing hand-printed characters and assume the point sets can be modeled with B-Splines. Hence their approach is not generally applicable.

Belongie et al. [7] and Mori et al. [95] describe a point matching approach using a novel shape context feature. The basic idea is to consider the vector from one point to all other points and create a histogram of the relative coordinates. Correspondences are then established based on the similarities of those histograms per point. It seems, however, unlikely that this approach would be able to deal with a sparse set of points.

The method proposed by Berg et al. [8] matches points by incorporating geometric blur as feature point descriptor. For establishing correspondences it therefore partly relies on matching feature points exposing a similar descriptor value.

Most of the above mentioned point matching methods focus on object recognition. Most of them also assume a rather large set of points to be matched. It is unclear how they would perform under a registration setting given only a sparse set of IPs and a rather high ratio of outliers/non corresponding points.

### 5.1.2 Methods

In our approach we address the strong linear deformations independently from the small non-linear ones. For those non-linear deformations the related works show many promising approaches, therefore we chose one to build upon. However, the strong rigid deformations are quite often not especially considered in the related work. It is often assumed that images are already fairly well aligned, that a coarse initial alignment can be achieved easily, or that the elastic registration method inherently takes care of it. The rigid deformations can be quite challenging to address, though. Especially if the shape of the two tissue sections is rather regular or different from one slide to the next (due to missing parts), simple methods like a principal component analysis are incapable of fixing the misalignment. Hence a closer look into the tissue is required also for the rigid deformations. Trying to solve it all in one elastic registration step can however impair the compensation for the actual small non-linear deformations, since the regularization has to allow for large

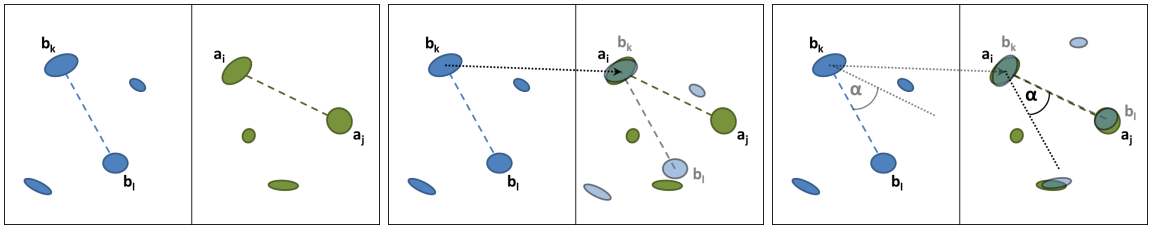



**Figure 5.3:** Compute a transformation based on a double pair: (a) Original configuration. (b) Translate $b_k$ onto $a_i$. (c) Pivot around $a_i$ to align $a_j$ and $b_l$.

deformations as well.

In our approach we therefore focused on developing a robust rigid registration which can be used in combination with any subsequent elastic registration. In this way the elastic registration can be tailored to focus on fixing the small non-linear deformations.

To be invariant towards different stainings our rigid registration method relies on processing interest points (IPs) instead of the image data itself. An IP is a prominent feature that will most likely appear at corresponding positions on consecutive slides. Since this cannot be guaranteed for all IPs, our method is especially designed to deal with cases in which several IPs from one image do not have corresponding IPs in the other image.

Compared to the registration methods for histological images discussed in the related work, this is a different approach. However, using IPs for registration is not a new idea in general, as we also saw in the related work. The difference of our approach towards the point matching algorithms presented by those other authors is that our methods aims at being able to register images given only a sparse set of IPs and can still handle a fairly high ratio of outliers. Our idea is to go for a few prominent points instead of many weaker ones — basically just like a human would only consider a few key points to establish the orientation of two similar shapes.

Furthermore, our IP-based registration is designed in such a way that it can be used with any type of IP as well as with different types of IPs at the same time. We do not make any assumptions about the nature of the IPs fed into the rigid registration algorithm. An optional descriptive value called "strength" may be used if applicable, but it is not required for the algorithm to work. We showcase two different types of IPs that worked well on histological images.

#### 5.1.2.1 Rigid Registration with Interest Points

Our registration algorithm considers two sets $A$ and $B$ of IPs to be matched, with $A = A_1 \cup .. \cup A_X$, $B = B_1 \cup .. \cup B_X$ where $X$ is the number of IP types. Thus the subsets $A_x, B_x$ represent IPs of the same type. Based on the IPs in $A$ and $B$ we determine different transformations to find the one creating the best match between $A$ and $B$. For that the

algorithm iterates over all possible combinations of double pairs $\{(a_i, a_j), (b_k, b_l)\}$, with $a_i \neq a_j$; $b_k \neq b_l$ and $a_i, a_j \in A_x$; $b_k, b_l \in B_x$ for all $x \in X$.

In each iteration step we compute the transformation matrix $T$ that will translate $b_k$ onto $a_i$ and rotate the positions such that $a_i$, $a_j$, $b_k$ and $b_l$ are in line (see also Figure 5.3):

$$\vec{t} = \begin{pmatrix} a_i.x - b_k.x \\ a_i.y - b_k.y \end{pmatrix}, \vec{v} = \begin{pmatrix} a_j.x - a_i.x \\ a_j.y - a_i.y \end{pmatrix}, \vec{w} = \begin{pmatrix} b_l.x - b_k.x \\ b_l.y - b_k.y \end{pmatrix} \tag{5.1}$$

$$\alpha = \arccos \frac{(\vec{v_0} \cdot \vec{w_0} + \vec{v_1} \cdot \vec{w_1})}{\sqrt{\vec{v_0}^2 + \vec{v_1}^2} \cdot \sqrt{\vec{w_0}^2 + \vec{w_1}^2}} \tag{5.2}$$

In case $\vec{v_0} \cdot \vec{w_1} - \vec{v_1} \cdot \vec{w_0} > 0$ we have to set $\alpha = -\alpha$.

$$T = \begin{pmatrix} \cos\alpha & -\sin\alpha & a_i.x + \cos\alpha \cdot (\vec{t_0} - a_i.x) - \sin\alpha \cdot (\vec{t_1} - a_i.y) \\ \sin\alpha & \cos\alpha & a_i.y + \sin\alpha \cdot (\vec{t_0} - a_i.x) + \cos\alpha \cdot (\vec{t_1} - a_i.y) \\ 0 & 0 & 1 \end{pmatrix} \tag{5.3}$$

Now the positions of all $b_m \in B$ are transformed by $T$. Those IPs with the altered position we denote $b_m^T \in B^T$.

To assess the matching quality between $A$ and $B^T$ we developed a custom similarity measure denoted $\Omega(B^T, A)$. For that we first define the matching cost $\delta_{m,n}$ between a $b_m^T \in B_x^T$ and an $a_n \in A_x$:

$$\delta_{m,n} = \delta_{m,n}^{dist} + \delta_{m,n}^{strength} \tag{5.4}$$

$$\delta_{m,n}^{dist} = \frac{\sqrt{(b_m^T.x - a_n.x)^2 + (b_m^T.y - a_n.y)^2}}{\omega + \Delta_{m,n}} \tag{5.5}$$

$$\Delta_{m,n} = 0.5 \cdot \left( \sqrt{\frac{strength(b_m^T)}{\pi}} + \sqrt{\frac{strength(a_n)}{\pi}} \right) \tag{5.6}$$

$$\delta_{m,n}^{strength} = \left( 1 - \frac{\min\left(strength(b_m^T), strength(a_n)\right)}{\max\left(strength(b_m^T), strength(a_n)\right)} \right)^2 \tag{5.7}$$

This matching cost considers two things: First, $b_m^T$ and $a_n$ are more likely to be a good match, the closer they are located to each other. The $\omega$ in the denominator of eq. 5.5 is the only adjustable parameter of the method, used to define the expected maximal spatial deviation of corresponding IPs.

The second consideration is the *strength* of an IP which can be any distinguishing feature among the same type of IPs. Hence eq. 5.7 implies that two IPs are more likely to

be a match if their difference in *strength* is small. Furthermore, the *strength* influences eq. 5.5 through $\Delta_{m,n}$, which allows more distance between strong IPs. This term is especially intended for IPs whose *strength* reflects their spatial size, but can be used in any case where it is desired to emphasize strength over distance. However, if this is not applicable for an IP type, we set $\Delta_{m,n} = 0$ for this particular IP type. Likewise, for types of IPs for which a meaningful strength cannot be assigned at all, we set $\delta_{m,n}^{strength} = 0$.

As mentioned before, our similarity measure also has to consider the possibility that some IPs of one set might not have corresponding IPs in the other set, since the IP detection cannot guarantee to return only corresponding points. To be able to include this in the similarity measure we define a fixed matching cost in case a $b_m^T \in B_x^T$ is not assigned to an $a_n \in A_x$ or vice versa (symbolized by $\phi$):

$$\delta_{m,\phi} = 1,\ \delta_{\phi,n} = 1 \tag{5.8}$$

Now the overall similarity measure for sets $B^T$ and $A$ is defined as

$$\Omega(B^T, A) = \sum_{x \in X} \sigma(B_x^T, A_x), \tag{5.9}$$

with $\sigma(B_x^T, A_x)$ being the similarity value for a type subset of IPs.

Since we do not know the correct assignments between IPs from $B_x^T$ and $A_x$, we calculate $\sigma(B_x^T, A_x)$ by finding the assignments that result in the lowest overall matching cost. This is determined as follows: Let $f : \{B_x^T \cup \phi\} \rightarrow \{A_x \cup \phi\}$ be a function of assignments between the IPs from $B_x^T$ and $A_x$, so that each $b_m^T \in B_x^T$ and each $a_n \in A_x$ appears exactly once while $\phi$ may appear arbitrarily often with the restriction that $\phi \rightarrow \phi$ is forbidden. Further let $\mathcal{F}$ be the set $\{f_1, .., f_p\}$ that represents all possible permutations of assignment functions. Note that we explicitly allow assignment functions where IPs from $B_x^T$ are assigned to $\phi$ while at the same time $\phi$ is assigned to IPs from $A_x$. This accounts for the fact that some IPs in $B_x^T$ might actually not have a corresponding IP in $A_x$ and vice versa.

We compute the similarity value $\sigma(B_x^T, A_x)$ for the current transformation $T$ by finding the assignment $f \in \mathcal{F}$ that yields the minimal sum of matching costs:

$$\sigma(B_x^T, A_x) = \min_{f \in \mathcal{F}} \sum_{b \rightarrow a \in f} \delta_{b,a} \tag{5.10}$$

Since we are iterating over all plausible transformations $T_1 ... T_q$, the final transformation $T_{final}$ for the rigid registration is selected such that

$$\Omega(B^{T_{final}}, A) = \min(\Omega(B^{T_1}, A), .., \Omega(B^{T_q}, A)). \tag{5.11}$$

#### 5.1.2.2 Detecting Interest Points

Since our IP-based registration does not make any assumptions regarding the nature of the IPs, you may use any detector that has a high likelihood for finding corresponding IPs

in your images. For our experiments we employed two classic approaches, which worked well for all our histological images: a blob detection within the tissue and a corner detector on the tissue border.

The blob detection is based on a difference-of-Gaussian (DoG) bandpass pyramid (described early by Crowley and Stern [28]). We use levels 4-6 of the DoG pyramid and threshold them at 0.1, 0.2, and 0.4 of the maximum intensity on levels 4, 5, and 6. This gives us a segmentation for each blob, roughly representing the spatially correct size of the underlying structure it represents. The blobs of all levels are then merged into one level to remove duplicates and the $n$ biggest are selected as IPs, with their size as *strength*.

For the corner detection we use Harris' corner detector [50] on a mask of the tissue. This mask is determined by a histogram analysis setting the threshold to cut off the last major peak, which is the bright background of the slide, followed by a morphological open-close operation and a hole filling.

The blob detection is designed to work at a pixel size of around 14.5$\mu$m/px $\pm$ one magnification level and the corner detection operates at 20% of that resolution.

#### 5.1.2.3 Elastic Registration

The rigidly pre-registered images are refined in the elastic registration step. The elastic registration is performed consecutively from slide to slide on the whole slides at a pixel size of around 7.3$\mu$m/px. To avoid a gradual volume collapse over several hundred consecutive registrations as well as to avoid a possible skew in z direction (also known as "banana" or "aperture" effect) we keep every tenth slide unchanged in its rigidly registered position. In the block between those key slides we perform the consecutive registration in forward and backward direction. The back and forward deformation fields for each slide are then combined by gradually blending them from the first to then tenth image in a block, giving more weight to the backward-registration and less to the forward-registration the closer we get to the tenth slide and vice versa.

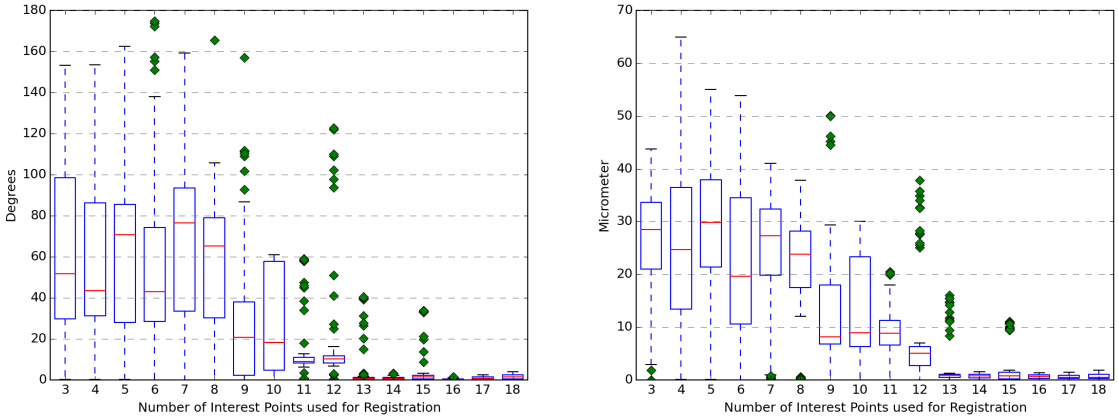
For the registration we use the elastic registration algorithm described by Modersitzki [90] in an efficient, matrix-free implementation that was presented by König and Rühaak [66]. As distance measure we choose the Normalized Gradient Field, as presented by Haber and Modersitzki [47], since it is able to robustly handle the variations in stainings. This setup for the elastic registration was inspired by the work of Lotz et al. [76, 77], who have already proven the applicability of this approach to histological images.

To reconstruct histological slides on a higher resolution than the one on which the registration was performed, the deformation field is simply up-sampled.

