Machine Learning Methods in Computed Tomography Image Analysis

Verfahren des maschinellen Lernens zur Analyse computertomographischer Bilder

Der Technischen Fakultät der
Universität Erlangen–Nürnberg

zur Erlangung des Grades

DOKTOR–INGENIEUR

vorgelegt von

Johannes Feulner

Erlangen — 2012

Als Dissertation genehmigt von der
Technischen Fakultät der
Universität Erlangen-Nürnberg

Tag der Einreichung:	6.10.2011
Tag der Promotion:	5.3.2012
Dekan:	Prof. Dr.-Ing. Marion Merklein
Berichterstatter:	Prof. Dr.-Ing. Joachim Hornegger
	Prof. Adrian Barbu

Abstract

Lymph nodes have high clinical relevance because they are often affected by cancer, and also play an important role in all kinds of infections and inflammations in general. Lymph nodes are commonly examined using computed tomography (CT).

Manually counting and measuring lymph nodes in CT volumes images is not only cumbersome but also introduces the problem of inter-observer variability and even intra-observer variability. Automatic detection is however challenging as lymph nodes are hard to see due to low contrast, irregular shape, and clutter. In this work, a top-down approach for lymph node detection in 3-D CT volume images is proposed. The focus is put on lymph nodes that lie in the region of the mediastinum.

CT volumes that show the mediastinum are typically scans of the thorax or even the whole thoracic and abdominal region. Therefore, the first step of the method proposed in this work is to determine the visible portion of the body from a CT volume. This allows pruning the search space for mediastinal lymph nodes and also other structures of interest. Furthermore, it can tell whether the mediastinum is actually visible. The visible body region of an unknown test volume is determined by 1-D registration along the longitudinal axis with a number of reference volumes whose body regions are known. A similarity measure for axial CT slices is proposed that has its origin in scene classification. An axial slice is described by a spatial pyramid of histograms of visual words, which are code words of a quantized feature space. The similarity of two slices is measured by comparing their histograms. As features, descriptors of the Speeded Up Robust Features are used. This work proposes an extension of the SURF descriptors to an arbitrary number of dimensions (N-SURF). Here, we make use of 2-SURF and 3-SURF descriptors.

The mediastinal body region contains a number of structures that can be confused with lymph nodes. One of them is the esophagus. Its attenuation coefficient is usually similar, and at the same time it is often surrounded by lymph nodes. Therefore, knowing the outline of the esophagus both helps to reduce false alarms in lymph node detection, and to put focus on the neighborhood. In the second part of this work, a fully automatic method for segmenting the esophagus in 3-D CT images is proposed. Esophagus segmentation is a challenging problem due to limited contrast to surrounding structures and a versatile shape and appearance. Here, a multi step method is proposed: First, a detector that is trained to learn a discriminative model of the appearance is combined with an explicit model of the distribution of respiratory and esophageal air. In the next step, prior shape knowledge is incorporated using a Markov chain model and used to approximate the esophagus shape. Finally, the surface of this approximation is non-rigidly deformed to better fit the boundary of the organ.

The third part of this work is a method for automatic detection and segmentation of mediastinal lymph nodes. Having low contrast to neighboring structures, it is vital to incorporate as much anatomical knowledge as possible to achieve good detection rates. Here, a prior of the spatial distribution is proposed to model this knowledge. Different variants of this prior are compared to each other. This is combined with a discriminative model that detects lymph nodes from their appearance. It first generates a set of possible lymph node center positions. Two model variants are compared. Given a detected center point, either the bounding box of the lymph node is directly detected, or a lymph node is segmented. A feature set is introduced that is extracted from this segmentation, and a classifier is trained on this feature set and used to reject false detections.

Kurzfassung

Lymphknoten sind von hoher klinischer Relevanz. Zum einen sind sie häufig von Krebs betroffen, und zum anderen spielen sie eine wichtige Rolle bei Infektionen und Entzündungen. Zur Untersuchung von Lymphknoten ist die Computertomographie (CT) die bildgebende Modalität der Wahl.

Lymphknoten in CT Aufnahmen von Hand zu zählen und zu vermessen ist allerdings nicht nur mühsam, sondern die Ergebnisse variieren auch stark von Beobachter zu Beobachter, und sogar auch innerhalb eines Beobachters. Eine Automatische Detektion ist wegen leichter Verwechselbarkeit mit benachbarten Strukturen allerdings schwierig. In dieser Arbeit wird ein Top-Down Verfahren zur Automatischen Lymphknotendetektion vorgestellt, wobei der Fokus auf der Region des Mediastinums liegt.

CT Aufnahmen, die das Mediastinum zeigen, sind meist Aufnahmen der Brust oder des gesamten Rumpfes. Der erste Schritt des vorgestellten Verfahrens schätzt deshalb die sichtbare Körperregion eines CT Bildes. Dies erlaubt es, den Suchraum einzuschränken, und auch festzustellen, ob das Mediastinum im aktuellen Bild überhaupt sichtbar ist. Die Körperregion, die in einem unbekannten Volumenbild sichtbar ist, wird durch 1D Registrierung entlang der Längsachse mit Volumenbildern, deren Körperregionen bekannt sind, ermittelt. Ein Ähnlichkeitsmaß für axiale CT Schnittbilder wird vorgestellt, das ursprünglich aus dem Bereich der Szenenklassifikation kommt. Ein Schnittbild wird durch eine räumliche Pyramide von Histogrammen visueller Wörter beschrieben. Dies sind die Codewörter eines quantisierten Merkmalsraumes. Die Ähnlichkeit zweier Schnittbilder wird durch den Vergleich ihrer Histogramme gemessen. Als Merkmale werden die Deskriptoren der Speeded Up Robust Features (SURF) verwendet. In dieser Arbeit werden diese auf eine beliebige Anzahl von Dimensionen erweitert (N-SURF). Verwendet werden hier 2-SURF und 3-SURF Deskriptoren.

Einige Strukturen der Region des Mediastinums können leicht mit Lymphknoten verwechselt werden. Darunter fällt auch der Ösophagus (Speiseröhre). Sein Abschwächungskoeffizient ähnelt dem von Lymphknoten, und gleichzeitig ist er oft von Lymphknoten umgeben. Eine Segmentierung des Ösophagus hilft daher einerseits, falsch positive Lymphknotendetektionen zu vermeiden, und kann zum anderen verwendet werden, mehr Aufmerksamkeit auf die direkte Nachbarschaft zu lenken. Im zweiten Teil dieser Arbeit wird ein vollautomatisches Verfahren zur Ösophagussegmentierung in 3D CT Bildern vorgestellt. Ösophagussegmentierung ist wegen schlechten Kontrastes zu benachbarten Strukturen und unterschiedlichen Aussehens ein schwieriges Problem. Das hier vorgestellte Verfahren besteht aus mehreren Schritten: Zunächst wird ein Detektor trainiert, Ösophagussegmente mit Hilfe eines diskriminativen Modells anhand ihres Aussehens zu erkennen. Dies wird mit einem expliziten Modell von ösophagealer Luft und Atemluft kombiniert. Im nächsten Schritt wird durch eine Markovkette modelliertes a priori Formwissen hinzugenommen und verwendet, um auf die ungefähre Form zu schließen. Schließlich wird die approximierte Oberfläche nichtrigide deformiert und besser an die Organgrenzen angepasst.

Im dritten Teil dieser Arbeit wird ein Verfahren zur automatischen Detektion und Segmentierung mediastinaler Lymphknoten vorgestellt. Um gute Detektionsergebnisse zu erreichen, ist es wegen der leichten Verwechselbarkeit mit anderen Strukturen wichtig, so viel anatomisches Vorwissen wie möglich zu Hilfe zu nehmen. In dieser Arbeit wird dieses Vorwissen mittels einer räumlichen Aufenthaltswahrscheinlichkeit modelliert. Mehrere

Varianten dieser Aufenthaltswahrscheinlichkeit werden miteinander verglichen. Dies wird mit einem diskriminativen Modell, das Lymphknoten anhand ihres Aussehens erkennt, kombiniert. Es erzeugt zunächst eine Menge möglicher Lymphknotenmittelpunkte. Zwei Modellvarianten werden verglichen. Ausgehend vom detektierten Zentrum wird entweder direkt eine Bounding Box des Lymphknotens detektiert, oder der Lymphknoten wird segmentiert. Ein aus einer solchen Segmentierung extrahierbarer Merkmalssatz wird vorgestellt und verwendet, um einen Klassifikator zu trainieren und falsch positive Detektionen auszusortieren.

Acknowledgment

This work was created at the Pattern Recognition Lab (LME) of the University of Erlangen and at the CT T DE TC4 Siemens department in Erlangen in close collaboration with Siemens Corporate Research in Princeton, New Jersey. I am very grateful for having had the opportunity to work in such a fruitful environment and would like to thank the members of all three teams. In particular, I would like to thank the heads of the three departments, Prof. Dr.-Ing. Joachim Hornegger, Dorin Comaniciu, PhD, Dr. Martin Huber and Dr. Michael Sühling, for giving me the opportunity to work in these excellent teams.

Special thanks goes to my Siemens-internal advisor Kevin Zhou, PhD, for many discussions and invaluable comments and ideas about my work. I did benefit a lot from his experience and expertise in computer vision, pattern recognition and machine learning.

Many thanks to the members of the LME's segmentation group for good application talks, theory lectures and valuable brainstorming sessions.

Furthermore I would like to thank Prof. Adrian Barbu from the Florida State University, Tallahassee, for acting as the secondary reviewer of this thesis, and also for his prior work on Marginal Space Learning together with Yefeng Zheng, PhD, that I was able to reuse in this work.

I would also like to thank Andreas Fieselmann for sharing data with me, thus enabling a better comparison of our methods.

Thanks to Elli Angelopoulou, PhD, for proof-reading and improving some of my papers.

Thanks also to Dr. med. Matthias Hammon and Prof. Alexander Cavallaro for clinical advice, help with ground truth annotations, and image data, and to PD Dr. med. Michael Lell for acting as an additional member of my PhD committee.

I am also grateful for the financial support I received as a member of the Theseus Medico research project, led by Dr. Martin Huber and later Dr. Sascha Seifert, from both Siemens and the German Federal Ministry of Economics and Technology.

Johannes Feulner

Contents

Chapter 1

Introduction

1.1 The lymphatic system

The lymphatic system can be considered as a part of the circulatory system, with the other part being the cardiovascular system. The cardiovascular system contains blood that provides tissue with oxygen, nutrition and other vital supplies. Fluid and solved substances flow through capillary walls into the tissue. However, the capillaries cannot remove all fluid and waste products from the perfused tissue.

This is the task of the lymphatic system. Lymph vessels that pervade the tissue take up interstitial fluids, proteins and other substances, and finally carry it back into the cardiovascular system just before the right atrium. The fluid flowing inside the lymph vessels is called lymph. On its way back to the heart, it is several times filtered in lymph nodes. These are small, usually bean-shaped nodes with a diameter that is normally in the range of a few millimeters up to 2 cm [Warw 58]. In total, a human typically has more than 500 lymph nodes. Figure 1.1 illustrates both the lymphatic system and a lymph node. Lymph nodes are also a part of the immune system. They contain a variety of immune cells that detect and filter germs and other foreign particles and can initiate an immune response.

In case of an infection, lymph nodes can swell to a size of several centimeters due to the production of immune cells, and also because the inflow of immune cells from the blood exceeds the outflow. Lymph nodes can furthermore be enlarged due to cancer. This can be secondary cancer that is also called metastatic, meaning that the cancer developed elsewhere and has spread to the lymph nodes. Or it is lymphoma, a cancer of the lymphatic system itself.

Therefore, lymph nodes play an important role in diagnosis. They are routinely considered during all kinds of examinations, in particular those related to cancer. Lymph nodes are commonly examined using computed tomography (CT) scans.

1.2 German research project Theseus Medico

Theseus Medico[1] is a research project that was initiated by the German Federal Ministry of Economics and Technology in 2007. The aim is to develop a system for semantic search in medical image databases. A huge amount of image data is acquired in modern hospitals

[1] `http://www.theseus-programm.de/de/920.php` (retrieved on 25.06.2011)

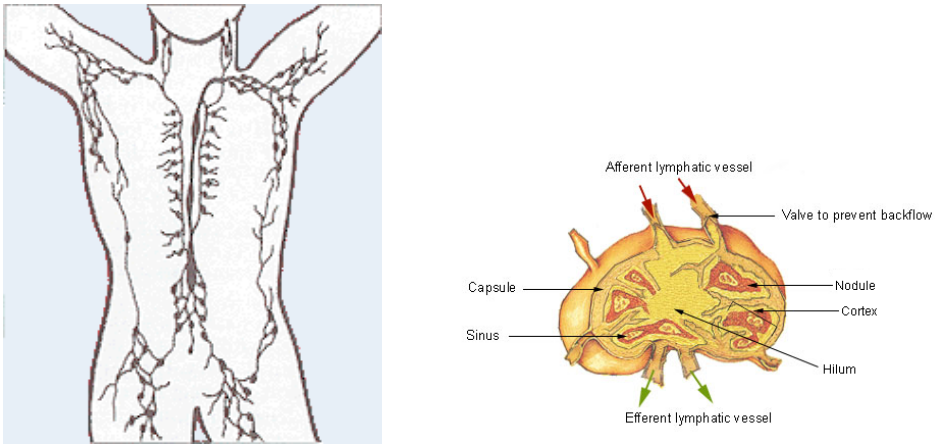


Figure 1.1: Left: Schematic illustration of the human lymphatic system. Right: Illustration of a lymph node. Sources: National Cancer Institute, SEER Training Modules[2], Module: „Lymph nodes“, U.S. National Institutes of Health, and VisualsOnline[3]. United States Government works, not copyright protected.

every day from a variety of modalities such as X-ray, CT, magnetic resonance (MR), and ultrasound. It becomes increasingly difficult for physicians to handle these amounts of data, and in particular to retrieve the information they are interested in from the databases. For instance, if a patient has a tumor in a certain region with a certain shape and appearance, a physician may be interested in patients that had similar lesions in order to find out which treatments were successful and which were not.

After an image is acquired, it is usually interpreted by the physician, and his findings are stored as text along with the image in the database. These texts can be searched by a user. Searching text is a standard problem, and efficient solutions based on index structures exist for decades. There are, however, limitations in this context:

- The way different physicians, or humans in general, describe the same image can vary a lot. While one description may be verbose, another one may be short. Or a physician only focuses on a particular aspect.
- The same concepts are often termed differently. This is obviously the case when findings are written in different languages. But even in the same language, there are often synonyms.
- Manual descriptions are in general not very detailed because physicians only have a limited amount of time to read an image. For instance, if there is a number of liver lesions visible in a CT image, the physician usually will not include statistics such as the volume, surface and precise location of the lesions because it simply would be too tedious.
- There is data that is difficult to describe with words, as for example shape and texture. Here, a numerical description is often more adequate.

[2] `http://training.seer.cancer.gov` (retrieved on 17.01.2012)

[3] `http://visualsonline.cancer.gov/details.cfm?imageid=3237` (retrieved on 17.01.2012)

These issues limit the use of the search tools for medical images that are available today. The strategy of the Theseus Medico project is to first develop a formal description language for medical images to solve the problems arising from synonyms and multiple languages, and that also includes numerical image features to allow visual search. Second, a *parsing system* for medical images is developed that understands what the images show, recognizes the visible objects and automatically generates formal descriptions.

Figure 1.2 gives an overview of the Theseus Medico system. A user can either pose a textual query or an image query. A text query is first semantically interpreted and converted into a formal description that is used for searching. If an image is used for a query, it is analyzed by an image parsing module. Depending on the type of query, it extracts information about what the image shows, such as the imaging modality, the body region, and statistics about the visible anatomic structures, or it extracts a set of features that describe the overall appearance of the image without a semantic interpretation. The database being searched contains medical images along with the corresponding findings from physicians. These are offline interpreted by the image parsing and text understanding modules that also analyze the search queries. The resulting formal descriptions of the images and the findings are also stored in the database and can be used for searching. For efficiency, the formal descriptions are stored in an index structure. Further details about the project and the system architecture can be found in [Zill 09] and [Mlle 07].

A problem is that the scope of the project is very wide. To be able to come up earlier with a working prototype, a small set of *use cases* has been defined, and one of them is lymphoma (see section 1.1).

This use case, and in particular the image parsing module, is the context of this work: Given an image, the aim is to automatically extract information about the visible lymph nodes, such as the number, the size, and the anatomical location.