### 5.1.3 Evaluation and Results

#### 5.1.3.1 Evaluation of the Rigid Registration

We performed our evaluation against a synthetic dataset as well as real histological data. The synthetic dataset was inspired by Feuerstein et al. [36] and designed to resemble a




**(a)** **(b)**

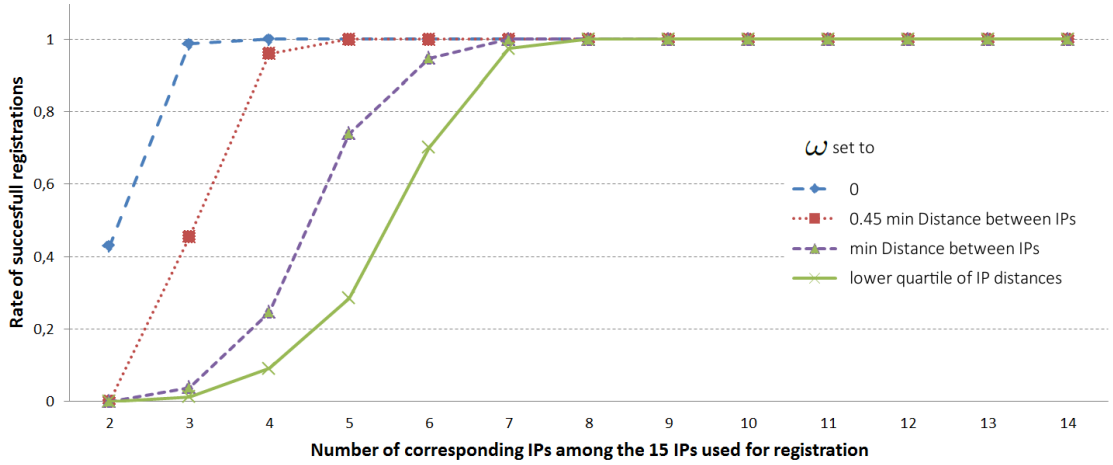
**Figure 5.4:** Registration errors on synthetic dataset depending on number of IPs used. Error in (a) rotation and (b) translation.

murine liver slide at 30$\mu$m and includes a vascular tree model computed by constrained constructive optimization. Details about the dataset can be found in [135].

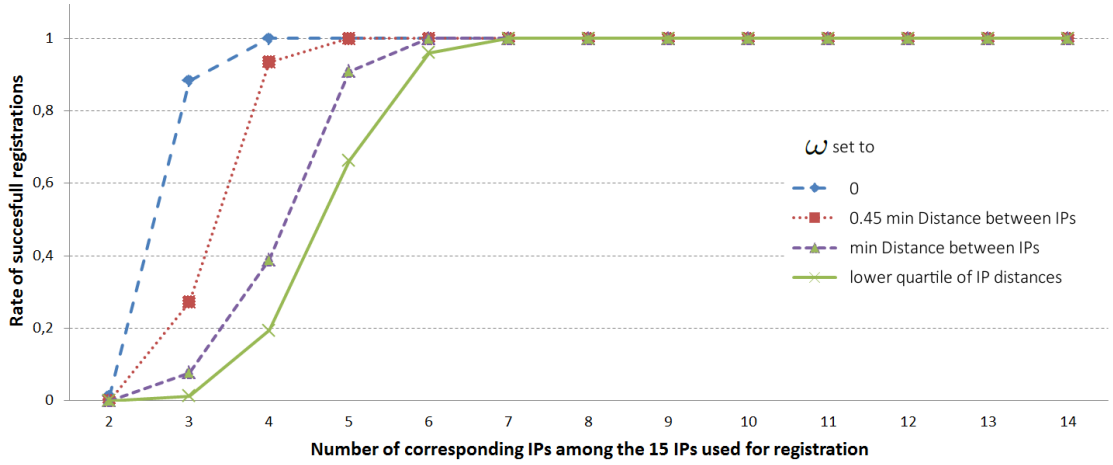
We randomly rotated and translated 100 slides of the synthetic dataset and then registered them with our method. We performed this experiment 16 times, covering a range of 3–18 IPs as input for the registration. The results in Fig. 5.4 show that there is no considerable registration error anymore (with the exception of a few outliers) when using 13 or more IPs for registration.

For the evaluation on real histological data we used 77 samples from mice, rats and humans. The tissue types included liver, spleen, kidney, lung, intestine and pulmonary sarcoma. The following stainings were represented: H&E, EvG, Laminin, BrdU, HMGB1, Beta-Catenin, CK7, Diastase, Fe, Gömöri, Ki67, ASMA, GS. On these cases we wanted to test, how many actually corresponding IPs among all input IPs are required for a successful registration by performing a self-registration. For this we calculated the 28 strongest IPs on each image from which two subsets $A_n$ and $B_n$ with 15 IPs each were selected. The IPs for $A_n$ and $B_n$ were selected so that they have $n$ IPs in common while the other $15-n$ IPs were mutually exclusive. Furthermore, the IPs for $A_n$ and $B_n$ were selected so that their distribution was as compact as possible. We tested our registration for all $n \in \{2, 3, ..., 14\}$ on all 77 images with different settings for $\omega$ (eq. 5.5), once considering IP's *strength* and once without. The results in Fig. 5.5 show that with setting $\omega = 0$ we reach 100% success already with only 4 corresponding IPs. As to be expected, setting $\omega$ higher lowers the early success rate. However, even setting $\omega$ higher than the minimal distance between IPs results in 100% success rate at 7 or 8 corresponding IPs. Fig. 5.5b shows that considering the *strength* improves the success rate for higher settings of $\omega$.

To consider the fact that in reality corresponding IPs most likely will not lie at the exact same position, we performed another self-registration test but this time shifted the

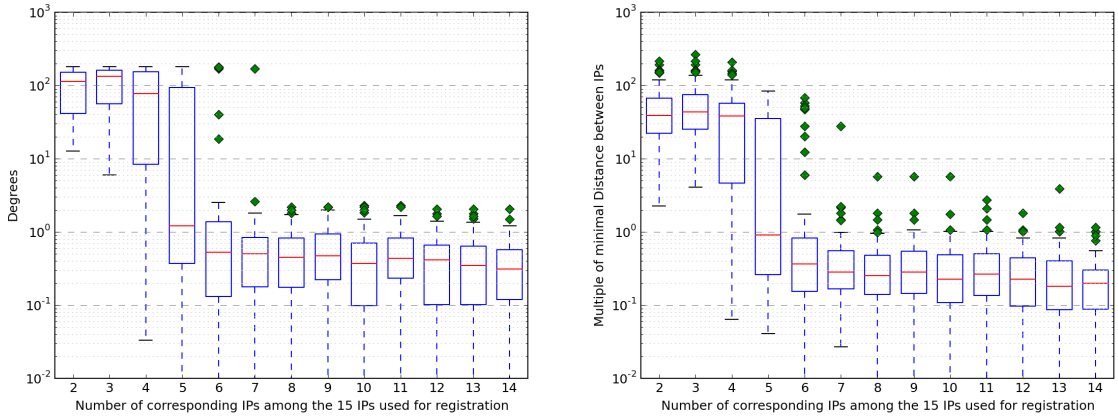



(a)




(b)

**Figure 5.5:** Successful self-registrations on histological images with non-shifted IPs: (a) Not using IP's *strength*. (b) Using IP's *strength*.




**(a)** **(b)**

**Figure 5.6:** Registration error for self-registrations on histological images with shifted IPs. Error in (a) rotation and (b) translation.

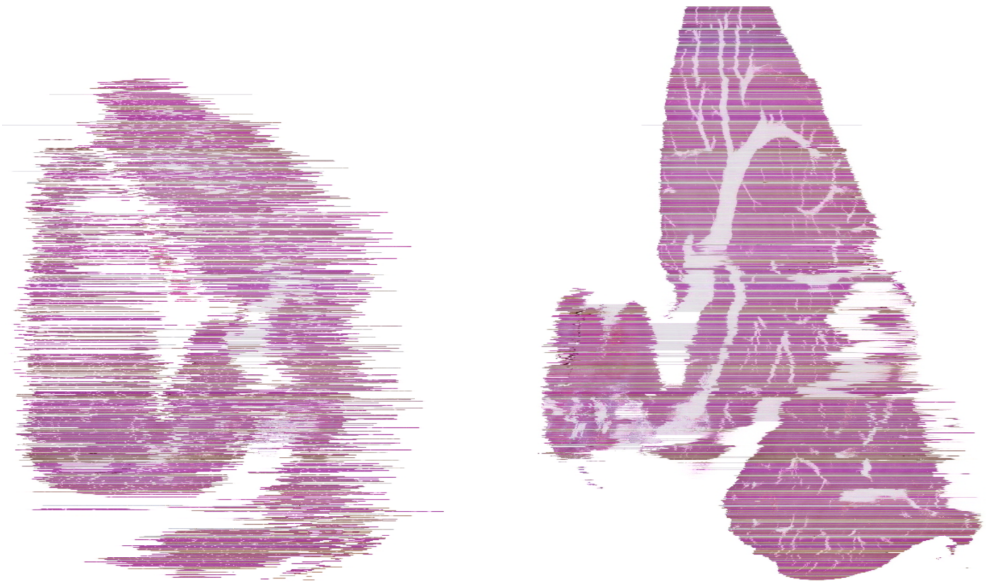
positions of all IPs in $B_n$ in a random direction and distance between 1px and 30% of the minimal distance between the IPs. We set $\omega$ to 65% of the minimal IP distance. In Fig. 5.6 we can see that at around 6–7 corresponding IPs the registration is successful already in most cases.

With our method we were able to reconstruct several datasets, among which the biggest were two mouse livers of 674 and 1403 slides, with neighboring slides always differently stained. Fig. 5.7 shows one of those liver datasets. For that dataset we used 12 blob and 8 corner IPs per image (automatically detected on 14.5$\mu$m resolution), the average registration computation time was 0.71 seconds (single-core) and a considerable registration error (by visual inspection) occurred in less than 2% of the slides.

#### 5.1.3.2 Evaluation of the Interest Point Detection

Since our rigid registration method depends on the interest point detection, we also wanted to evaluate, if the chosen IP detectors are providing a sufficient number of corresponding IPs for our task. For this we used 400 pairs of consecutive histological images of different stainings which were semi-automatically registered beforehand. All images were checked and manually corrected to assure a good registration for each image pair.

For each pair we performed a corner and blob detection at 14.5$\mu$m/px and checked how many corresponding IPs were found between the two neighboring slides. We considered two IPs as corresponding, when the distance between them was less than 250$\mu m$. For one evaluation the IP detector was configured to find 15 IPs (the 5 strongest corners and the 10 strongest blobs) to be able to compare the results to our evaluation of the rigid registration (see section 5.1.3.1). The results are shown in Fig. 5.8. We can see that, considering the distribution of the total number of corresponding IPs in Fig. 5.8c, in



(a) (b)

**Figure 5.7:** Cross section of a stack of mouse liver serial sections (a) before and (b) after rigid registration. Both images show a maximum intensity projection over 300$\mu$m in depth.

almost all cases more than 6 matching IPs are found. And as our evaluation of the rigid registration before showed, given 15 IPs in total, a number of 6–7 matches almost always results in a successful registration.

We conducted the same evaluation again with another setup of the IP detector. This time it was set to find 20 IPs in total, the 8 strongest corners and the 12 strongest blobs. This setting was our standard setup for registration tasks on histological images. It has a slightly higher rate of corresponding IPs. Consider the histogram in Fig. 5.9a.

Overall we can conclude that the two simple IP detectors we have chosen are sufficient to provide enough corresponding IPs for a successful rigid registration, even when they are setup to find only a small number of IPs.

#### 5.1.3.3 Evaluation of the Elastic Registration

Since the applicability of the elastic registration method we are using has already been shown by Lotz et al. [76, 77] we did not perform an extensive evaluation of this particular step in our pipeline. The visual results look convincing and confirm that the elastic registration improves the rigid registration in details, as Fig. 5.10 shows exemplarily.

However, we did perform an evaluation on the deformation fields of our registration to confirm their general plausibility. For this we analyzed the Jacobi determinants $det(J_f)$ of the deformation fields, following an evaluation approach also performed by Polzin

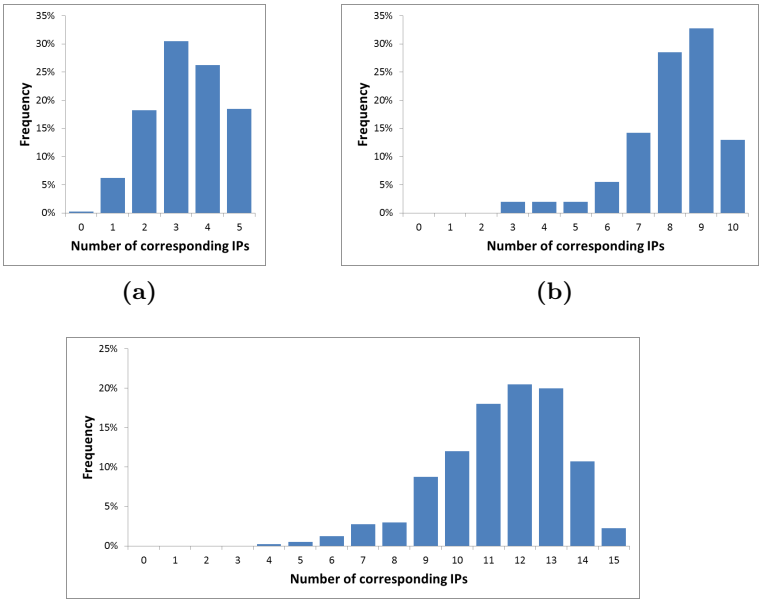



**Figure 5.8:** Distributions for number of corresponding IPs over 400 test image pairs using 15 IPs in total (5 corner IPs and 10 blob IPs). Histograms considering (a) only corner IPs, (b) only blob IPs, (c) corner and blob IPs.

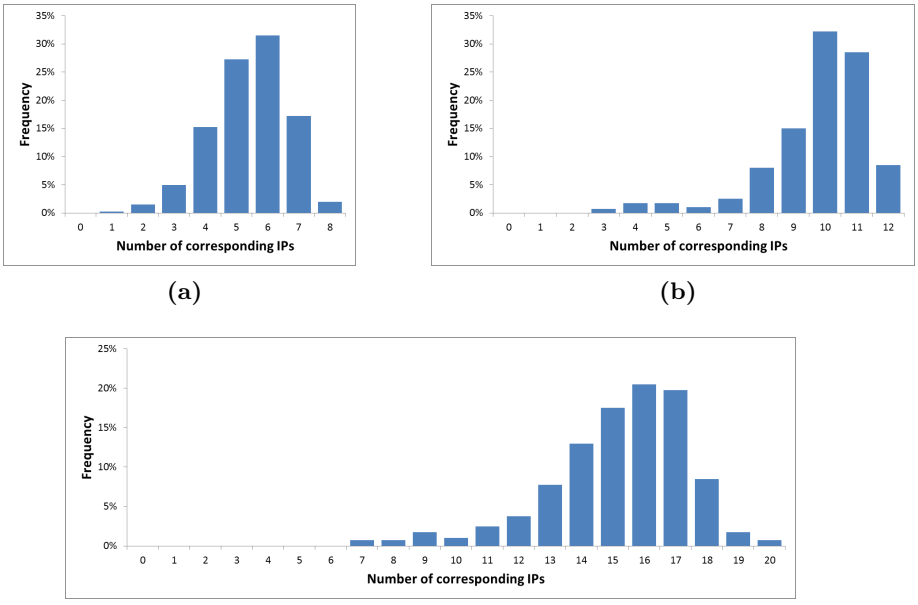



**Figure 5.9:** Distributions for number of corresponding IPs over 400 test image pairs using 20 IPs in total (8 corner IPs and 12 blob IPs). Histograms considering (a) only corner IPs, (b) only blob IPs, (c) corner and blob IPs.

(a) (b)

**Figure 5.10:** Cross section of a stack of mouse liver serial sections (a) after rigid registration and (b) after subsequent elastic registration. Both images show a maximum intensity projection over 300$\mu$m in depth.

et al. [117]. The Jacobi determinant $det(J_f)$ indicates the factor of local volume change caused by the deformation.

We analyzed the deformation fields of 2370 registered histological slides, which were taken from stacks of serial sections from 4 different mice. The minimum encountered value for $det(J_f)$ was 0.52, the maximum was 1.65. This indicates that no alarmingly strong contraction or expansion occurs. The average value for $det(J_f)$ was 1.0 with a standard deviation of 0.03. This proves an overall volume preservation and shows that the elastic registration indeed only induces small deformations to improve in details.

### 5.1.4 Discussion

We presented a staining-invariant interest-point-based rigid registration for histological images. Our algorithm works robustly on small sets of around 15 IPs even if less than 50% of them are actually corresponding. This is a considerable advantage over other approaches. The prerequisites for our method are good IP detectors that are able to find small sets of IPs on two differently stained slides, so that at least a subset of them corresponds in both sets. The proposed blob and corner detectors do that for histological images.

Due to the optimal cost search for the similarity measure our approach is limited in the set size it can handle time-efficiently. Even with an optimized implementation, having more than 30–40 IPs in one set can result in single-core computation times of minutes

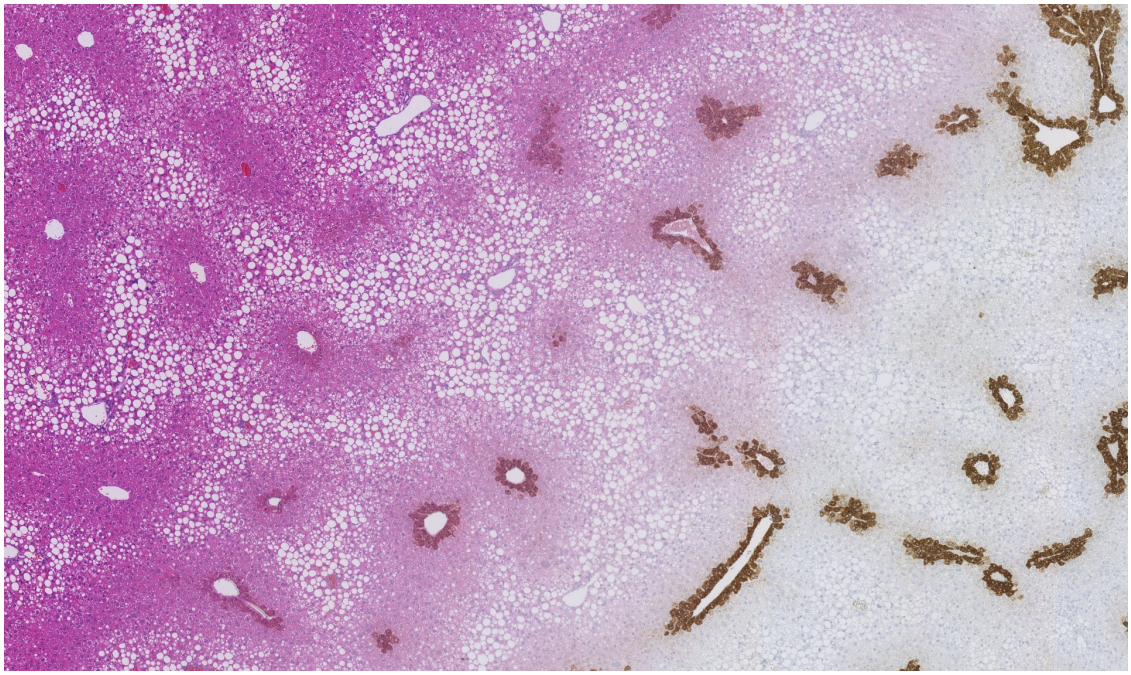


**Figure 5.11:** Example of a virtual double staining by applying our registration method to consecutive H&E and GS stained slides and then blending them.

rather than seconds. However, in our experience on different datasets, using at most 12–18 IPs per set and IP type was always sufficient.

Furthermore the evaluation of the IP detectors showed that they are usually able to find enough corresponding IPs for a successful registration, even when limited to 20 or less IPs in total. Hence the optimal cost search is not a limitation. It could actually be considered a strength, since our method is finding the global minimum.

A current limitation of our method is that it is only designed to work in 2D. However the principle of the rigid registration could be easily extended to 3D. For this we would need to use 3 points (forming a plane) to generate the transformation candidates and adapt the similarity measure to consider 3D coordinates.

While we have shown that the two rather simple IP detectors we chose work well for the histological images available to us, this might not be the case for all types of tissue or staining. Hence it might be necessary for different histological images to use more advanced IP detectors. Some examples from the literature to consider are: Shi and Tomasi [147], Smith and Brady [152], Mahesh and Subramanyam [81], Dalal and Triggs [30].

Furthermore, by choosing proper IP detectors, our rigid registration method may be used with any type of images also outside the histological domain. This is a particular strength of our approach. It does not make any assumptions regarding the type of images. It also makes no special assumptions regarding the IPs. Any type of IP detector can be used as long as it provides coordinates (plus optionally a "strength" per IP) and sufficient

probability of finding corresponding positions on the two images to be registered.

The combination of our novel rigid registration with an established elastic registration method allows us to reconstruct stacks of serial sections. This is a prerequisite for our next step, the segmentation of vessel structures (see section 5.2). Besides that, the presented registration method was already successfully used to reconstruct stacks for the purpose of a zonated quantification of steatosis in an entire mouse liver [134]. Furthermore our registration pipeline could also be used to create virtual double stainings. See Fig. 5.11 for an example of a double staining created with our registration method. As mentioned in the introduction to this chapter, this kind of technology could be of great benefit for histological routine work.

## 5.2 Reconstruction of Vessel Structures from Whole Slide Image Stacks

The segmentation of vessel structures in stacks of histological images poses several challenges. As for the registration we have to deal with different stainings and with the fact that even the same staining may show variations in its visual appearances among different slides. Furthermore, the vessel structures might still contain residue of blood cells which makes them much harder to distinguish from the actual surrounding tissue. Another challenge is the size. If we want to segment vessels on a fine scale for the whole organ, we have to think about strategies to keep the computation time within feasible limits. The amount of data for a stack of serial sections is large, considering that one stack can easily be 1000–2000 slides, with each slide covering several gigapixels at full resolution.

We present a method for the fully automatic segmentation of vascular structures in stacks of serial sections. Our approach is focused on murine liver, but the underlying methodology is not limited to this.

### 5.2.1 Related Work

Segmenting 3D vessel structures in histological whole slide images is not very present in literature yet. Many publications can be found regarding the detection, density estimation and/or segmentation of microvessels in tumors. Among them are Reyes-Aldasoro et al. [125] who use a seeded region growing to detect microvessels in CD31-stained sections of tumor biopsies. As seeds they rely on the distinctive brown color of endothelial cells under this staining. They use a distance and shape analysis to identify vessels among the connected components. A very particular staining procedure is also used by Goddard et al. [43], which exposes vessels and makes it easy to detect them with a threshold-based method. User interaction is required for this method. Luukkaa et al. [79] use CD34 staining to expose vessels for analysis of tumor sections, similarly to Steiniger et al. [156] who combine CD34 with other stainings to highlight microvessels in spleen samples. They focus on the staining procedure and do not give many details about the image processing,

which seems to be very basic.

Rodriguez et al. [128] use CD34 in combination with methyl green to highlight areas where new blood vessels form. They use neural networks to detect the vessels in those images. They conducted several experiments with different settings. The best experiment reports a detection rate of 75% for vessels.

All the above publications originate from a histology/pathology research background rather than from an image processing background. Furthermore they only regard 2D segmentations.

In a somehow related domain, Funke et al. [40] presented an interesting approach towards reconstructing neurons from electronic microscopy (EM) images in 3D. They use graph cuts to generate several segmentation hypotheses. Then an assignment model is employed, which is using a Random Forest classifier trained on positive and negative examples of assignments. They use geometric and texture features.

Regarding actual publications focusing on segmentation in histological images we selected a few that aim to select different structures than vessels, though. Peng et al. [114] presented a method for the segmentation of glandular structures. They apply an adaptive k-means clustering and identify their target structures by brightness. This is followed by morphological refinement steps and a seeded region growing to reach the final segmentation of individual structures. Their approach, however, is exclusively focusing on H&E-stained images.

A promising approach for fat droplet segmentation and identification on H&E-stained whole slide sections was developed by Homeyer et al. [57]. First pixels are classified into tissue and background based on brightness and saturation thresholding. Among the clusters of background pixels, fat droplets are identified by a Random Forest classifier using shape features and adjacency statistics as texture features.

To our best knowledge so far no one attempted to segment and reconstruct vessel trees from stacks of histological images of murine liver samples.

### 5.2.2 Methods

Our approach for the automatic segmentation of vessel structures has three main processing steps. The first step is the over-segmentation of the image into small regions to create the initial objects for OBIA. The details are covered in section 5.2.2.1. In the second step we perform a classification of these regions to determine if they are part of a vessel or not. This is described in section 5.2.2.2. The third step is a refinement classification to improve the results of the initial classification. Details on the refinement step can be found in section 5.2.2.3.

In section 5.2.2.4 we also briefly discuss an approach to reduce computation time by a multi-scale classification.

Prerequisite for the methods described in this chapter is a registered stack of histological image slides for which our registration method, as described in section 5.1, is used. Even though we basically have a 3D histological image now, all following operations work

in 2D on a per-slice basis unless noted otherwise. We chose this approach to accommodate for different stainings from slide to slide. Since different stainings can produce vastly different visual appearances, we use the staining information, which is available for each of the slides, to adapt our method to that.

#### 5.2.2.1 Over-Segmentation

For the over-segmentation of the image we start by reducing the RGB color image to grayscale. For this we use the "lightness" channel from the HSL color space, which can be derived from RGB as (see for example S. E. Umbaugh [165]):

$$grayvalue = \frac{max(R,G,B) + min(R,G,B)}{2} \tag{5.12}$$

This is followed by calculating the Gauss gradient magnitude with $\sigma = 2.0$ (in pixel units). On the gradient magnitude image a watershed segmentation is performed using the algorithm variant proposed by Hahn and Peitgen [48] with the pre-flooding set to 0.5. This yields an over-segmentation that divides the image in small regions that still respect important borders in the image. These regions define our objects set $O$ for the subsequent OBIA processing. All classification and further processing is performed by means of OBIA.

#### 5.2.2.2 Initial Classification

The initial classification is performed per slide for all objects on that slide. The per-slide processing is required due to the different stainings that have to be expected from slide to slide. For each staining type we require a specific classifier, which has been trained only with samples from that type of staining (but from many different images, of course). Using strictly separate classifiers for different staining types removes unnecessary ambiguities in the training process and thus results in a set of overall stronger classifiers than if we would mix up all staining types in one classifier.

The classification itself is performed by a Random Forest [14] classifier which has shown good performance in many applications (see e.g. Fernández-Delgado et al. [35]) and is easy to parametrize.

We use the following basic image-based object features derived from HSV [151] color channels:

- $localBinaryPattern(V_{HSV})$
- $mean(H_{HSV})$
- $median(H_{HSV})$
- $lowerQuartile(H_{HSV})$
- $upperQuartile(H_{HSV})$
- $mean(S_{HSV})$
- $mean(V_{HSV})$

The local binary patterns as a texture feature is derived from the *value* channel because the perceivable texture is mostly represented in the brightness of the pixels. The overall color of an object is mainly represented by the *hue* with the additional information from the *saturation* and *value* channels.

The over-segmentation does not produce very particular shapes, hence we only included some basic shape features:

- *size*
- *eccentricity*

Particularly helpful for the correct classification of an object are also its surroundings. Especially if the object itself is difficult to distinguish only by its own features. If human observers are unsure about the nature of a particular region, considering the context often allows to derive the correct conclusion. Therefore we also include the following context features:

- $contextPattern(localBinaryPattern(V_{HSV}))$
- $contextPattern(mean(H_{HSV}))$
- $contextPattern(mean(S_{HSV}))$
- $contextPattern(mean(V_{HSV}))$

The classification result weight is saved in a new object feature $classWeight_{initial}$ representing the determined probability that an object is part of a vessel.

#### 5.2.2.3 Refinement Classification

The second classification step is a refinement phase intended to improve the results of the initial classification. Just as before it is performed as a per-slide processing with a special classifier for the staining on the particular slide. A prerequisite for the refinement classification is existing $classWeight_{initial}$ values for all objects on the slide to be processed as well as on the top and bottom neighboring slides. The classification itself is again performed by a Random Forest classifier.

As features we use the same as for the initial classification, but we also include the results of the initial classification as features. On the one hand we directly use the $classWeight_{initial}$ as feature, on the other hand we use the class weights of the top and bottom neighboring objects as context features:

- $contextPattern_{Top}(classWeight_{initial})$
- $contextPattern_{Bottom}(classWeight_{initial})$

As argued in the section before, considering the classifications in the neighborhood of an object can help deriving the correct decision for this object. This is especially an advantage in case a classifier for a particular staining type is rather weak. By considering the top and bottom neighbors, which could have a different staining, the classification for the current slide can profit from the results of possibly stronger classifiers. The reason for including not only the neighboring initial class weights, but also the initial class weight of the object itself, is to give the classifier training phase the chance to learn to give more significance to the context feature for more uncertain cases.

One might ask why we use the initial class weights as context features and not directly image-based features. The reason is that, since the stainings are so different, we would require a specifically trained classifier for each possible triple combination of consecutive stainings. Considering having $n$ different stainings, we would require $n^3$ different classifiers and more importantly enough training samples for each of them. The exponential growth makes this approach infeasible. With our approach we just require as many classifiers as stainings. The image-based features are implicitly represented in $classWeight_{initial}$ from the specialized classifiers of the first classification phase.