1.3 Contributions

In this work, a top-down approach for lymph node detection and segmentation in CT volume images is proposed. The focus is put on lymph nodes that lie in the region of the mediastinum, which is the area between the lungs. It is of particular interest for radiologists during oncological examinations.

CT volumes that show the mediastinum are typically scans of the thorax or even the whole thoracic and abdominal region. Therefore, the first step of the method proposed in this work is to determine the visible portion of the body from a CT volume. This allows to prune the search space for mediastinal lymph nodes and also other structures of interest. Furthermore, it can tell whether the mediastinum is actually visible. The scientific contributions of this part of this work are listed below. The insights and results were also published in [Feul 09b, Feul 11c].

- A novel method was developed that estimates the visible body region from a CT volume image or a small number of axial image slices.

- A similarity measure for axial CT image slices is proposed that originates from scene classification.

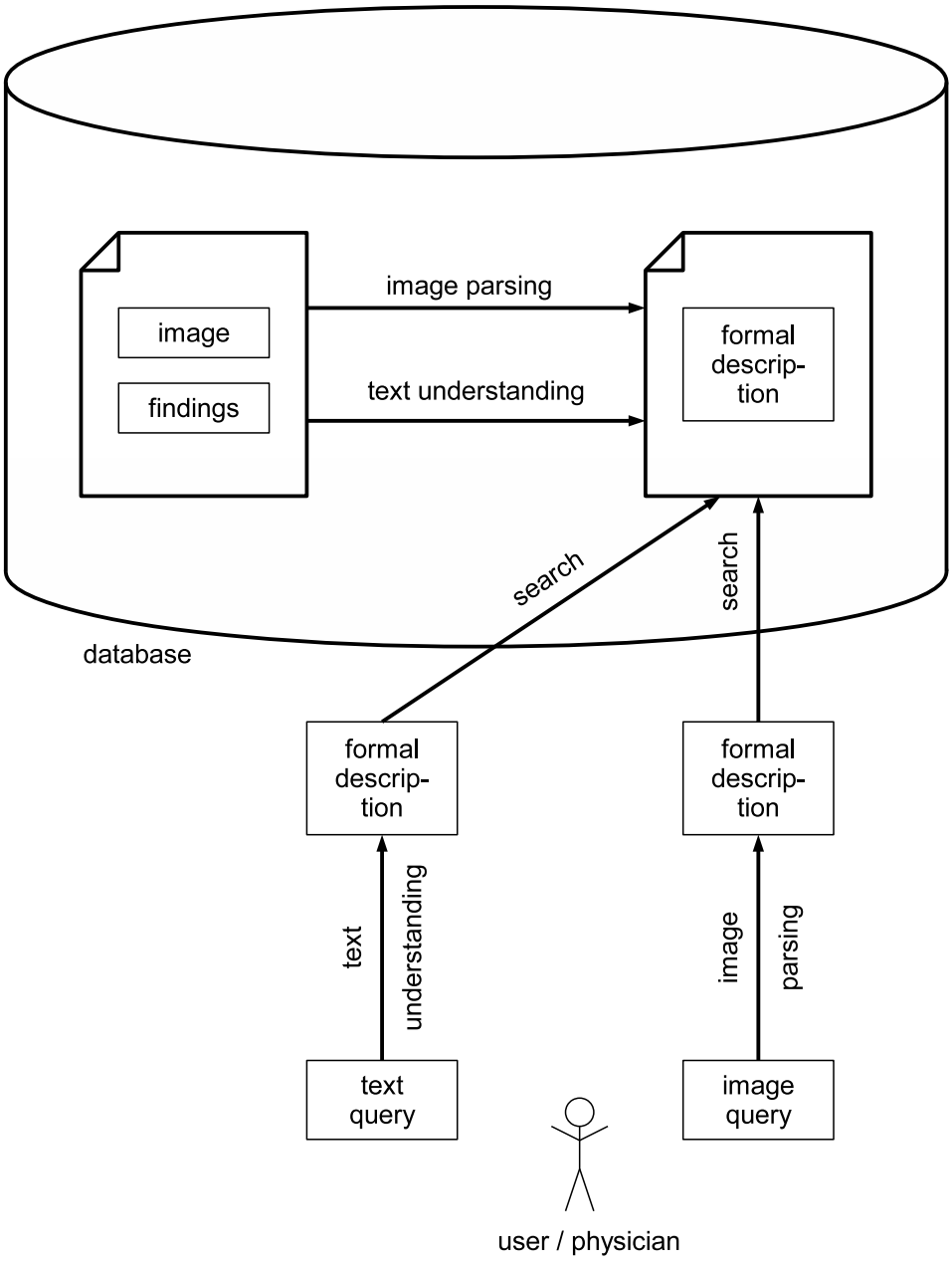


Figure 1.2: Overview of the system developed in the Theseus Medico project.

- This similarity measure makes use of a quantized high dimensional feature space. As features, the descriptors of the Speeded Up Robust Features (SURF) are used. These are high dimensional and very descriptive features that are popular in the computer vision community. However, these features were only described for 2-D images. In this work, N-SURF descriptors are proposed, which are an extension of SURF to an arbitrary number of dimensions. For efficient computation, SURF makes use of integral images, which are also extended to N dimensions.

The mediastinal body region contains a number of structures that can be confused with lymph nodes. One of them is the esophagus. Its attenuation coefficient is often similar to lymph nodes, and furthermore it is often surrounded by lymph nodes. Therefore, knowing the outline of the esophagus both helps to reduce false alarms in lymph node detection, and to put focus on the neighborhood. In the second part of this work,

- the first fully automatic method for segmenting the esophagus in 3-D CT images is proposed.
- Prior knowledge about the shape of the esophagus is modeled using a Markov chain framework. This is combined with a powerful detector based on discriminative learning.

This second part was previously published in [Feul 09a, Feul 10b, Feul 11a].

The third part of this work addresses the problem of detecting and segmenting mediastinal lymph nodes and contains the following contributions to research:

- A method is proposed that combines anatomical knowledge about where lymph can occur and are likely to occur with a detector that was trained to recognize lymph nodes from their appearance.
- The anatomical knowledge is modeled using a spatial prior of the lymph node density. Different variants of this prior are compared against each other.
- A semiautomatic segmentation method for lymph nodes is proposed that requires a single seed point as input and is a variant of the graph cuts method for image segmentation. A radial weighting factor of the graph is proposed to resolve the small cut problem and serve as a prior for blob-like shapes.
- A set of features in introduced that is extracted from segmented lymph nodes and used to classify them into true and false positives to further improve the detection performance.

This third part was published in [Feul 10a, Feul 11b].

1.4 Outline

The remainder of this thesis is structured as follows:

In chapter 2, state of the art classification techniques are explained that are used in later chapters. Understanding their principles helps to understand the rest of this work. The focus is clearly on boosting techniques. The most popular boosting method, AdaBoost, is

explained in detail. Next, the probabilistic boosting tree classifier is explained. This is a decision tree that uses AdaBoost classifiers at its inner nodes.

Chapter 3 explains standard features for image analysis that are used in later chapters. These are on the one hand two large pools of simple scalar features that are well suited to be used together with boosting techniques. The first pool contains Haar-like features that are computed from integrals over axis-aligned boxes. The second pool consists of steerable features. These are simple point features, evaluated on each point of a sampling pattern. On the other hand, the SURF descriptor is explained. This is a higher dimensional feature with a good descriptive power.

Chapter 4 presents the first major contribution of this work, the system for automatic estimation of the visible body region.

The second major contribution, the system for automatic segmentation of the esophagus in 3-D CT, is presented in chapter 5.

Chapter 6 contains the third major contribution of this work, the system for automatic lymph node detection and segmentation.

Chapter 7 gives an outlook, and chapter 8 summarizes this work.

Chapter 2

Boosting techniques for discriminative learning

2.1 Motivation

Discriminate learning techniques, and in particular boosting techniques, have become very popular for computer vision applications. This chapter gives an overview of classification techniques that are used in this thesis. Discriminative learning in general is explained and related to its counterpart generative learning. The *probably approximately correct* (PAC) property that boosting algorithms have by definition is explained together with AdaBoost, the most popular boosting algorithm. Finally, the probabilistic boosting tree classifier is described, which is a decision tree consisting of multiple AdaBoost classifiers.