#### 5.2.2.4 Multi-Scale Classification

Histological images get very large on higher zoom levels. At the highest magnification one slide can easily reach a resolution of $100\,000 \times 100\,000$ pixels. Completely performing our classification pipeline on a high resolution is therefore a considerable computational effort.

We implemented a simple approach to a multi-scale classification. The whole slide is classified on a user-selected lower resolution level. The results are then propagated to the objects on the next higher resolution level by weighted averaging of class weights of the overlapping objects on the lower level.

Objects on the higher level only receive a full re-classification, if their class weight is within a defined range around 50%. In this way computational effort only goes into objects which are not certainly enough detected as vessel or tissue.

### 5.2.3 Evaluation and Results

For the evaluation we used 8 stacks of consecutive serial sections of murine liver samples: 6 stacks of 4 slides, 1 stack of 30 slides, and 1 stack of 40 slides. The images were acquired from three different animals. Of the 94 slides in total, 38 were Hematoxylin and eosin (H&E) stained, 31 Elastika-van-Gieson (EvG) stained, 11 glutamine synthetase (GS) stained, and 14 bromodeoxyuridine (BrdU) stained.

Two assistant medical technicians labeled representative example regions for vessels and tissue on all 94 images and for two zoom levels: One at a pixel spacing of 3.63$\mu$m and the other at a pixel spacing of 1.82$\mu$m. The reference labeling was also limited to sub-regions of $1000 \times 1000$ pixels. For each image of a consecutive stack, the same sub-region was selected. Furthermore, for the higher zoom level (1.82$\mu$m) the sub-region was selected in such a way that it covers exactly the middle of the respective lower zoom sub-region. In this way we collected more than 4200 sample regions for each class (vessel/tissue) on each zoom level for each staining type to perform our evaluation.

For all evaluations of the classification performance, the set of reference samples was balanced to contain the same number of vessel and tissue samples. Within the set of tissue samples, the subsets containing regions that are in the vicinity of vessels and regions that are only surrounded by other tissue were balanced as well. The selection of the samples during the balancing was done randomly.

We set the number of trees for the Random Forest to 25, since larger ensemble sizes increase the execution time but did not significantly improve the classification performance in our experiments.

For our evaluation of the overall classification performance we performed a cross validation. The folds for the cross validation were obtained by a randomized stratified shuffle split that was set to split the reference samples into 20% test and 80% training samples for each generated fold. This validation was repeated 100 times and summarized through the mean value and standard deviation of the accuracy measures. We used the following measures to assess the accuracy of the detection:

$$\text{Sensitivity} = \frac{\text{No. correctly detected vessel objects}}{\text{Total No. vessel objects}} \tag{5.13}$$

$$\text{Specificity VT} = \frac{\text{No. correctly detected tissue objects neighboring vessels}}{\text{Total No. tissue objects neighboring vessels}} \tag{5.14}$$

$$\text{Specificity T} = \frac{\text{No. correctly detected tissue objects not neighboring vessels}}{\text{Total No. tissue objects not neighboring vessels}} \tag{5.15}$$

We differentiated the Specificity by tissue being close to vessels and tissue away from vessels, because the tissue close to vessels is supposedly more difficult to distinguish from vessels. Hence, by evaluating them separately, we are able to see this difference. Please note that for the training of the Random Forest classifiers this differentiation was not applied. The training samples were only labeled as "vessel" or "tissue".

Table 5.1 shows the evaluation for all stainings for 3.63$\mu$m pixel spacing. We assessed the performance for three different classification setups: First, only for the Initial classification phase excluding the context features. We wanted to see how the classification performs only with basic features compared to the whole feature set. Second, we evaluated the full Initial classification alone. And third, we performed the full classification including the Refinement classification. Table 5.2 shows the same evaluation for the 1.82$\mu$m pixel spacing.

For almost all cases, Sensitivity increases or stays stable on a high level with added features and Refinement classification. Only for the GS staining we observe a slight decline in Sensitivity. The Specificity T shows a significant improvement for BrdU, EvG, and GS stainings reaching 95% or higher for the full classification. Specificity T for H&E already starts off at 99% even for the Initial classification with only basic features and stays stable at this rate for the other classification variants. However, for Specificity VT the results show no general improvement with the more advanced classification approaches, but rather a slight decline. It is interesting that the results are actually mostly worse for the initial classification with all features as compared to the initial one with only basic features. The best explanation for this is that by incorporating the context features the border areas are sometimes confused with residue inside a vessel. The Refinement classification

**Table 5.1:** Classification performance measurements for pixel spacing of 3.63$\mu$m. The numbers give the mean values over repetitive cross validations. The numbers in brackets give the corresponding standard deviations.

| Staining at 3.63$\mu$m/px | Classification phase (feature set) | Sensitivity | Specificity VT | Specificity T |
|---|---|---|---|---|
| BrdU | Initial (basic) | 0.86 (± 1E-2) | 0.89 (± 1E-2) | 0.90 (± 1E-2) |
| | Initial (full) | 0.89 (± 1E-2) | 0.82 (± 1E-2) | 0.96 (± 1E-2) |
| | Refinement (full) | 0.93 (± 1E-2) | 0.88 (± 1E-2) | 0.99 (± 3E-3) |
| EvG | Initial (basic) | 0.92 (± 5E-3) | 0.96 (± 5E-3) | 0.96 (± 5E-3) |
| | Initial (full) | 0.94 (± 5E-3) | 0.93 (± 1E-2) | 0.99 (± 3E-3) |
| | Refinement (full) | 0.97 (± 4E-3) | 0.94 (± 1E-2) | 0.99 (± 1E-3) |
| GS | Initial (basic) | 0.95 (± 1E-2) | 0.96 (± 1E-2) | 0.90 (± 1E-2) |
| | Initial (full) | 0.93 (± 1E-2) | 0.94 (± 1E-2) | 0.95 (± 1E-2) |
| | Refinement (full) | 0.95 (± 1E-2) | 0.93 (± 1E-2) | 0.99 (± 5E-3) |
| H&E | Initial (basic) | 0.98 (± 2E-3) | 0.96 (± 4E-3) | 0.99 (± 2E-3) |
| | Initial (full) | 0.98 (± 3E-3) | 0.94 (± 5E-3) | 1.00 (± 1E-3) |
| | Refinement (full) | 0.98 (± 2E-3) | 0.95 (± 5E-3) | 1.00 (± 1E-3) |

**Table 5.2:** Classification performance measurements for pixel spacing of 1.82$\mu$m. The numbers give the mean values over repetitive cross validations. The numbers in brackets give the corresponding standard deviations.

| Staining at 1.82$\mu$m/px | Classification phase (feature set) | Sensitivity | Specificity VT | Specificity T |
|---|---|---|---|---|
| BrdU | Initial (basic) | 0.85 (± 1E-2) | 0.89 (± 1E-2) | 0.81 (± 1E-2) |
| | Initial (full) | 0.88 (± 1E-2) | 0.88 (± 1E-2) | 0.92 (± 1E-2) |
| | Refinement (full) | 0.89 (± 1E-2) | 0.90 (± 1E-2) | 0.96 (± 1E-2) |
| EvG | Initial (basic) | 0.85 (± 5E-3) | 0.96 (± 4E-3) | 0.90 (± 1E-2) |
| | Initial (full) | 0.88 (± 4E-3) | 0.94 (± 1E-2) | 0.95 (± 4E-3) |
| | Refinement (full) | 0.92 (± 4E-3) | 0.95 (± 1E-2) | 0.97 (± 4E-3) |
| GS | Initial (basic) | 0.92 (± 1E-2) | 0.94 (± 1E-2) | 0.84 (± 1E-2) |
| | Initial (full) | 0.89 (± 1E-2) | 0.95 (± 1E-2) | 0.92 (± 1E-2) |
| | Refinement (full) | 0.90 (± 1E-2) | 0.93 (± 1E-2) | 0.95 (± 1E-2) |
| H&E | Initial (basic) | 0.96 (± 3E-3) | 0.97 (± 3E-3) | 0.99 (± 2E-3) |
| | Initial (full) | 0.96 (± 3E-3) | 0.96 (± 4E-3) | 0.99 (± 2E-3) |
| | Refinement (full) | 0.96 (± 3E-3) | 0.96 (± 4E-3) | 0.99 (± 2E-3) |

manages to compensate for this and improve the results in most cases. However a marginal decline compared to the basic Initial classification remains. This can be explained by the incorporation of the class context features of the top and bottom neighboring slides. Since the exact location of vessel borders changes from slide to slide, due to the course of the vessel, the classification information on neighboring slides might actually confuse the classifier, when abrupt changes occur.

Altogether the overall best performance is reached using the full classification including the Refinement. Sensitivity and especially Specificity T are significantly better compared to the other variants and the rates of Specificity VT are only marginally lower. Hence the borders of vessels might not always be captured very well, but with the high Specificity T rates the chances of capturing vessels where there are none are low.

We further evaluated the Refinement classification with a leave-one-slide-out (LOSO) cross validation. The process of staining histological slides does not produce the exact same visual result for each slide. Therefore by a LOSO cross validation we may estimate how well our classification process and the features handle these variations. Fig. 5.12 shows the receiver operating characteristic (ROC) curve for all stainings on 3.63$\mu$m pixel spacing. Fig. 5.13 shows the ROC curve for 1.82$\mu$m pixel spacing. For this evaluation we considered all vessels as positive and all tissue as negative samples.

The ROC curves show very good results that correspond to the results in the tables 5.1 and 5.2 before. The areas under curve (AUC) on the 1.82$\mu$m pixel spacing are a little bit lower than for the respective ROCs on 3.63$\mu$m. Also the standard deviation is a little bit stronger on the higher zoom level, while on the lower zoom level the standard deviation for EvG and H&E staining is almost close to zero. For all others the standard deviation is also considerably stronger compared to the stratified shuffle split evaluation.

The higher zoom level supposedly shows more details and subtle difference between different slides of the same staining, which could explain the slightly lower performance and higher standard deviation. However, we can conclude that overall our classification performance is very good.

Fig. 5.15a–5.15c show some example ROIs and overlays of the classification weights after the Inital and the Refinement classification stage. We can see how the classifier becomes much more certain about the classes of objects in the Refinement step. A rendering of an ROI of vessels that were automatically segmented by our method is depicted in Fig. 5.15d.

As a final evaluation we looked into the multi-scale classification. We applied the full classification pipeline to the 3.63$\mu$m/px images and propagated the results to the objects on the 1.82$\mu$m level as described in section 5.2.2.4.

We now defined a re-classification offset $\delta$ and started the feature extraction and classification only for objects which had a classification weight within $0.5 \pm \delta$. In Fig. 5.14 we plotted the classification performance and required re-classification rate for $\delta$ in a range from 0.0–0.5.

We can see that setting $\delta$ somewhere between 0.3 and 0.4 would already achieve almost

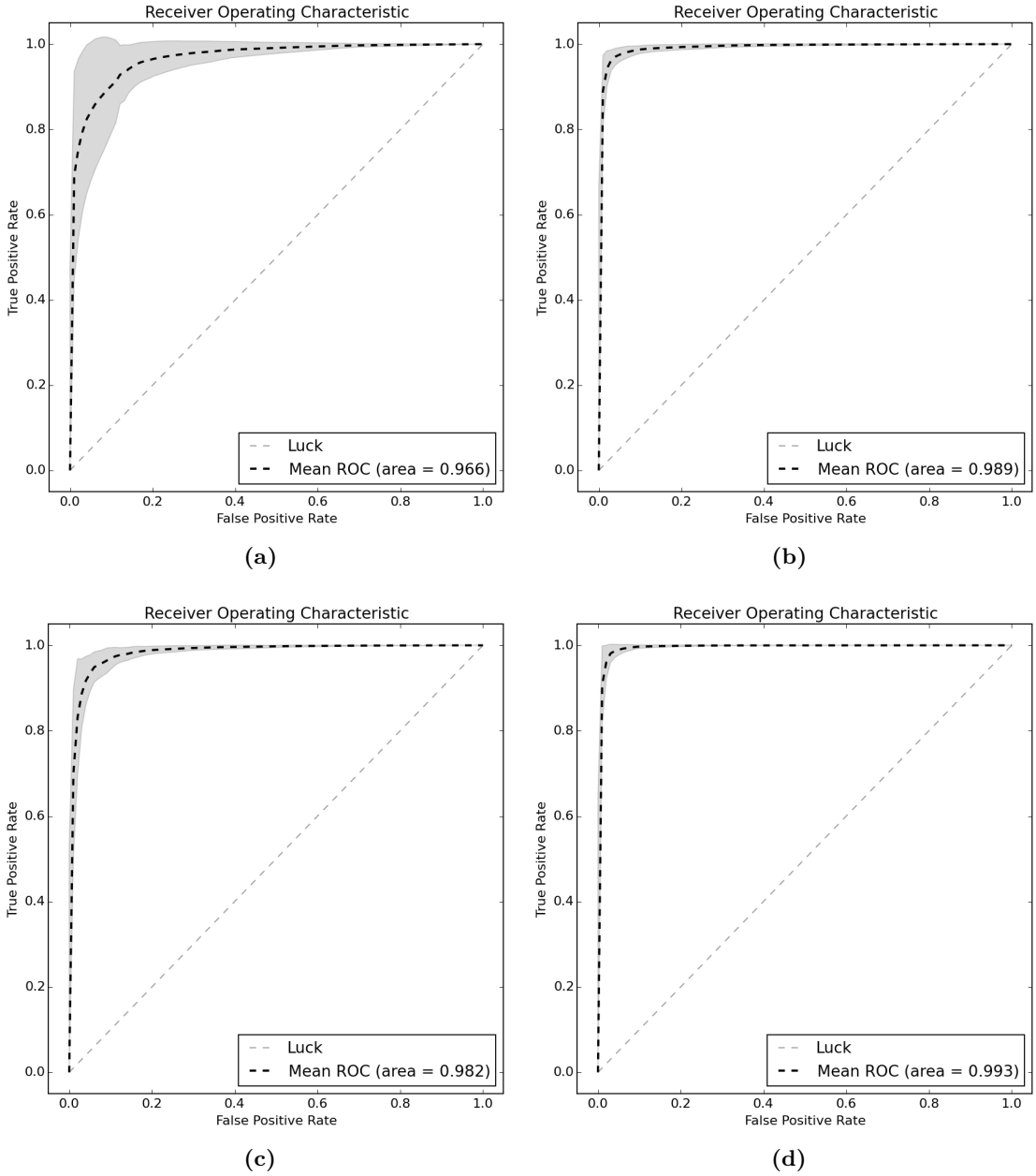



**Figure 5.12:** Leave-one-slide-out cross validation for the full classification (Initial + Refinement) at a pixel spacing of 3.63$\mu$m for (a) BrdU, (b) EvG, (c) GS, and(d) H&E staining. The gray area shows the standard deviation.

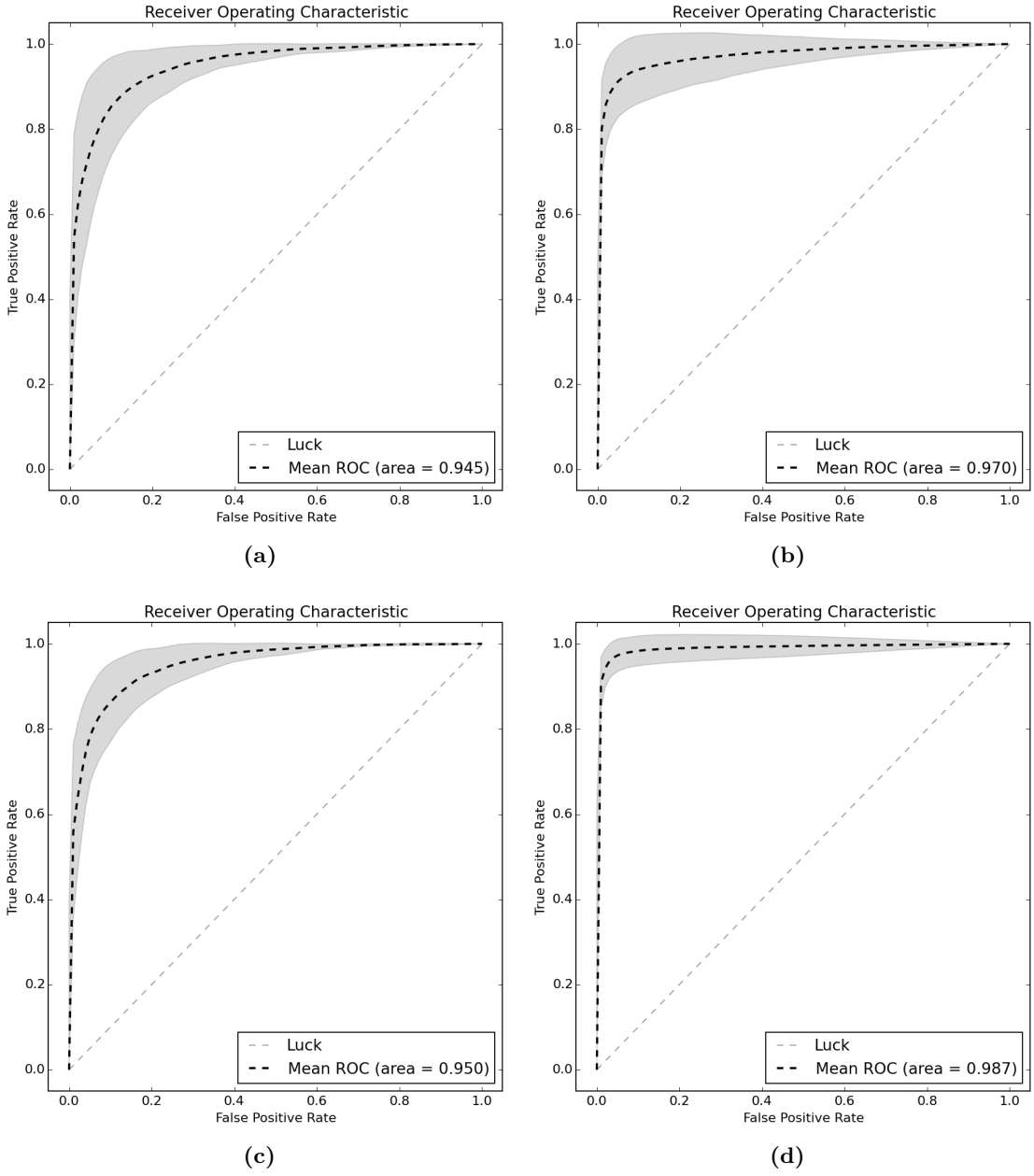



**Figure 5.13:** Leave-one-slide-out cross validation for the full classification (Initial + Refinement) at a pixel spacing of 1.82$\mu$m for (a) BrdU, (b) EvG, (c) GS, and(d) H&E staining. The gray area shows the standard deviation.

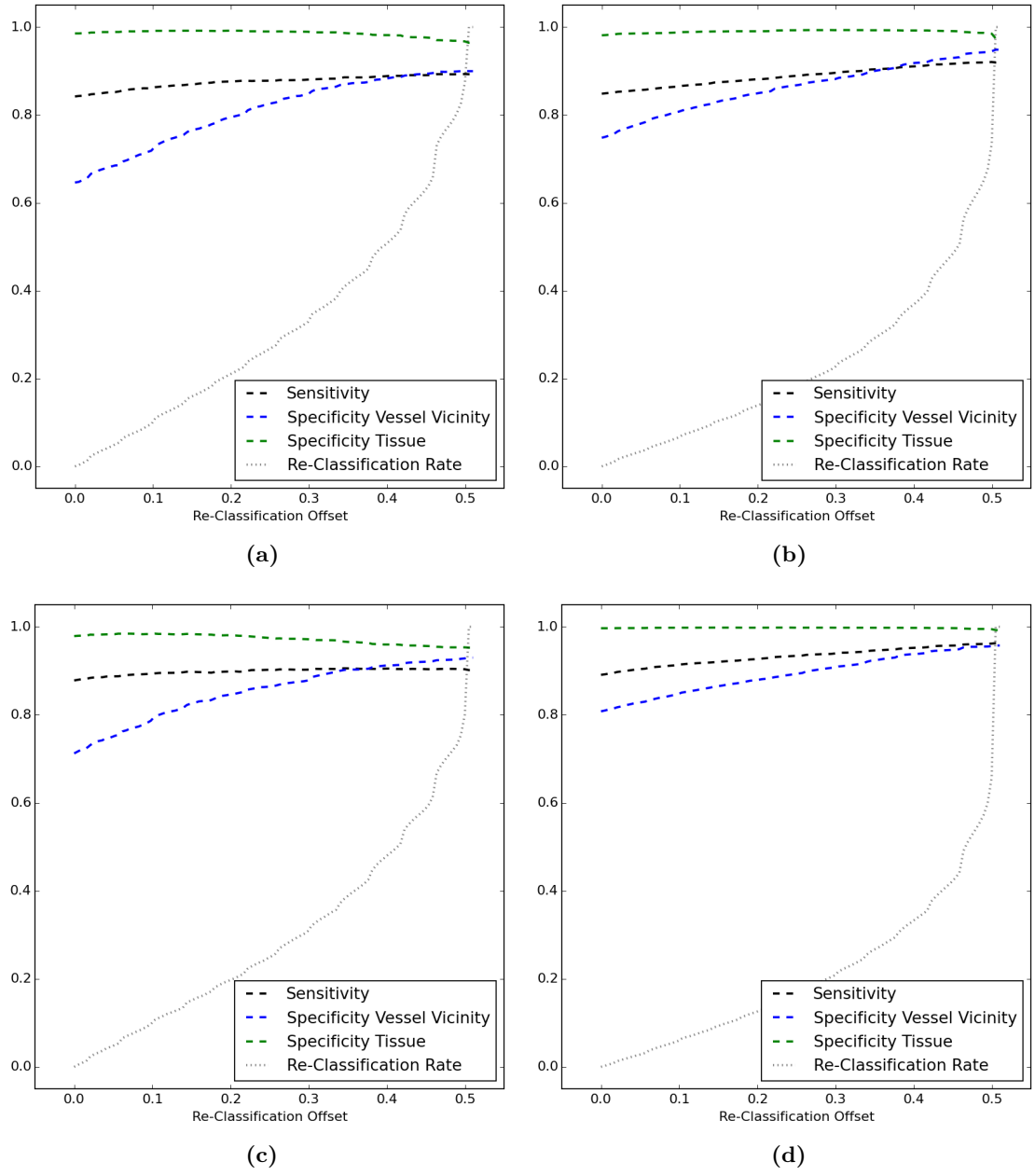
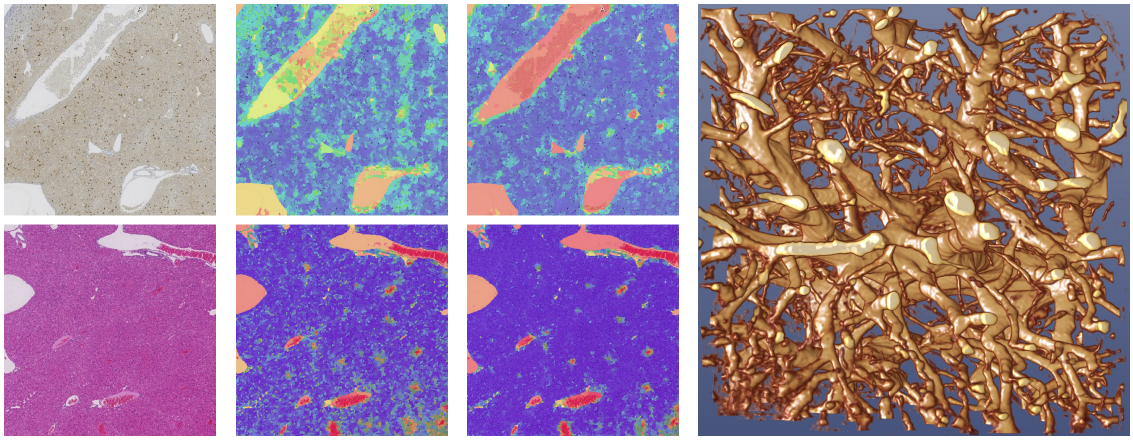



**Figure 5.14:** Evaluation of sensitivity, specificities and rate of required re-classification depending on the re-classification offset for (a) BrdU, (b) EvG, (c) GS, and(d) H&E staining.



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

**Figure 5.15:** (a) Two sample slides (ROI) with corresponding (b) Initial vessel classification result weights overlay (blue/purple = 0.0; red = 1.0) and (c) Refinement vessel classification result weights. (d) 3D rendering of segmented vessels in a ROI of $1000px \times 1000px \times 236slides$ at $3.63\mu$m/px.

the same classification performance as a full re-classification but would only require to recompute less than 40% of all objects.

### 5.2.4 Discussion

We presented a method for detecting and segmenting vessels in stacks of pre-registered serial sections. Our approach has a very high specificity from 95% up to almost 100% in pure tissue, depending on the staining. The detection rate for vessels is also high with rates of 89%–98%, depending on the staining. Specificity along the borders of vessels is reasonably high with rates between 88% and 96%.

Our method is able to automatically and robustly detect vessels in stacks of differently stained histological slides (see figure 5.15d for an example). It can be easily adapted to new stainings and zoom levels by providing appropriate training samples. In this way it could also be trained to detect different tissue structures.

Key elements of our approach are the context features, which are used to gather in-slide information of an object's surroundings. Especially in the second classification phase, using the class weight context of neighboring slides allows the classifier to refine and possibly correct uncertain decisions of the first classification stage. In this way a weaker classifier for a particular staining can profit from more certain decision on neighboring slides.

Our future work would aim at improving the detection rate for vessels and especially for the tissue around the vessel border. One reason for errors is the confusion of residue inside vessels with actual tissue. This could be addressed by not including residue as samples for vessels. We could either add an extra class for residue or just add those

samples as tissue samples. A post classification could then be trained to identify such regions as being located inside a vessel.

Another idea for future work concerns the selection of features for classification. Currently we use the same set of features for all stainings. However, since stainings have such diverse visual appearances, specific sets of features for each staining might improve the classification performance. Therefore, we would like to perform a more thorough evaluation regarding the influence of the features on the classification.

# Chapter 6

# Conclusion

The objective of this thesis was to explore the capabilities and applicability of OBIA for biomedical image analysis.

First the foundation was laid with our own formalization of OBIA in chapter 2, where also a sketch of the efficient and generic implementation was presented. We discussed the general motivation for OBIA as well as basic strategies and prerequisites of applying it to image analysis tasks. Object-based image analysis is not supposed to replace traditional pixel-based image processing. For calculating objects' features or the initial over-segmentation of an image it actually often relies itself on pixel-based methods. Hence OBIA is an extension of conventional methodologies and adds particular strengths. It allows to transform an image into a graph structure with rich attributes describing the objects themselves as well as their relations. OBIA allows for efficiently managing and querying this graph structure, and to build multiple hierarchies within. This enables us to establish complex reasoning processes either based on explicitly formulated rules or through a supervised learning process. Of course, not all the possibilities are always needed in practice. As the application chapters show, OBIA can play a major or minor role in a solution, depending on the particular task.

The core of this thesis are three tasks from the field of biomedical image analysis to which we applied OBIA to reach a solution. They demonstrate the wide applicability of OBIA and show various approaches of how to involve OBIA into a solution for a practical image analysis problem.

The *Spine and Vertebrae Detection in CT* chapter started with an explicit OBIA approach. We tried as humans to fully understand the whole context of the given problem and map it into explicit rules. The OBIA framework proved to effectively allow this. However, we also learned that with increasing complexity it becomes hard to manually grasp all interrelations and thus difficult to handle the explicit approach. Therefore an alternative processing pipeline was proposed as well, which relies on machine learning methods in all steps where the complexity was becoming too overwhelming. This allowed us to compare the results and experiences of two different OBIA approaches on the same task.

In chapter 4 several methods were developed to establish a whole system for the automatic detection of pregnancy in pigs, which is a world novelty. The main contribution is the automatic pregnancy detection in ultrasound images. A random forest classifier was used to find embryonic vesicles in an automatically generated object hierarchy of possible segmentations. A novel and task-specific border feature was presented that helped to significantly improve the detection quality and allowed us to establish an algorithm that is suitable to be applied in practice. The results were confirmed by an independent study conducted under real-world conditions. Two other sub-projects were concerned with analyzing video camera images. One to identify the uterus position on the pig as well as the position of the ultrasound arm, to automatically steer and attach the ultrasound probe. The other to detect dirt on the camera pane. Here OBIA was used as a rather small asset in the processing pipeline, to handle several regions and to apply a small set of specific rules.

Processing histological whole slide images is a rather new topic in computer science, since only the recent development of commercial whole slide scanners allowed the digitization of complete slides of histological samples at microscopic resolution. For image processing the size of such images alone is a challenge. The *Histological Vessel Reconstruction on Murine Liver Samples* was approached in two steps. At first the 2D slide images had to be registered. We presented a novel rigid registration method based on interest point matching. The algorithm is able to robustly register images with only a sparse set of interest points and can still handle a comparably high number of outliers. This allowed us the virtual 3D reconstruction of whole mice livers from histological slides at full resolution, which to our knowledge has not been done before. On such registered stacks the automatic vessel detection is performed using OBIA. Our algorithm relies on machine learning to assess the interrelations of features distinguishing vessel structures from tissue. Among some conventional features also context features are employed. Furthermore a two-step classification scheme is proposed, in which the second step utilizes the previous classification results of the spatial 3D context of an object to refine the classification decision. With this strategy high detection rates could be achieved over a set of different staining types. Besides for the classification itself, OBIA also showed its strengths by its scalability and by enabling multi-scale approaches.