2.2 Discriminative learning

Classification deals with the problem of inferring a discrete class variable $y \in Y = \{y_1 \dots y_K\}$ from a feature vector $\boldsymbol{x} \in X$. A classifier that maps a feature vector to a class variable can be seen as a partitioning of the feature space into disjoint decision regions

$$X = X_1 \cup X_2 \cup \ldots \cup X_K; \;\; X_i \cap X_j = \emptyset \; \forall i \neq j. \tag{2.1}$$

The region X_n covers all features that are assigned to class y_n.

We are now interested in the optimal decision regions X_i that minimize the expected cost C. This cost depends on a user-defined cost function $C(k, j)$ that maps a decision for a class y_j if the true class is y_k to a scalar cost. Note that k and j may also be the same, which corresponds to a correct classification. The expected cost of the decision regions is [Bish 07]

$$\mathbb{E}(C) = \sum_k \sum_j \int_{X_j} C(k, j) p(\boldsymbol{x}, y_k) d\boldsymbol{x}. \tag{2.2}$$

For a given feature vector $\boldsymbol{x}$, the class y_{opt} that minimizes the expected cost is then

$$y_{\text{opt}} = \underset{y_j}{\operatorname{argmin}} \sum_k C(k, j) p(\boldsymbol{x}, y_k), \tag{2.3}$$

which is equal to

$$y_{\text{opt}} = \underset{y_j}{\operatorname{argmin}} \sum_k C(k,j) p(y_k|\boldsymbol{x}) \tag{2.4}$$

because $p(\boldsymbol{x}, y) = p(y|\boldsymbol{x})p(\boldsymbol{x})$, and $p(\boldsymbol{x})$ does not depend on y_j.

Thus, we can make the optimal decision in terms of the cost function given the posterior probability $p(y|\boldsymbol{x})$. Estimating the posterior from training data directly is called discriminative learning.

Instead of directly modelling the posterior $p(y_k|\boldsymbol{x})$, it is also possible to express it as

$$p(y|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|y)p(y)}{p(\boldsymbol{x})} \tag{2.5}$$

and to model the likelihood $p(\boldsymbol{x}|y)$ and the priors $p(y)$ and $p(\boldsymbol{x})$. This is commonly referred to as generative learning [Bish 07]. For many problems, however, the likelihood $p(\boldsymbol{x}|y)$ and the prior $p(\boldsymbol{x})$ is not available or hard to compute. For instance, if we consider the problem of classifying image patches into the classes "face" and "no face", the likelihood functions for both classes will have a very complicated form due to the great diversity of image patches showing faces and something else than faces.

2.3 Boosting and probably approximately correct learning

Boosting is a technique for combining a number of binary classifiers, resulting in a classifier that is often much more powerful than any of the original ones. The combined classifier is commonly referred to as strong classifier, the original ones as weak classifiers.

Boosting can be viewed as a voting technique: Each of the weak classifiers votes for a class, and all votes are averaged to generate the final result.

By definition, all boosting algorithms are *probably approximately correct* (PAC) learning algorithms. PAC is a framework proposed in [Vali 84] for the mathematical analysis of machine learning. It basically means that given enough independent random training samples, with arbitrarily high probability, boosting will find an algorithm that has a generalization error that is arbitrarily low, given that the problem is *PAC learnable*. This in turn means that such an algorithm exists.

More formally, let us consider a binary classification problem with classes $Y = \{-1, 1\}$. Let further $c \in \mathcal{C}$ denote a *concept* that is to be learned. A concept is the subset of all features $\boldsymbol{x} \in X$ that belong to class 1. It is represented as a function $c : X \mapsto Y$. $\mathcal{C}$ is the set of all allowed concepts and is called a *concept class*. Let h denote a *hypothesis concept*, or classifier, that is the learned decision rule. Just like c, it is represented as a function $h : X \mapsto Y$. Let $\mathcal{D}$ denote a probability distribution over the feature space, and $p_{\mathcal{D}}(\boldsymbol{x})$ the prior probability of observing feature $\boldsymbol{x}$. Let further $\text{EX}(c\mathcal{D})$ denote a process that, every time when invoked, generates a labeled sample $(\boldsymbol{x}, y)$ that is randomly drawn from X according to $\mathcal{D}$.

We now define the *error* of the classifier h as

$$error(h) = \int_{\boldsymbol{x} \in X_{c \neq h}} p_{\mathcal{D}}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}, \tag{2.6}$$

where

$$X_{c \neq h} = \{\boldsymbol{x} \in X | c(\boldsymbol{x}) \neq h(\boldsymbol{x})\} \tag{2.7}$$

is the subset of X in which the classifier h is wrong. Note that $error(h)$ equals the probability that a randomly drawn sample is misclassified.

The PAC property of a learning algorithm can now be defined as follows:

Definition 1. Algorithm $\mathcal{L}$ is a strong PAC learning algorithm if for each concept $c \in \mathcal{C}$, for each probability distribution $\mathcal{D}$ over X, for each ε in the range $0 < \varepsilon < \frac{1}{2}$, for each δ in the range $0 < \delta < \frac{1}{2}$, with inputs ε and δ and access to EX($c\mathcal{D}$), $\mathcal{L}$ will find with a probability of at least $1 - \delta$ a hypothesis concept $h \in \mathcal{C}$ with $error(h) \leq \varepsilon$, given that an algorithm with this property exists for $\mathcal{C}$.

A more detailed introduction to the topic and an illustrative example can be found in [Kear 94]. We further define the weak PAC property of an algorithm as follows:

Definition 2. Algorithm $\mathcal{L}$ is a weak PAC learning algorithm if it has the same properties as a PAC learning algorithm but with the difference that ε is only in the range $\frac{1}{2} - \gamma \leq \varepsilon < \frac{1}{2}$, where $\gamma > 0$ is constant or decreases with $\frac{1}{p(s,n)}$. s in turn is the „size“ of the concept c in some encoding, n is a measure for the „size“ of a feature such as the dimension, and p is a polynomial in s and n.

In [Scha 90] it was shown that any weak PAC learning algorithm can be transformed into a strong PAC learning algorithm, and this transformation is called „boosting“. Algorithms that are similar to boosting but do not fulfill the PAC property are commonly called leveraging algorithms [Krau 04].

Note that many real world problems are not PAC learnable because in practice, different classes often overlap in feature space. If there are samples with different labels at the same position in the feature space, and the probability of observing such samples is above zero, then the classification error obviously cannot become arbitrarily small. An example is the recognition of single handwritten characters. There are, for instance, people that write the letters “u” and “n” in exactly the same way. Or classes overlap because of imperfect features that do not contain enough information. Consider, for instance, the problem of guessing the gender of a person given the size of the shoes and the weight. While there is certainly a correlation, there is also overlap that prevents the classification error from approaching zero, even if an arbitrary number of training examples is available.

The first boosting algorithms were proposed by Schapire and Freund in [Scha 90] and [Freu 90]. Later, they proposed an improved boosting algorithm called AdaBoost. This became one of the most popular boosting algorithms.

2.4 AdaBoost

AdaBoost means “adaptive boosting”. In contrast to previous earlier boosting algorithms, AdaBoost puts special focus on samples that are misclassified. It was proposed in [Freu 95]

1: **Input:** Labeled samples $(\boldsymbol{x}_n, y_n)$, $n = 1 \dots N$, $\boldsymbol{x}_n \in X$, $y_n \in Y = \{-1, 1\}$, number T of weak classifiers to train.
2: Initialize sample weights $w_i = \frac{1}{N}$
3: **for** $t = 1 \dots T$ **do**
4: Find the weak classifier $h_t : X \mapsto \{-1, 1\}$ in the pool $\mathcal{H}$ of weak classifiers that has the minimum weighted classification error ϵ_t:

$$h_t \leftarrow \underset{h \in \mathcal{H}}{\operatorname{argmin}}\, \epsilon, \text{ where } \epsilon = \sum_{n=1}^{N} w_n \left[y_n \neq h(\boldsymbol{x}_n) \right]. \tag{2.8}$$

5: Set weak classifier weight

$$\alpha_t \leftarrow \frac{1}{2} \ln \frac{1 - \epsilon_t}{\epsilon_t}. \tag{2.9}$$

6: Update sample weights

$$w_n \leftarrow \frac{w_n \exp\left(-\alpha_t y_n h_t(\boldsymbol{x}_n)\right)}{Z} \text{ where } Z \text{ is chosen such that } \sum_{n=1}^{N} w_n = 1 \tag{2.10}$$

7: **end for**
8: **Output:** The classifier

$$\operatorname{sign} H(\boldsymbol{x}), \text{ where } H(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \tag{2.11}$$

Figure 2.1: The AdaBoost algorithm [Freu 96]. The computational complexity is $O(N \cdot |\mathcal{H}| \cdot T)$.