In conclusion from the experiences and results from the considered projects, the proposed formalization and implementation of OBIA allows the effective formulation of explicit rules for implementing reasoning processes. Explicit classification rules are a good way to go for tasks in which the underlying interrelations are clear. This is usually the case if only a small number of features is required to describe and distinguish the structures in question. An explicit approach has the advantage that we know exactly what is happening during the analysis process an thus can also understand failures better.

However, the more features are involved or required for the classification and the more complex their interdependencies are, the more difficult it becomes to properly set up rules manually. One factor is the exponentially growing amount of time required for the manual

analysis. Another factor is the high chance that one would miss an important aspect and thus the classification performance would be worse than it could be.

From the experience with the applications and solutions presented in this thesis we conclude that for such more complex scenarios it is advisable to combine OBIA with a supervised machine learning approach. Especially since the concept of OBIA facilitates the learning of interrelations of features and objects even within different generated object hierarchies. Another advantage of a machine learning approach is that the classifiers can be easily re-trained given new datasets or more promising features. Re-adapting an explicit approach would require a significant amount of time and effort, possibly even as much as the initial development.

Given the solutions of OBIA in combination with trained classifiers in this thesis we can derive a general processing pattern. In the following we will describe a general strategy that we would recommend for anyone who wants to develop an OBIA classification pipeline for any kind of task.

## 6.1 Generic OBIA Classification Strategy

An OBIA classification strategy basically comprises four essential ingredients: We need an appropriate over-segmentation, we need to establish an object hierarchy, we have to find good features, and we need to classify objects.

### 6.1.1 Over-segmentation

The initial over-segmentation is crucial because it defines the basic objects. It must respect the important borders of the final structures to be detected, but it should also not be too fine-grained because too tiny initial objects might not bear enough feature information to build a hierarchy and classification upon. As a researcher or developer this step should not be neglected and different over-segmentation strategies should be experimented with. Several standard segmentation algorithms to be considered were discussed in section 2.2, but developing task-specific methods might also be advantageous (see e. g. section 4.1.2.1). In fact, this step is quite a challenge, since all following steps depend on it. An inappropriate or instable over-segmentation might forfeit the capabilities and advantages of the following actual OBIA steps.

### 6.1.2 Object Hierarchy

If a hierarchy of objects is indeed required depends on the initial segmentation. In some cases it might only be required to identify the right objects among the basic objects. In other cases the structures to be detected have to be assembled from smaller objects. For this an object-merge hierarchy has to be established. The hierarchy can be completely built from the basic object level as part of the classification strategy. It can also be supported by initial guesses of possible hierarchies, which are then refined. Initial guesses

can be generated simply as coarser versions of the initial over-segmentation. They can also be derived from a custom strategy.

### 6.1.3 Feature Selection

The feature selection is the most crucial part of the whole classification strategy. This is where the domain/experts' knowledge has to be captured. Not only should one select good features among the standard features available in OBIA or well known in the community of pattern recognition, but one should also think about designing very problem-specific features that describe attributes specific to the detection problem.

There are approaches that achieve good classification results with standard features or even no explicit input features at all. Lately Deep Learning [70] has caused a lot of attention in the computer vision community and has shown impressive performance. In particular deep convolutional neural network architectures are used, like presented in the paper of Krizhevsky et al. [67] for classifying the ImageNet dataset, which basically triggered the wide adoption of Deep Learning [70]. An intriguing aspect of a convolutional neural network is that it only takes the image itself as input and inherently generates/learns appropriate features. While in the top layers basic low level features are learned, in deeper layers the network also learns to correlate these into more complex features and is thus able to capture context. An investigation of Simonyan and Zisserman [150] supports this, since their results indicate that deeper networks can improve the classification performance. Consequently the latest most successful approaches to the ImageNet classification use very deep networks, like for example the work of He et al. [53], who employ up to 152 layers.

Regarding medical images Ciompi et al. [24] presented a method to classify pre-detected lung nodule candidates. An interesting aspect of their work is that they used the pre-trained convolutional neural network OverFeat [143] to calculate features and then used those features to train an ensemble of new classifiers. In this way features learned on completely different types of images are used in a new domain. With this they basically follow the idea of Razavian et al. [121] who tested this approach on several non-medical image datasets.

The papers mentioned above focus on recognition of image content. Other works have also successfully applied deep convolutional networks for image segmentation. Ciresan et al. [25] presented a method in which they process an image patchwise, classifying the center pixel of a patch to reach a segmentation of the whole image. This is computationally expensive and the limited patch size also limits the learning of contextual features. To overcome this, Ronneberger et al. [129] presented the U-Net architecture in which the typical convolutional network path (contracting path) is followed by an expansive path, which is also connected to corresponding layers in the contracting path. They used their approach for segmentation of neuronal structures in electron microscopy images and cell tracking in transmitted light microscopy images. They also presented a training strategy that uses strong data augmentation to achieve robust segmentation results even with

small sets of only 20–35 annotated reference images. However, whether the latter is transferable to segmentation and classification tasks with more variations and complexity is questionable.

This was only a brief insight into what is currently happening regarding Deep Learning in computer vision and image analysis. Given these successes, Deep Learning should definitely not be ignored for image analysis. However, it must be considered that usually huge amounts of training data are required for a successful training that generalizes well. In many cases this is not easily available, especially in the bio-medical field. Furthermore, some researches have also shown some surprising properties: Szegedy et al. [157] demonstrated adversarial examples that for the human eye are indistinguishable from regular samples, but completely fool Deep Learning methods. Nguyen et al. [103] could create images that practically show only noise, but achieve high-confidence predictions.

Whether features designed by humans will become superfluous in the future cannot be answered yet. We would be cautious with such a claim considering the size of 3D medical images and their complexity, especially since the significant Deep Learning successes are currently made on 2D images. Sophisticated classifications on 3D images would most likely require even deeper networks to capture the required context and as He et al. [53] note: "Deeper neural networks are more difficult to train." They would also require even more training data, which could become an obstacle in practice.

It is also noteworthy that AlphGo—the computer program that plays the board game Go at world class level—also uses some pre-detected game-specific features additional to the raw board positions [149]. While it was considered a breakthrough for artificial intelligence thanks to Deep Learning, AlphaGo is actually a combination of methods, still also relying on some hand-crafted features as well as a Monte Carlo tree search. This could be interpreted as an indication that on the one hand Deep Learning will definitely be a key technology for classification and recognition tasks, but that on the other hand it does not completely replace other methodologies and the incorporation of human domain expertise.

Especially for the analysis of bio-medical images we believe that using domain knowledge to develop a few specific and rather well discriminating features and/or select the best of the standard features is still beneficial. Domain knowledge can also be employed to combine existing features into meta-features that capture certain contextual properties which are known to an expert to be significant for recognition. Incorporating domain knowledge like this will also help to get good training results even from smaller sets of training data, because the classification pipeline does not have to learn everything from scratch. Furthermore, a pure machine learning approach without incorporation of any domain knowledge is generally susceptible to a lack of representative training data, for example if an object to be detected is not captured in all possible contextual environments, or if the negative training samples do not capture the full range of possible scenarios. In such cases a small set of hand-crafted rules and features based on domain knowledge would help to handle this. However, we would agree with Razavian et al. [121], who argue that

generic deep features, like those extracted from OverFeat [143], combined with a simple classifier are a strong baseline. Hence we would consider for the future to incorporate such features into OBIA so they can be easily used. This could possibly make the hard part of finding good features a bit easier.

### 6.1.4 Classification

For the classification any established classifier can be used. Throughout this thesis we relied on Random Forests [14], since they are remarkably easy to parametrize and have shown a competitive performance compared to other well-known classifiers [35, 20].

As we also have shown in this thesis the classification step does not need to be a singular final step. Cascading classifiers can be established. Furthermore classification could also be used intermediately to drive the generation of object merging hierarchies.

Of course for a successful training a set of images with reference labels is required. The more the better and the more of the typical variance among them is captured the better.

## 6.2 Future Work: The OBIA Classification Wizard

Considering the elements of an OBIA classification strategy as described above, we were wondering if it wasn't possible to partly automate the process? The idea would be to feed a set of images with reference labels into a system that automatically establishes a classification pipeline. Of course this could not include individual adaptations to domain-specific knowledge. However, it could provide one or a few suggestions of base versions of a classification scheme from which to start working on task-specific modifications, instead of starting from scratch each time. In the following we will sketch our ideas for such an OBIA Classification Wizard.

### 6.2.1 Over-segmentation

Deriving an appropriate over-segmentation would be based on the given reference segmentations. The system would iteratively try different settings of basic preprocessing (smoothing, edge detection) and standard segmentation methods (see section 2.2). The result would be evaluated by how well an over-segmentation could approximate the borders of the labeled reference structures. As measures the typical ones like Hausdorff distance [51] and Dice coefficient [33] come to mind. Furthermore the number and size of the generated basic objects could be a measure as well.

### 6.2.2 Object Hierarchy

In a similar manner as above, initial segmentation hierarchies could be established by allowing more relaxed criteria and enforcing a larger average object size.

### 6.2.3 Feature Selection

As mentioned before, we believe that the selection of good features is the most important part of the classification strategy. Since it is also the one in which the domain knowledge has to be captured most, we believe this step requires significant manual investigation. However, a first set of features selected from the available standard OBIA features could be established by applying automatic feature selection. See for example Dash and Liu [31] or Liu and Yu [74] for more on this topic.

Furthermore as mentioned in section 6.1.3 we could imagine to incorporate generic deep features which proved to work well on different types of images already [121, 24]. Using deep convolutional neural networks to learn features on the actual given images is also worth considering. However, appropriately setting up and training such a network does also require quite some manual labor.

### 6.2.4 Classification

Since the supervised learning is essentially automatic, the only thing a wizard system could do is trying different types of classifiers or classification cascades. However, if the parametrization is crucial, it could become difficult to set up automatically. Due to its ease of parametrization we would stick with the Random Forest. A classification with a default Random Forest would most likely already indicate if the automatic proposed classification pipeline has potential and then leave it to the user to customize and improve the individual steps.

## 6.3 Final Remarks

This thesis has proposed a formalization and implementation of OBIA and demonstrated its potential and wide applicability for biomedical image analysis. On three quite different and relevant tasks various OBIA approaches have been explored and evaluated. The presented solutions on their own also represent significant contributions, some of them showing results that have not been achieved before and even reaching direct practical application.

# Bibliography


[1] A. Abouelela, H. M. Abbas, H. Eldeeb, A. A. Wahdan, and S. M. Nassar. Automated vision system for localizing structural defects in textile fabrics. *Pattern Recognition Letters*, 26(10):1435–1443, 2005.

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2274–2282, 2012.

[3] S. Aksoy. Modeling of Remote Sensing Image Content Using Attributed Relational Graphs. *Lecture Notes in Computer Science*, 4109:475–483, 2006.

[4] G. W. Almond and G. D. Dial. Pregnancy diagnosis in swine: a comparison of the accuracies of mechanical and endocrine tests with return to estrus. *Journal of the American Veterinary Medical Association*, 189(12):1567, 1986.

[5] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011.

[6] U. Bagci and L. Bai. Automatic best reference slice selection for smooth volume reconstruction of a mouse brain from histological images. *IEEE Transactions on Medical Imaging*, 29(9):1688–96, sep 2010.

[7] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[8] A. C. Berg, T. L. Berg, and J. Malik. Shape Matching and Object Recognition using Low Distortion Correspondences. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005.*, volume 1, pages 26–33, 2005.

[9] P. J. Besl and N. D. McKay. A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[10] T. Blaschke. Object based image analysis for remote sensing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(1):2–16, jan 2010.

[11] O. Boiman and M. Irani. Detecting Irregularities in Images and in Video. *International Journal of Computer Vision*, 74(1):17–31, 2007.

[12] O. Botero, F. Martinat-Botte, and F. Bariteau. Use of ultrasound scanning in swine for detection of pregnancy and some pathological conditions. *Theriogenology*, 26(3):267–278, 1986.

[13] U. D. Braumann, N. Scherf, J. Einenkel, L. C. Horn, N. Wentzensen, M. Loeffler, and J. P. Kuska. Large histological serial sections for computational tissue volume reconstruction. *Methods of Information in Medicine*, 46(5):614–622, 2007.

[14] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

[15] T. Brosnan and D.-W. Sun. Inspection and grading of agricultural and food products by computer vision systems-a review. *Computers and Electronics in Agriculture*, 36:193–213, 2002.

[16] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[17] W. Burger and M. J. Burge. *Principles of Digital Image Processing: Core Algorithms*. Springer, London, 2009.

[18] J. Carballido-Gamio, S. J. Belongie, and S. Majumdar. Normalized cuts in 3-D for spinal MRI segmentation. *IEEE Transactions on Medical Imaging*, 23(1):36–44, 2004.

[19] G. Carneiro, B. Georgescu, S. Good, and D. Comaniciu. Detection and measurement of fetal anatomies from ultrasound images using a constrained probabilistic boosting tree. *IEEE Transactions on Medical Imaging*, 27(9):1342–1355, 2008.

[20] R. Caruana and A. Niculescu-Mizil. An Empirical Comparison of Supervised Learning Algorithms. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 161–168, 2006.

[21] N.-S. Chang and K.-S. Fu. A relational database system for images. In *Pictorial Information Systems*, pages 288–321. Springer, 1979.

[22] H. Chui and A. Rangarajan. A New Point Matching Algorithm for Non-rigid Registration. *Computer Vision and Image Understanding*, 89(2-3):114–141, 2003.

[23] A. Cifor, T. Pridmore, and A. Pitiot. Smooth 3-D reconstruction for 2-D histological images. In *Proc. Information Processing in Medical Imaging*, volume 21, pages 350–61, jan 2009.

[24] F. Ciompi, B. de Hoop, S. J. van Riel, K. Chung, E. T. Scholten, M. Oudkerk, P. A. de Jong, M. Prokop, and B. van Ginneken. Automatic classification of pulmonary peri-fissural nodules in computed tomography using an ensemble of 2D views and a convolutional neural network out-of-the-box. *Medical Image Analysis*, 26(1):195–202, 2015.

[25] D. C. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images. In *Conference on Neural Information Processing Systems*, pages 2852–2860, 2012.

[26] J. J. Corso, R. S. Alomari, and V. Chaudhary. Lumbar Disc Localization and Labeling with a Probabilistic Model on both Pixel and Object Features. In *International Conference on Medical Image Computing and Computer-Assisted Intervention - MICCAI*, volume 11, pages 202–210. Springer, 2008.

[27] A. D. J. Cross and E. R. Hancock. Graph Matching With a Dual-Step EM Algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1236–1253, 1998.

[28] J. L. Crowley and R. M. Stern. Fast Computation of the Difference of Low-Pass Transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(2):212–222, mar 1984.

[29] J. E. Cutting. *Why our stimuli look as they do.* Washington, DC, US: American Psychological Association, 1991.

[30] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005.*, volume 1, pages 886–893, 2005.

[31] M. Dash and H. Liu. Feature Selection for Classification. *Intelligent Data Analysis*, 1(1):131–156, 1997.

[32] J. Dauguet, T. Delzescaux, F. Condé, J.-F. Mangin, N. Ayache, P. Hantraye, and V. Frouin. Three-dimensional reconstruction of stained histological slices and 3D non-linear registration with in-vivo MRI for whole baboon brain. *Journal of Neuroscience Methods*, 164(1):191–204, aug 2007.

[33] L. R. Dice. Measures of the Amount of Ecologic Association Between Species. *Ecology*, 26(3):297–302, 1945.

[34] M. Enzweiler and D. M. Gavrila. Monocular Pedestrian Detection: Survey and Experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2179–2195, 2009.

[35] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim. Do we Need Hundreds of Classifiers to Solve Real World Classification Problems? *Journal of Machine Learning Research*, 15:3133–3181, 2014.

[36] M. Feuerstein, H. Heibel, J. Gardiazabal, N. Navab, and M. Groher. Reconstruction of 3-D histology images by simultaneous deformable registration. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, volume 14, pages 582–9, jan 2011.

[37] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[38] W. Flowers, J. Armstrong, S. White, T. Woodard, and G. Almond. Real-time ultrasonography and pregnancy diagnosis in swine. *Journal of Animal Science*, 77(E-Suppl):1–9, 2000.

[39] W. Flowers, R. Knox, and W. Singleton. Pregnancy Diagnosis in Swine. Technical report, U.S. Pork Center of Excellence, 2006.

[40] J. Funke, B. Andres, F. a. Hamprecht, A. Cardona, and M. Cook. Efficient Automatic 3D-Reconstruction of Branching Neurons from EM Data. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1004–1011, 2012.

[41] D. Geronimo, A. Lopez, A. Sappa, and T. Graf. Survey of Pedestrian Detection for Advanced Driver Assistance Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(7):1239–1258, 2010.

[42] S. Ghebreab and A. W. M. Smeulders. Combining Strings and Necklaces for Interactive Three-Dimensional Segmentation of Spinal Images Using an Integral Deformable Spine Model. *IEEE Transactions on Biomedical Engineering*, 51(10):1821–1829, 2004.

[43] J. C. Goddard, C. D. Sutton, P. N. Furness, R. C. Kockelbergh, and K. J. O. Byrne. A computer image analysis system for microvessel density measurement in solid tumours. *Angiogenesis*, 5(1):15–20, 2002.

[44] C. Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 285—-339, 1991.

[45] R. L. Graham. An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set. *Information Processing Letters*, 1(4):132–133, 1972.

[46] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, pages 47–57. ACM New York, NY, USA, 1984.

[47] E. Haber and J. Modersitzki. Intensity Gradient Based Registration and Fusion of Multi-modal Images. In C. Barillot, D. R. Haynor, and P. Hellier, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2006*, volume 3216, pages 591–598. Springer LNCS, 2006.

[48] H. K. Hahn and H.-O. Peitgen. IWT - Interactive Watershed Transform: A hierarchical method for efficient interactive and automated segmentation of multidimensional gray-scale images. In *Proc. SPIE Medical Imaging*, volume 5032, pages 643–653, 2003.

[49] C. W. Hanna and A. B. M. Youssef. Automated Measurements in Obstetric Ultrasound Images. In *International Conference on Image Processing*, pages 504–507, 1997.

[50] C. Harris and M. Stephens. A Combined Corner and Edge Detector. *Procedings of the Alvey Vision Conference 1988*, pages 147–151, 1988.

[51] F. Hausdorff. *Grundzüge der Mengenlehre*. Veit & Comp, 1914.

[52] G. J. Hay and G. Castilla. Object-based image analysis: strengths, weaknesses, opportunities and threats (SWOT). In S. Lang, T. Blaschke, and E. Schöpfer, editors, *1st International Conference on Object-based Image Analysis (OBIA 2006)*, 2006.

[53] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[54] H. v. Helmholtz. Handbuch der physiologischen Optik. *Leipzig: Leopold Voss*, 1867.

[55] J. L. Herring and B. M. Dawant. Automatic lumbar vertebral identification using surface-based registration. *Journal of Biomedical Informatics*, 34(2):74–84, apr 2001.

[56] G. E. Hinton, C. K. I. Williams, and M. D. Revow. Adaptive Elastic Models for Hand-Printed Character Recognition. In J. Moody, S. Hanson, and R. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 512–519. Morgan-Kaufmann, 1992.

[57] A. Homeyer, A. Schenk, J. Arlt, U. Dahmen, and H. K. Hahn. Fast and accurate identification of fat droplets in histological images. *Computer Methods and Programs in Biomedicine*, 121(2):59–65, 2015.

[58] A. Homeyer, M. Schwier, and H. K. Hahn. A Generic Concept for Object-Based Image Analysis. In P. Richard and J. Braz, editors, *Proc. International Conference on Computer Vision Theory and Applications*, volume 2, pages 530–533. INSTICC Press, 2010.

[59] W. Hu, T. Tan, L. Wang, and S. Maybank. A Survey on Visual Surveillance of Object Motion and Behaviors. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 34(3):334–352, 2004.

[60] K. Huang, L. Cooper, A. Sharma, and T. Pan. Fast Automatic Registration Algorithm for Large Microscopy Images. In *Life Science Systems and Applications Workshop*, number 1, pages 1–2, 2006.

[61] S. H. Huang, Y. H. Chu, S. H. Lai, and C. L. Novak. Learning-Based Vertebra Detection and Iterative Normalized-Cut Segmentation for Spinal MRI. *IEEE Transactions on Medical Imaging*, 28(10):1595–1605, 2009.

[62] S.-H. Huang, S.-H. Lai, and C. L. Novak. A statistical learning appproach to vertebra detection and segmentation from spinal MRI. In *5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2008. ISBI 2008.*, pages 125–128, 2008.

[63] S. V. B. Jardim and M. A. T. Figueiredo. Automatic contour estimation in fetal ultrasound images. In *ICIP 2003 - International Conference on Image Processing*, pages 1065–1068, 2003.

[64] T. Klinder, J. Ostermann, M. Ehm, A. Franz, R. Kneser, and C. Lorenz. Automated model-based vertebra detection, identification, and segmentation in CT images. *Medical Image Analysis*, 13(3):471–482, jun 2009.

[65] R. Knox, W. Flowers, T. Safranski, and W. Singleton. Using real-time ultrasound for pregnancy diagnosis in swine. Technical report, U.S. Pork Center of Excellence, 2006.

[66] L. König and J. Rühaak. A fast and accurate parallel algorithm for non-linear image registration using Normalized Gradient fields. In *2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI)*, pages 580–583, 2014.

[67] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105, 2012.

[68] A. Kumar. Computer-Vision-Based Fabric Defect Detection: A Survey. *IEEE Transactions on Industrial Electronics*, 55(1):348–363, 2008.

[69] S. Lang and D. Tiede. Definiens Developer. *GIS Business*, pages 34–37, sep 2007.

[70] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[71] S.-C. Lee and P. Bajcsy. Three-dimensional volume reconstruction based on trajectory fusion from confocal laser scanning microscope images. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2221–2228. IEEE, 2006.

[72] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi. TurboPixels: Fast Superpixels Using Geometric Flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2290–2297, 2009.

[73] J. Lindblad. Surface area estimation of digitized 3D objects using weighted local configurations. *Image and Vision Computing*, 23(2):111–122, 2005.

[74] H. Liu and L. Yu. Algorithms for Classification and Clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):491–502, 2005.

[75] W. E. Lorensen and H. E. Cline. Marching Cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Computer Graphics*, 21(4):163–169, 1987.

[76] J. Lotz, J. Berger, B. Müller, K. Breuhahn, N. Grabe, S. Heldmann, A. Homeyer, B. Lahrmann, H. Laue, J. Olesch, M. Schwier, O. Sedlaczek, and A. Warth. Zooming in: high resolution 3D reconstruction of differently stained histological whole slide images. In *Proc. SPIE Medical Imaging*, volume 9041, page 904104, mar 2014.

[77] J. Lotz, J. Olesch, B. Müller, T. Polzin, P. Galuschka, J. M. Lotz, S. Heldmann, H. Laue, M. González-Vallinas, A. Warth, B. Lahrmann, N. Grabe, O. Sedlaczek, K. Breuhahn, and J. Modersitzki. Patch-Based Nonlinear Image Registration for Gigapixel Whole Slide Images. *IEEE Transactions on Biomedical Engineering*, PP(99):1–9, 2015.

[78] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157, 1999.

[79] H. Luukkaa, J. Laitakari, T. Vahlberg, P. Klemi, and R. Grénman. Morphometric analysis using automated image analysis of CD34-positive vessels in salivary gland acinic cell carcinoma. *Acta Oto-Laryngologica*, 127(8):869–873, 2007.

[80] D. Magee, S. Gilbert, A. Bulpitt, Y. Song, N. Roberts, N. Wijayathunga, R. Wilcox, and D. Treanor. Histopathology in 3D: From three-dimensional reconstruction to multi-stain and multi-modal analysis. *Journal of Pathology Informatics*, 6(1):6, 2015.

[81] M. Mahesh and M. V. Subramanyam. Invariant Corner Detection Using Steerable Filters and Harris Algorithm. *Signal & Image Processing*, 3(5), 2012.

[82] N. Maillot, M. Thonnat, and A. Boucher. Towards ontology-based cognitive vision. *Machine Vision and Applications*, 16(1):33–40, 2004.

[83] M. A. Maraci, R. Napolitano, A. Papageorghiou, and J. A. Noble. Fetal Head Detection on Images from a Low-Cost Portable USB Ultrasound Device. Technical report, University of Oxford, 2013.

[84] D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* Henry Holt and Co., Inc., New York, NY, USA, 1982.

[85] A. Mastmeyer, K. Engelke, C. Fuchs, and W. A. Kalender. A hierarchical 3D segmentation method and the definition of vertebral body coordinate systems for QCT of the lumbar spine. *Medical Image Analysis*, 10(4):560–577, aug 2006.

[86] G. K. Matsopoulos and S. Marshall. Use of morphological image processing techniques for the measurement of a fetal head from ultrasound images. *Pattern Recognition*, 27(10):1317–1324, 1994.

[87] H. Meine. *The GeoMap Representation: On Topologically Correct Sub-pixel Image Analysis.* Phd thesis, University of Hamburg, 2009.

[88] K. Mertens, B. De Ketelaere, B. Kamers, F. R. Bamelis, B. J. Kemps, E. M. Verhoelst, J. G. De Baerdemaeker, and E. M. Decuypere. Dirt Detection on Brown Eggs by Means of Color Computer Vision. *Poultry Science*, 84(10):1653–1659, 2005.

[89] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. YALE (now: RapidMiner): Rapid Prototyping for Complex Data Mining Tasks. In *Proceedings of the ACM SIGKDDInternational Conference on Knowledge Discovery and Data Mining(KDD 2006)*, pages 935–940. ACM Press, 2006.

[90] J. Modersitzki. *FAIR: Flexible Algorithms for Image Registration.* SIAM, Philadelphia, 2009.

[91] J. Modersitzki, O. Schmitt, and S. Wirtz. Registration of Histological Serial Sectionings. In *Mathematical Models for Registration and Applications to Medical Imaging*, pages 63–80. 2006.

[92] T. B. Moeslund and E. Granum. A Survey of Computer Vision-Based Human Motion Capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.

[93] A. P. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones. Superpixel Lattices. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008.*, pages 1–8, 2008.

[94] G. Mori. Guiding model search using segmentation. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005.*, volume 2, pages 1417–1423, 2005.

[95] G. Mori, S. Belongie, and J. Malik. Efficient Shape Matching Using Shape Contexts. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 11:1832–1837, 2005.

[96] K. Mosaliganti, T. Pan, R. Sharp, R. Ridgway, S. Iyengar, A. Gulacy, P. Wenzel, A. de Bruin, R. Machiraju, K. Huang, G. Leone, and J. Saltz. Registration and 3d visualization of large microscopy images. In J. M. Reinhardt and J. P. W. Pluim, editors, *Proc. SPIE Medical Imaging*, volume 6144, pages 61442V—-1, mar 2006.

[97] D. Moura, J. Barbosa, J. M. R. S. Tavares, and A. Reis. Calibration of bi-planar radiography with a rangefinder and a small calibration object. *Advances in Visual Computing*, pages 572–581, 2008.

[98] D. C. Moura, J. G. Barbosa, A. M. Reis, and J. M. R. S. Tavares. A flexible approach for the calibration of biplanar radiography of the spine on conventional radiological systems. *Computer Modeling in Engineering & Sciences*.

[99] D. C. Moura, J. Boisvert, J. G. Barbosa, H. Labelle, and J. M. R. S. Tavares. Fast 3D reconstruction of the spine from biplanar radiographs using a deformable articulated model. *Medical Engineering & Physics*, 33(8):924–933, 2011.

[100] D. C. Moura, J. Boisvert, J. G. Barbosa, and J. M. R. S. Tavares. Fast 3d reconstruction of the spine using user-defined splines and a statistical articulated model. *Lecture Notes in Computer Science (LNCS)*, 5875:586.

[101] D. C. Moura, J. Boisvert, J. G. Barbosa, J. M. R. S. Tavares, and H. Labelle. Fast 3D reconstruction of the spine by non-expert users using a statistical articulated model. In *Research into Spinal Deformities 7*, Studies in Health Technology and Informatics.

[102] A. Neubert, J. Fripp, K. Shen, O. Salvado, R. Schwarz, L. Lauer, C. Engstrom, and S. Crozier. Automated 3D Segmentation of Vertebral Bodies and Intervertebral Discs from MRI. In *International Conference on Digital Image Computing Techniques and Applications (DICTA)*, pages 19–24, dec 2011.

[103] A. Nguyen, J. Yosinski, and J. Clune. Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 427–436, 2015.

[104] D. Ni, Y. Yang, S. Li, J. Qin, S. Ouyang, T. Wang, and P. A. Heng. Learning based automatic head detection and measurement from fetal ultrasound images via prior knowledge and imaging parameters. In *2013 IEEE 10th International Symposium on Biomedical Imaging (ISBI)*, pages 772–775, 2013.

[105] J. Nithya and M. Madheswaran. Detection of Intrauterine Growth Retardation Using Fetal Abdominal Circumference. In *ICCTD '09. International Conference on Computer Technology and Development, 2009.*, pages 371–375, 2009.

[106] J. A. Noble and D. Boukerroui. Ultrasound Image Segmentation: A Survey. *IEEE Transactions on Medical Imaging*, 25(8):987–1010, 2006.