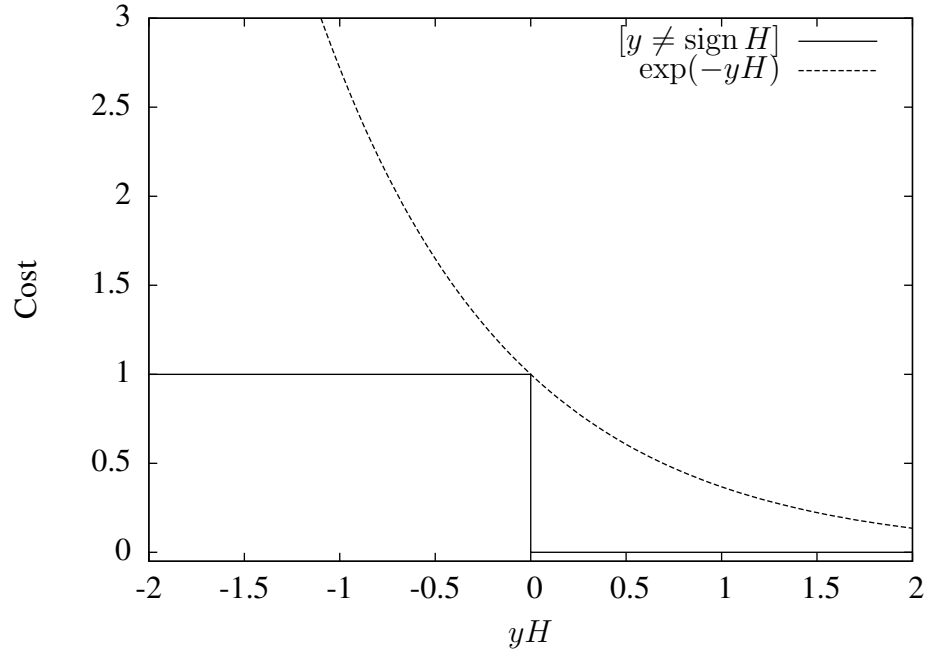


Figure 2.2: The misclassification and the exponential cost functions.

by Freund and Schapire. An introduction that is shorter than [Freu 95] can be found in [Freu 99].

The algorithm in shown in Figure 2.1. Its input is the number T of weak classifiers to train, and a set of labeled training samples

$$\{(\boldsymbol{x}_n, y_n) | n = 1 \dots N\}, \tag{2.12}$$

where $\boldsymbol{x}_n$ is an element of the feature space X, and $y_n \in Y = \{-1, 1\}$ is the class label of $\boldsymbol{x}_n$. Each sample $(\boldsymbol{x}_n, y_n)$ is associated with a weight w_n. In each iteration t, the weak classifier h_t with the lowest weighted classification error is selected from a pool $\mathcal{H}$ of weak classifiers. The classifier h_t is weighted with α_t that depends on its classification error (2.9). Then, the weights of the samples are multiplicatively updated: The weights of samples that are misclassified by h_t are increased, while the weights of the correctly classified samples are decreased. In further iterations, the algorithm will try harder to correctly classify the samples with a high weight. The final classifier is the majority vote of all weighted weak classifiers (2.11).

AdaBoost works surprisingly well [Freu 96], and it was also observed that it is not very prone to overfitting, meaning that for many problems, even for high values of T the performance on test data still does not decrease.

In [Frie 00], AdaBoost was examined from a statistical point of view. It turns out that AdaBoost has an exponential cost function, and that it optimizes the expected cost $J(H)$

$$J(H) = \mathbb{E}\left(\exp(-yH(\boldsymbol{x}))\right) \approx \frac{1}{N} \sum_{n=1}^{N} \exp(-y_n H(\boldsymbol{x}_n)) \tag{2.13}$$

using adaptive Newton updates. This can be proven by showing that Newton optimization of $J(H)$ results in update rules that are identical to (2.9) and (2.10), and it also results in the AdaBoost rule for selecting the next weak classifier:

In each iteration t of AdaBoost, the next weak classifier h_t and its weight α_t are

$$(h_t, \alpha_t) = \underset{\alpha, h \in \mathcal{H}}{\operatorname{argmin}} \sum_{n=1}^{N} \exp\left(-y_n H_t(\boldsymbol{x}_n)\right) \tag{2.14}$$

$$= \underset{\alpha, h \in \mathcal{H}}{\operatorname{argmin}} \sum_{n=1}^{N} \exp\left(-y_n (H_{t-1}(\boldsymbol{x}_n) + \alpha h(\boldsymbol{x}_n))\right) \tag{2.15}$$

$$= \underset{\alpha, h \in \mathcal{H}}{\operatorname{argmin}} \sum_{n=1}^{N} \exp\left(-y_n H_{t-1}(\boldsymbol{x}_n)\right) \exp\left(-y_n \alpha h(\boldsymbol{x}_n)\right), \tag{2.16}$$

given that AdaBoost optimizes the exponential cost function (2.13). In (2.16), the term right of the argmin operator can be viewed as a weighted sum over the samples, with a weight

$$w_n^{(t-1)} \propto \exp\left(-y_n H_{t-1}(\boldsymbol{x}_n)\right) \tag{2.17}$$

of sample n in iteration $t-1$. The weights of the samples are normalized and defined such that their sum equals one:

$$\sum_{n=1}^{N} w_n^{(t-1)} = 1. \tag{2.18}$$

Note that the definitions (2.17) and (2.18) correspond directly to the weight update rule (2.10) of AdaBoost, because

$$\exp\left(-y_n H_t(\boldsymbol{x}_n)\right) = \exp\left(-y_n (H_{t-1}(\boldsymbol{x}_n) + \alpha_t h_t(\boldsymbol{x}_n))\right) \tag{2.19}$$

$$= \exp\left(-y_n H_{t-1}(\boldsymbol{x}_n)\right) \exp\left(-y_n \alpha_t h_t(\boldsymbol{x}_n)\right) \tag{2.20}$$

$$\propto w_n^{(t-1)} \exp\left(-y_n \alpha_t h_t(\boldsymbol{x}_n)\right). \tag{2.21}$$

With (2.17), (2.16) becomes

$$(h_t, \alpha_t) = \underset{\alpha, h \in \mathcal{H}}{\operatorname{argmin}} \sum_{n=1}^{N} w_n^{(t-1)} \exp\left(-y_n \alpha h(\boldsymbol{x}_n)\right). \tag{2.22}$$

We now approximate the sum in (2.22) with its second order Taylor expansion around $h(\boldsymbol{x}) = 0$:

$$\sum_{n=1}^{N} w_n^{(t-1)} \exp\left(-y_n \alpha h(\boldsymbol{x}_n)\right) \approx \sum_{n=1}^{N} w_n^{(t-1)} \left[1 - y_n \alpha h(\boldsymbol{x}_n) + \frac{y_n^2 \alpha^2 h^2(\boldsymbol{x}_i)}{2}\right] \tag{2.23}$$

$$= \sum_{n=1}^{N} w_n^{(t-1)} \left[1 - y_n \alpha h(\boldsymbol{x}_n) + \frac{\alpha^2}{2}\right]. \tag{2.24}$$

From (2.23) to (2.24), we used that $y_n^2 = 1$ and $h^2(\boldsymbol{x}_i) = 1$ because both y_n and $h(\boldsymbol{x}_i)$ are in $\{-1, 1\}$.