[107] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[108] M. L. Onozato, V. E. Klepeis, Y. Yagi, and M. Mino-Kenudson. A role of three-dimensional (3D)-reconstruction in the classification of lung adenocarcinoma. *Analytical Cellular Pathology*, 35(2):79–84, 2011.

[109] S. Ourselin, A. Roche, and G. Subsol. Reconstructing a 3D structure from serial histological sections. *Image and Vision Computing*, 19(2000):25–31, 2001.

[110] S. E. Palmer. *Vision Science: Photons to Phenomenology.* The MIT press, 1999.

[111] S. Paris and F. Durand. A topological approach to hierarchical segmentation using mean shift. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07.*, pages 1–8. IEEE, 2007.

[112] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[113] B. Peng, L. Zhang, and D. Zhang. Automatic Image Segmentation by Dynamic Region Merging. *IEEE Transactions on Image Processing*, 20(12):3592–3605, 2011.

[114] Y. Peng, Y. Jiang, L. Eisengart, M. A. Healy, F. H. Straus, and X. J. Yang. Computer-aided identification of prostatic adenocarcinoma: Segmentation of glandular structures. *Journal of Pathology Informatics*, 2(1):33, 2011.

[115] Z. Peng, J. Zhong, W. Wee, and J.-h. Lee. Automated Vertebra Detection and Segmentation from the Whole Spine MR Images. In *IEEE-EMBS 2005. 27th Annual International Conference of the Engineering in Medicine and Biology Society, 2005.*, volume 3, pages 2527–2530. Electr. & Comput. Eng. & Comput. Sci., Cincinnati Univ., OH., 2005.

[116] E. G. M. Petrakis, C. Faloutsos, and K. I. Lin. ImageMap: An Image Indexing Method Based on Spatial Similarity. *IEEE Transactions on Knowledge and Data Engineering*, 14(5):979–987, 2002.

[117] T. Polzin, J. Rühaak, R. Werner, H. Handels, and J. Modersitzki. Lung Registration Using Automatically Detected Landmarks. *Methods of Information in Medicine*, 53(4):250–256, 2014.

[118] F. Preteux. On a Distance Function Approach for Gray-Level Mathematical Morphology: Connection Cost: Definition and Properties. In E. R. Dougherty, editor, *Mathematical Morphology in Image Processing*, chapter 10, pages 335–337. CRC Press, 1992.

[119] S. Ramanujan. Modular equations and approximations to pi. In G. H. Hardy, P. V. Seshu Aiyar, and B. M. Wilson, editors, *Collected Papers of Srinivasa Ramanujan*, chapter 6, pages 23–39. AMS Chelsea Publishing, Providence, Rhode Island, 1962.

[120] A. Rangarajan, H. Chui, and F. Bookstein. The softassign Procrustes matching algorithm. In J. Duncan and G. Gindi, editors, *Information Processing in Medical Imaging*, volume 1230 of *Lecture Notes in Computer Science*, pages 29–42. Springer Berlin Heidelberg, 1997.

[121] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN Features off-the-shelf: an Astounding Baseline for Recognition. In *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 512–519, 2014.

[122] N. J. Redding, D. J. Crisp, D. Tang, and G. N. Newsam. An efficient algorithm for Mumford-Shah segmentation and its application to SAR imagery. In *Proc. Conference on Digital Image Computing: Techniques and Applications*, pages 35–41, 1999.

[123] X. Ren and J. Malik. Learning a classification model for segmentation. In *Proceedings. Ninth IEEE International Conference on Computer Vision, 2003.*, pages 10 –17 vol.1, 2003.

[124] A. Renouf, R. Clouard, and M. Revenu. How to formulate image processing applications. In *Proc. International Conference on Computer Vision Systems*, 2007.

[125] C. C. Reyes-Aldasoro, L. J. Williams, S. Akerman, C. Kanthou, and G. M. Tozer. An automatic algorithm for the segmentation and morphological analysis of microvessels in immunostained histological tumour sections. *Journal of Microscopy*, 242(3):262–278, 2011.

[126] F. Ritter, T. Boskamp, A. Homeyer, H. Laue, M. Schwier, F. Link, and H.-O. Peitgen. Medical Image Analysis: A Visual Approach. *IEEE Pulse*, 2(6):60–70, 2011.

[127] N. Roberts, D. Magee, Y. Song, K. Brabazon, M. Shires, D. Crellin, N. M. Orsi, R. Quirke, P. Quirke, and D. Treanor. Toward routine use of 3D histopathology as a research tool. *The American Journal of Pathology*, 180(5):1835–42, may 2012.

[128] R. Rodríguez, T. E. Alarcón, and J. J. Abad. Blood vessel segmentation via neural network in histological images. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 36(4):451–465, 2003.

[129] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234—-241, 2015.

[130] S. Rueda, S. Fathima, C. Knight, M. Yaqub, A. Papageorghiou, B. Rahmatullah, A. Foi, M. Maggioni, A. Pepe, J. Tohka, R. V. Stebbing, J. E. McManigle, A. Ciurte, X. Bresson, M. B. Cuadra, C. Sun, G. V. Ponomarev, M. S. Gelfand, M. D. Kazanov, C. Wang, H. Chen, C. Hung, and J. A. Noble. Evaluation and Comparison of Current Fetal Ultrasound Image Segmentation Methods for Biometric Measurements: A Grand Challenge. *IEEE Transactions on Medical Imaging*, PP(99):1, 2013.

[131] Y. Sato, S. Nakajima, N. Shiraga, H. Atsumi, S. Yoshida, T. Koller, G. Gerig, and R. Kikinis. Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. *Medical Image Analysis*, 2(2):143–168, 1998.

[132] S. Schmidt, J. Kappes, M. Bergtholdt, V. Pekar, S. Dries, D. Bystrov, and C. Schnörr. Spine Detection and Labeling Using a Parts-Based Graphical Model. In N. Karssemeijer and B. Lelieveldt, editors, *Information Processing in Medical Imaging*, volume 4584 of *Lecture Notes in Computer Science*, pages 122–133. Springer Berlin / Heidelberg, 2007.

[133] O. Schmitt, J. Modersitzki, S. Heldmann, S. Wirtz, and B. Fischer. Image Registration of Sectioned Brains. *International Journal of Computer Vision*, 73(1):5–39, sep 2006.

[134] L. O. Schwen, A. Homeyer, M. Schwier, U. Dahmen, O. Dirsch, A. Schenk, L. Kuepfer, T. Preusser, and A. Schenk. Zonated quantification of steatosis in an entire mouse liver. *Computers in Biology and Medicine*, 73:108–118, 2016.

[135] M. Schwier, T. Böhler, H. K. Hahn, U. Dahmen, and O. Dirsch. Registration of histological whole slide images guided by vessel structures. *Journal of Pathology Informatics*, 4(Suppl 10):1–8, jan 2013.

[136] M. Schwier, T. Chitiboi, L. Bornemann, and H. K. Hahn. An Object-based Image Analysis Approach to Spine Detection in CT Images. In *Proceedings of the III ECCOMAS Thematic Conference on Computational Vision and Medical Image Processing: VipIMAGE 2011*, pages 173–178, 2011.

[137] M. Schwier, T. Chitiboi, T. Hülnhagen, and H. K. Hahn. Automated spine and vertebrae detection in CT images using object-based image analysis. *International Journal for Numerical Methods in Biomedical Engineering*, 29(9):938–963, 2013.

[138] M. Schwier and H. K. Hahn. Segmentation Hierarchies and Border Features for Automatic Pregnancy Detection in Porcine Ultrasound. In *2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI)*, pages 931–934, 2014.

[139] M. Schwier, H. K. Hahn, U. Dahmen, and O. Dirsch. Reconstruction of Vessel Structures from Serial Whole Slide Sections of Murine Liver Samples. In *Proc. SPIE Medical Imaging*, volume 8676, page 86760D, mar 2013.

[140] M. Schwier, H. K. Hahn, U. Dahmen, and O. Dirsch. Segmentation of Vessel Structures in Serial Whole Slide Sections using Region-based Context Features. In *Proc. SPIE Medical Imaging*, page 97910E, 2016.

[141] M. Schwier, J. H. Moltz, and H.-O. Peitgen. Object-based analysis of CT images for automatic detection and segmentation of hypodense liver lesions. *International Journal of Computer Assisted Radiology and Surgery*, 6(6):737–747, 2011.

[142] D. Selle, B. Preim, A. Schenk, and H.-O. Peitgen. Analysis of Vasculature for Liver Surgical Planning. *IEEE Transactions on Medical Imaging*, 21(11):1344–1357, 2002.

[143] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. In *International Conference on Learning Representations (ICLR 2014)*, pages 1–16, 2014.

[144] A. K. Shackelford and C. H. Davis. A combined fuzzy pixel-based and object-based approach for classification of high-resolution multispectral data over urban areas. *IEEE Transactions on GeoScience and Remote Sensing*, 41(10):2354–2363, 2003.

[145] Y. Sharma, R. a. Moffitt, T. H. Stokes, Q. Chaudry, and M. D. Wang. Feasibility analysis of high resolution tissue image registration using 3-D synthetic data. *Journal of Pathology Informatics*, 2:S6, jan 2011.

[146] Y. Shen, J. Yu, Y. Shen, and Y. Wang. Fetal Skull Analysis in Ultrasound Images Based on Iterative Randomized Hough Transform. In *SPIE Medical Imaging*, volume 7265, pages 72650F–72650F–9, 2009.

[147] J. Shi and C. Tomasi. Good features to track. In *Proceedings CVPR '94., 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1994.*, pages 593–600, jun 1994.

[148] R. Shi, D. Sun, Z. Qiu, and K. L. Weiss. An Efficient Method for Segmentation of MRI Spine Images. In *International Conference on Complex Medical Engineering, 2007. CME 2007. IEEE/ICME*, pages 713–717, 2007.

[149] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7585):484–489, 2016.

[150] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR 2015)*, pages 1–14, 2015.

[151] A. R. Smith. Color Gamut Transform Pairs. In *SIGGRAPH '78: Proceedings of the 5th Annual Conference on Computer Graphics and Interactive Techniques*, pages 12–19, 1978.

[152] S. M. Smith and J. M. Brady. SUSAN - A New Approach to Low Level Image Processing. *International Journal of Computer Vision*, 23(1):45–78, 1997.

[153] P. Soille. Discrete Geometry: Grids and connectivity. In *Morphological Image Analysis: Principals and Applications*, chapter 2, pages 35–28. Springer, second edition, 2003.

[154] P. Soille. Geodesic Transformations: Fillhole. In *Morphological Image Analysis: Principals and Applications*, chapter 6, page 208. Springer, second edition, 2003.

[155] Y. Song, D. Treanor, A. J. Bulpitt, and D. R. Magee. 3D reconstruction of multiple stained histology images. *Journal of Pathology Informatics*, 4(Suppl):S7, jan 2013.

[156] B. Steiniger, M. Bette, and H. Schwarzbach. The Open Microcirculation in Human Spleens : A Three-Dimensional Approach. *Journal of Histochemistry & Cytochemistry*, 59(6):639–648, 2011.

[157] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv:1312.6199*, 2013.

[158] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010.

[159] Y. Tan, J. Hua, and M. Dong. Feature curve-guided volume reconstruction from 2D images. In *Proc. ISBI*, pages 716–719, 2007.

[160] Z. Tang and J. Pauli. Fully automatic extraction of human spine curve from MR images using methods of efficient intervertebral disk extraction and vertebra registration. *International Journal of Computer Assisted Radiology and Surgery*, 6(1):21–33, 2011.

[161] T. Tasdizen, P. Koshevoy, B. C. Grimm, J. R. Anderson, B. W. Jones, C. B. Watt, R. T. Whitaker, and R. E. Marc. Automatic mosaicking and volume assembly for

high-throughput serial-section transmission electron microscopy. *Journal of Neuroscience Methods*, 193(1):132–144, oct 2010.

[162] M. A. M. Taverne, L. Oving, M. Van Lieshout, and A. H. Willemse. Pregnancy diagnosis in pigs: A field study comparing linear-array real-time ultrasound scanning and amplitude depth analysis. *Veterinary Quarterly*, 7(4):271–276, 1985.

[163] A. Torralba. Contextual Priming for Object Detection. *International Journal of Computer Vision*, 53(2):169–191, 2003.

[164] G. Toussaint. Solving Geometric Problems with the Rotating Calipers. In *In Proc. IEEE MELECON '83*, 1983.

[165] S. E. Umbaugh. Computer Imaging Systems: Image Representation. In *Digital Image Processing and Analysis: Human and Computer Vision with VCIPtools*, chapter 2, pages 50–65. CRC Press, second edition, 2010.

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

[167] P. Viola and M. J. Jones. Robust Real-Time Face Detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[168] P. Viola, M. J. Jones, and D. Snow. Detecting Pedestrians Using Patterns of Motion and Appearance. *International Journal of Computer Vision*, 63(2):153–161, 2005.

[169] W. M. Wells III. Statistical Approaches to Feature-Based Object Recognition. *International Journal of Computer Vision*, 21:63–98, 1997.

[170] S. I. Williams, P. Piñeyro, and R. L. de la Sota. Accuracy of pregnancy diagnosis in swine by ultrasonography. *The Canadian Veterinary Journal*, 49(3):269, 2008.

[171] S. Wirtz, B. Fischer, J. Modersitzki, and O. Schmitt. Super-fast elastic registration of histologic images of a whole rat brain for three-dimensional reconstruction. In *Proc. SPIE Medical Imaging*, pages 328–334, 2004.

[172] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann, 2005.

[173] M. Wübbels. *Evaluierung einer automatischen Trächtigkeitskontrolle bei Sauen*. Master thesis, Christian-Albrechts-Universität zu Kiel, 2013.

[174] X. Xie. A Review of Recent Advances in Surface Defect Detection using Texture analysis Techniques. *Electronic Letters on Computer Vision and Image Analysis*, 7(3):1–22, 2008.

[175] M.-H. Yang, D. J. Kriegman, and N. Ahuja. Detecting Faces in Images : A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

[176] J. Yao, S. D. O'Connor, and R. M. Summers. Automated spinal column extraction and partitioning. In *3rd IEEE International Symposium on Biomedical Imaging: Nano to Macro, 2006.*, pages 390–393, apr 2006.

[177] A. Yilmaz, O. Javed, and M. Shah. Object Tracking: A Survey. *ACM Computing Surveys (CSUR)*, 38(4):13, 2006.

[178] P. A. Yushkevich, B. B. Avants, L. Ng, M. Hawrylycz, P. D. Burstein, H. Zhang, and J. C. Gee. 3D Mouse Brain Reconstruction from Histology Using a Coarse-to-fine Approach. In *Proceedings of the Third International Conference on Biomedical Image Registration*, WBIR'06, pages 230–237, Berlin, Heidelberg, 2006. Springer-Verlag.