We first find the optimal weak classifier h_t for a fixed (but arbitrary) positive value of α:

$$h_t = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \sum_{n=1}^{N} w_n^{(t-1)} \left[1 - y_n \alpha h(\boldsymbol{x}_n) + \frac{\alpha^2}{2}\right]. \tag{2.25}$$

With $\alpha > 0$, this equals

$$h_t = \underset{h \in \mathcal{H}}{\operatorname{argmax}} \sum_{n=1}^{N} w_n^{(t-1)} y_n h(\boldsymbol{x}_n). \tag{2.26}$$

Since there are only two possible values of both y_n and $h(\boldsymbol{x}_n)$, this can be written as

$$h_t = \underset{h \in \mathcal{H}}{\operatorname{argmax}} \sum_{n=1}^{N} w_n^{(t-1)} [y_n = h(\boldsymbol{x}_n)] - \sum_{n=1}^{N} w_n^{(t-1)} [y_n \neq h(\boldsymbol{x}_n)], \tag{2.27}$$

where

$$[y_n = h(\boldsymbol{x}_n)] = \begin{cases} 1 & \text{if } y_n = h(\boldsymbol{x}_n) \\ 0 & \text{otherwise,} \end{cases} \tag{2.28}$$

and $[y_n \neq h(\boldsymbol{x}_n)]$ is defined analogously. Equation (2.27) in turn can be expressed as

$$h_t = \underset{h \in \mathcal{H}}{\operatorname{argmax}} \left(1 - \sum_{n=1}^{N} w_n^{(t-1)} [y_n \neq h(\boldsymbol{x}_n)] \right) - \sum_{n=1}^{N} w_n^{(t-1)} [y_n \neq h(\boldsymbol{x}_n)] \tag{2.29}$$

by using (2.18), and further rewritten as

$$h_t = \underset{h \in \mathcal{H}}{\operatorname{argmax}}\, 1 - 2 \sum_{n=1}^{N} w_n^{(t-1)} [y_n \neq h(\boldsymbol{x}_n)] \tag{2.30}$$

$$= \underset{h \in \mathcal{H}}{\operatorname{argmin}} \sum_{n=1}^{N} w_n^{(t-1)} [y_n \neq h(\boldsymbol{x}_n)]. \tag{2.31}$$

Note that equation (2.31) equals the weak classifier selection rule (2.8) of the AdaBoost algorithm shown in 2.1.

Next, we optimize the classifier weight α for a given weak classifier h:

$$\alpha_t = \underset{\alpha}{\operatorname{argmin}} \sum_{n=1}^{N} w_n^{(t-1)} \exp\left(-y_n \alpha h(\boldsymbol{x}_n)\right). \tag{2.32}$$

The sum in (2.32) can be rewritten as

$$\begin{aligned} \sum_{n=1}^{N} w_n^{(t-1)} \exp\left(-y_n \alpha h(\boldsymbol{x}_n)\right) = &\sum_{n=1}^{N} w_n^{(t-1)} \exp(-\alpha) [y_n = h(\boldsymbol{x}_n)] \\ &+ \sum_{n=1}^{N} w_n^{(t-1)} \exp(\alpha) [y_n \neq h(\boldsymbol{x}_n)]. \end{aligned} \tag{2.33}$$

If we define the weighted misclassification error ϵ as

$$\epsilon = \sum_{n=1}^{N} w_n^{(t-1)} [y_n \neq h(\boldsymbol{x}_n)], \tag{2.34}$$

then (2.33) becomes

$$\sum_{n=1}^{N} w_n^{(t-1)} \exp\left(-y_n \alpha h(\boldsymbol{x}_n)\right) = e^{-\alpha}(1-\epsilon) + e^{\alpha}\epsilon, \tag{2.35}$$

where again (2.18) was used. Equation (2.35) can be minimized by setting the derivative with respect to α to zero

$$\frac{d}{d\alpha} e^{-\alpha}(1-\epsilon) + e^{\alpha}\epsilon = -e^{-\alpha}(1-\epsilon) + e^{\alpha}\epsilon \tag{2.36}$$

$$:= 0, \tag{2.37}$$

which leads to

$$\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon}{\epsilon}. \tag{2.38}$$

Note that (2.38) is precisely the AdaBoost update rule (2.9) for the weak classifier weight. This proves that AdaBoost optimizes an exponential cost function using Newton updates.

Since the exponential function is convex and the expectation operator $\mathbb{E}$ can be approximated by the average (2.13) and due to the fact that a sum of convex functions is again a convex function, $J(H)$ is convex. This means that Newton updates are a very efficient method for optimizing $J(H)$. The exponential cost function $\exp(-yH(\boldsymbol{x}))$ can also be viewed as a convex upper bound of the misclassification cost function $[y \neq \operatorname{sign} H(\boldsymbol{x})]$ (see Figure 2.2).

In [Frie 00], it was further shown that the result of $H(\boldsymbol{x})$ has a probabilistic interpretation. The expectation of $\exp(-yH(\boldsymbol{x}))$ for a given $\boldsymbol{x}$ can be written as

$$\mathbb{E}\left(e^{-yH(\boldsymbol{x})} \middle| \boldsymbol{x}\right) = p(y=1|\boldsymbol{x})e^{-H(\boldsymbol{x})} + p(y=-1|\boldsymbol{x})e^{H(\boldsymbol{x})}. \tag{2.39}$$

Because the expectation of the exponential cost with respect to H is convex, there is exactly one local minimum, which is also the global one. Therefore, the derivative of the expected cost with respect to H

$$\frac{\partial \mathbb{E}\left(e^{-yH(\boldsymbol{x})} \middle| \boldsymbol{x}\right)}{\partial H(\boldsymbol{x})} = -p(y=1|\boldsymbol{x})e^{-H(\boldsymbol{x})} + p(y=-1|\boldsymbol{x})e^{H(\boldsymbol{x})} = 0 \tag{2.40}$$

must be zero for a H that minimizes the cost. After solving for H, we get

$$H(\boldsymbol{x}) = \frac{1}{2} \log \frac{p(y=1|\boldsymbol{x})}{p(y=-1|\boldsymbol{x})}, \tag{2.41}$$

and solving for $p(y=1|\boldsymbol{x})$ yields

$$p(y=1|\boldsymbol{x}) = \frac{e^{H(\boldsymbol{x})}}{e^{-H(\boldsymbol{x})} + e^{H(\boldsymbol{x})}}. \tag{2.42}$$

This shows that AdaBoost can be used to directly estimate the posterior probability $p(y = 1|\boldsymbol{x})$.

2.5 Probabilistic boosting tree

AdaBoost combines a set of weak classifiers into a strong classifier. In turn, we can combine multiple strong classifiers into a new one that may be even more powerful.

Viola and Jones [Viol 01] proposed to use a cascade of AdaBoost classifiers for object detection. The detection threshold of all classifiers except the last one is selected such that the recall is close to one, at the cost of a high false positive rate like 50%. However, as both the detection rate and the false positive rate are multiplied at each level of the cascade, the false positive rate quickly approaches zero, while the detection rate remains close to one. The negative training examples at each level are selected from the false positives of the previous stage. The cascade also helps to speed up the detection. In object detection problems, there is typically a strong bias toward the negative class, and negative samples can already be rejected in early stages.

This idea was extended in [Tu 05]: There, the cascade is replaced by a binary decision tree that has an AdaBoost classifier at each node. If a sample is classified as negative at a node, it is not necessarily rejected but can also be passed down to a child that is specialized to detect positive samples that were previously classified as negatives. The classifier is called probabilistic boosting tree (PBT).

2.5.1 Training and testing

A PBT with two levels of strong classifiers is illustrated in Figure 2.3 (top). The inner nodes of the tree (circles) are AdaBoost classifiers. Each inner node stores its empirical class distribution. These class distributions are visualized as boxes. All leaf node are empirical class distributions as well.

The basic principle is that a sample $\boldsymbol{x}$ is classified by a strong classifier that was trained to learn $p(y|\boldsymbol{x})$. Then, the sample is passed down either to the right if $p(y|\boldsymbol{x})$ is close to one, to the left if $p(y|\boldsymbol{x})$ is small, or to both child nodes if $p(y|\boldsymbol{x})$ is close to $\frac{1}{2}$. If the sample is only passed down to one child, the result from the other child is set to its empiric class distribution. The final classification score is the average of the scores from the child nodes weighted by $p(y|\boldsymbol{x})$. It is claimed in [Tu 05] that this classification score approximates the posterior $p(y|\boldsymbol{x})$.

Figure 2.4 summarizes the training algorithm of the PBT. It starts by training an AdaBoost classifier on the set of labeled input samples $(\boldsymbol{x}_n, y_n)$, $n = 1 \ldots N$. This classifier then divides the set of samples into two overlapping subsets. Samples with a high classification score $p(y = 1|\boldsymbol{x})$ are only inserted into the right subset, samples with a low score are only inserted into the left set. If the score $p(y = 1|\boldsymbol{x})$ of a sample is close to $\frac{1}{2}$, it is inserted into both sets. The amount of overlap is controlled by the parameter Δp. For $\Delta p = 0$, there is no overlap between the sets. For each strong classifier, the empirical class distribution is estimated from its training samples. This procedure is repeated recursively for the two child nodes. If the maximum tree depth L is reached, no classifier is trained and the node only stores the class distribution of the samples. Figure 2.3 (bottom) illustrates how a set of training samples is split by the nodes and passed down to the children.

The testing algorithm is summarized in Figure 2.5. The classification score $F_{\mathcal{N}}(\boldsymbol{x})$ of a tree node $\mathcal{N}$ is determined by first computing the score $p(y|\boldsymbol{x})$ of the node's AdaBoost classifier. The final classification score is then set to the weighted average of the child

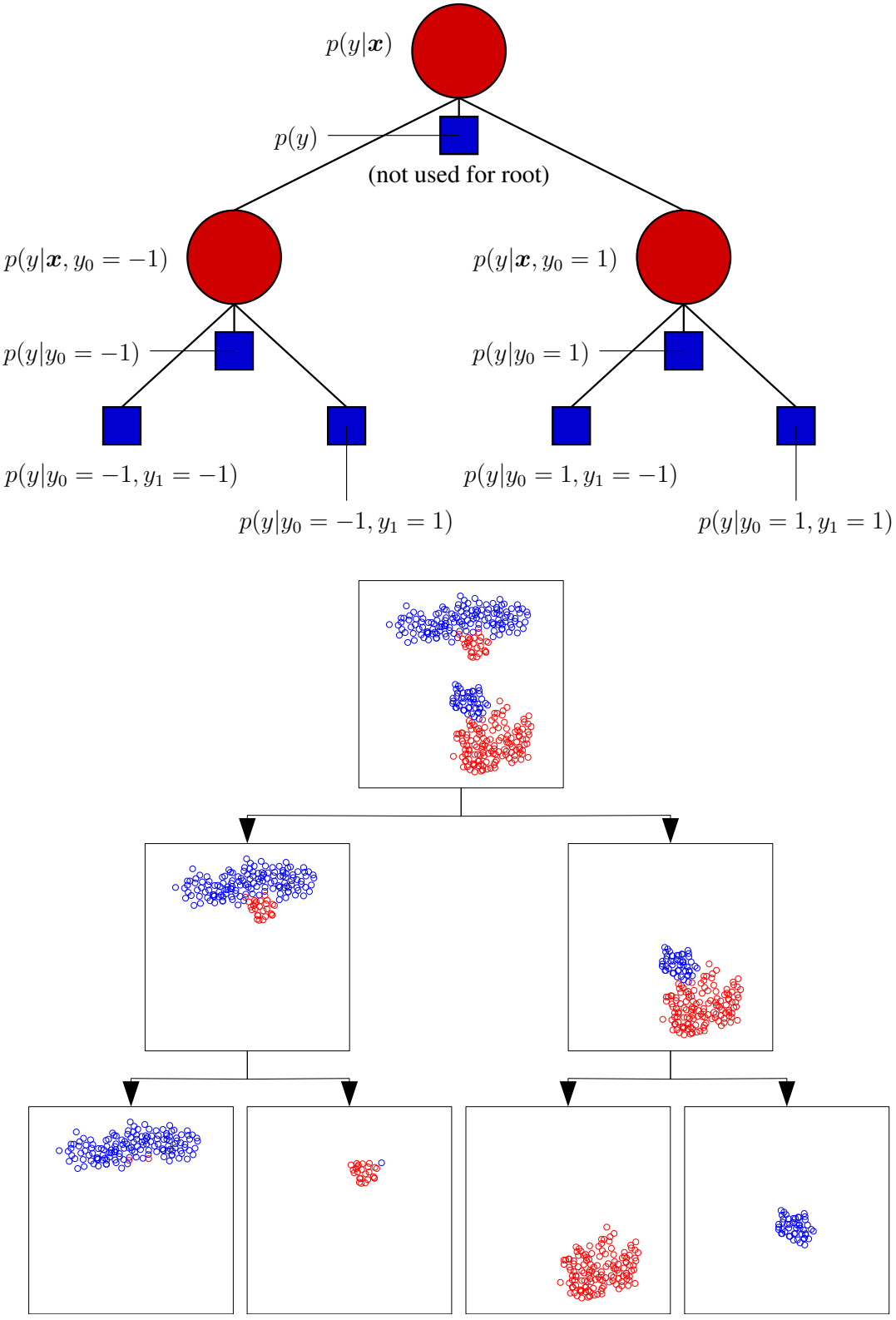


Figure 2.3: Top: Illustration of a probabilistic boosting tree with two levels. The circles are AdaBoost classifiers, the squares contain the empirical class distribution. Bottom: Illustration of how the sample set is separated by the nodes. The first node is trained on all samples. It splits the set into two subsets that may in general overlap, and the child nodes are trained on the subsets. Red and blue correspond to the two classes.

1: **Input:** Labeled samples $(\boldsymbol{x}_n, y_n)$, $n = 1 \dots N$, $\boldsymbol{x}_n \in X$, $y_n \in Y = \{-1, 1\}$, number of layers L, margin parameter Δp, maximum weak classifier error $\epsilon_{\max}$, maximum number of weak classifiers T.
2: Define set of uniformly weighted samples $S = \left\{(\boldsymbol{x}_n, y_n, w_n) \middle| \, n = 1 \dots N, \, w_n = \frac{1}{N}\right\}$
3: Create a node. Empirically estimate the distribution

$$p(y) \approx \sum_n w_n [y_i = y]. \tag{2.43}$$

4: Store it with the node.
5: **if** the current tree depth is L **then**
6: This is a leaf node. Exit.
7: **end if**
8: This is an inner node. Train an AdaBoost classifier on S to learn $p(y|\boldsymbol{x})$. Stop if $\epsilon_t > \epsilon_{\max}$ OR $t \geq T$. Store the strong classifier with the node.
9: Initialize two empty sets S_{left}, S_{right}.
10: **for** every weighted sample $(\boldsymbol{x}_n, y_n, w_n) \in S$ **do**
11: **if** $p(y = 1|\boldsymbol{x}_n) - \frac{1}{2} > \Delta p$ **then**
12:

$$S_{\text{right}} \leftarrow S_{\text{right}} \cup \{(\boldsymbol{x}_n, y_n, w_n)\} \tag{2.44}$$

13: **else if** $p(y = -1|\boldsymbol{x}_n) - \frac{1}{2} > \Delta p$ **then**
14:

$$S_{\text{left}} \leftarrow S_{\text{left}} \cup \{(\boldsymbol{x}_n, y_n, w_n)\} \tag{2.45}$$

15: **else**
16:

$$S_{\text{right}} \leftarrow S_{\text{right}} \cup \{(\boldsymbol{x}_n, y_n, p(y = 1|\boldsymbol{x}))\} \tag{2.46}$$

$$S_{\text{left}} \leftarrow S_{\text{left}} \cup \{(\boldsymbol{x}_n, y_n, p(y = -1|\boldsymbol{x}))\} \tag{2.47}$$

17: **end if**
18: **end for**
19: Normalize the weights in S_{left}
20: Recursively continue with S_{left} and step 5
21: Normalize the weights in S_{right}
22: Recursively continue with S_{right} and step 5
23: **Output:** The PBT classifier.

Figure 2.4: Training algorithm of the probabilistic boosting tree [Tu 05].

scores $f_{\text{left}}(\boldsymbol{x})$ and $f_{\text{right}}(\boldsymbol{x})$ (see equation (2.55)). Here, the score of the child $f_{\text{left}}(\boldsymbol{x})$ is either set to $F_{\text{left}}(\boldsymbol{x})$ and computed recursively if $p(y = 1|\boldsymbol{x}) < \frac{1}{2} + \Delta p$, or else the left subtree is pruned and $f_{\text{left}}(\boldsymbol{x})$ is set to the empirical class distribution at the left child. $f_{\text{right}}(\boldsymbol{x})$ is computed likewise. In this work, the parameter Δp is always set to 0.1.

According to [Tu 05], the PBT classifier can also separate classed with complicated and interwoven distributions. However, it is also more prone to overfitting compared to a single AdaBoost classifier, and the probabilistic interpretation of the detection score is only an heuristic. For instance, the final detection score is bounded by the empirical distributions in the outer leaf nodes. If $p(y = 1) = 80\%$ in the right outer leaf node, the final classifier score will never be above that value, even if the AdaBoost classifiers are very certain about a particular sample.

It is worth mentioning that, despite of its name, the probabilistic boosting tree does work with any sort of strong classification algorithm that outputs a probability estimate. For instance, the AdaBoost classifiers could be exchanged with support vector machines.

2.5.2 Time complexity

To the best of our knowledge, the computational complexity of the PBT training and testing algorithm has not been analyzed so far. We will start with analyzing the computational requirements of the PBT training algorithm shown in Figure 2.4 with respect to the number of layers L in the tree, the number of training samples N and the sample set growth factor a from layer l to layer $l + 1$. a depends on the margin parameter Δp and the posterior distribution $p(y|\boldsymbol{x})$ learned by the AdaBoost classifier. It encodes how many samples are inserted in both children. For $\Delta p = 0$, no samples are inserted twice and thus $a = 1$. For $\Delta p = \frac{1}{2}$, all samples are inserted into both children, and the number of samples doubles at each layer. Thus, $a = 2$ in this case.

In layer l, there are 2^l nodes that have to be trained. The expected number of training samples $N(l)$ for a certain node in layer l is

$$N(l) = \frac{Na^l}{2^l}. \tag{2.56}$$

The time complexity for training a single AdaBoost classifier is (see Figure 2.1)

$$O(N \cdot |\mathcal{H}| \cdot T). \tag{2.57}$$

Thus, the total time complexity of the PBT training algorithm is in

$$O\left(\sum_{l=0}^{L} 2^l \frac{Na^l}{2^l} |\mathcal{H}|T\right) = O\left(N|\mathcal{H}|T \sum_{l=0}^{L} a^l\right). \tag{2.58}$$

The exponential sum that appears in (2.58) is bounded by

$$\int_0^{L+1} a^{l-1} dl < \sum_{l=0}^{L} a^l < \int_0^{L+1} a^l dl \tag{2.59}$$

because the mid term is the upper sum of the left integral and at the same time the lower sum of right integral for $a > 1$, as illustrated in Figure 2.6. By computing the integrals, we

Function $F_{\mathcal{N}} : \boldsymbol{x} \mapsto [0, 1]$ that computes the classification score at tree node $\mathcal{N}$:
Input: Sample $\boldsymbol{x}$
if $\mathcal{N}$ is a leaf node **then**
    Use empirical distribution at this node as classification score

$$F_{\mathcal{N}}(\boldsymbol{x}) \leftarrow p(y = 1) \tag{2.48}$$

    Exit.
end if
Estimate $p(y|\boldsymbol{x})$ using the AdaBoost classifier of the current tree node $\mathcal{N}$
Compute $f_{\text{left}}(\boldsymbol{x})$ and $f_{\text{right}}(\boldsymbol{x})$ as follows:
if $p(y = 1|\boldsymbol{x}) - \frac{1}{2} > \Delta p$ **then**
    Recursively descend into right child node:

$$f_{\text{left}}(\boldsymbol{x}) \leftarrow p_{\text{left}}(y = 1) \tag{2.49}$$
$$f_{\text{right}}(\boldsymbol{x}) \leftarrow F_{\text{right}}(\boldsymbol{x}) \tag{2.50}$$

else if $p(y = -1|\boldsymbol{x}) - \frac{1}{2} > \Delta p$ **then**
    Recursively descend into left child node:

$$f_{\text{left}}(\boldsymbol{x}) \leftarrow F_{\text{left}}(\boldsymbol{x}) \tag{2.51}$$
$$f_{\text{right}}(\boldsymbol{x}) \leftarrow p_{\text{right}}(y = 1) \tag{2.52}$$

else
    Recursively descend into both child nodes:

$$f_{\text{left}}(\boldsymbol{x}) \leftarrow F_{\text{left}}(\boldsymbol{x}) \tag{2.53}$$
$$f_{\text{right}}(\boldsymbol{x}) \leftarrow F_{\text{right}}(\boldsymbol{x}) \tag{2.54}$$

end if
Set classification score to weighted average

$$F_{\mathcal{N}}(\boldsymbol{x}) \leftarrow p(y = 1|\boldsymbol{x}) f_{\text{right}}(\boldsymbol{x}) + p(y = -1|\boldsymbol{x}) f_{\text{left}}(\boldsymbol{x}) \tag{2.55}$$

Output: Classification score $F_{\mathcal{N}}(\boldsymbol{x})$

Figure 2.5: Testing algorithm of the probabilistic boosting tree [Tu 05].

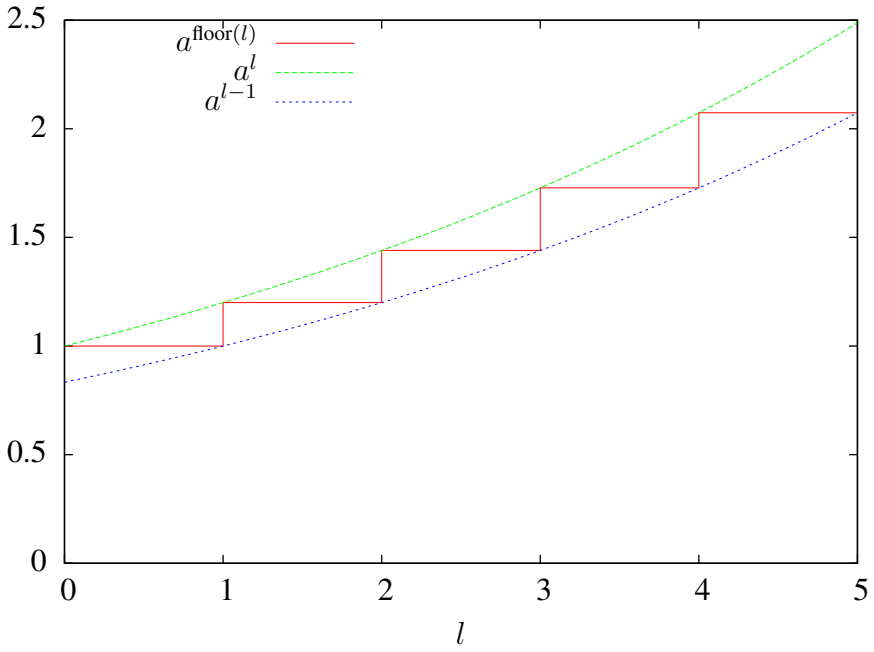


Figure 2.6: Illustration of (2.59) for $a = 1.2$.

see that they are both in the same complexity class

$$O\left(\int_0^{L+1} a^{l-1} dl\right) = O\left(\int_0^{L+1} a^l dl\right) = O(a^L). \tag{2.60}$$

Therefore, the complexity class of the exponential sum is

$$O\left(\sum_{l=0}^{L} a^l\right) = O(a^L) \tag{2.61}$$

and the complexity of the PBT training algorithm is

$$O\left(N|\mathcal{H}|Ta^L\right). \tag{2.62}$$

To analyze the time complexity of the PBT testing algorithm, we again make use of the sample growth factor a. The complexity class of the AdaBoost testing algorithm is in $O(T)$. A sample is examined T times by weak classifiers at the root level of the tree, Ta^2 times at the second level, Ta^3 times at the third level, and so on. Thus, the complexity class of the total PBT testing algorithm is in

$$O\left(\sum_{l=0}^{L} Ta^l\right) = \begin{cases} O\left(Ta^L\right) & \text{for } a > 1 \\ O\left(TL\right) & \text{for } a = 1 \end{cases} \tag{2.63}$$

where again (2.59) and (2.60) was used to resolve the exponential sum.

2.6 Conclusion

This chapter showed that the AdaBoost algorithm minimizes the expected exponential cost $\mathbb{E}\left(\exp(-yH(\boldsymbol{x}))\right)$, which is a convex optimization problem and can therefore be solved very efficiently. It was also shown that the score $H(\boldsymbol{x})$ of the trained classifier has a probabilistic interpretation because it can be directly converted into an estimate of the posterior distribution $p(y = 1|\boldsymbol{x})$. These two facts are reasons for the popularity of AdaBoost.

The probabilistic boosting tree classifier that originated from the idea of using boosted classifiers in a cascade was explained, and its training and testing time complexity was analyzed.

The performance of a classifier is however bounded by the quality of the features. The following chapter discusses state of the art features for image analysis.

Chapter 3

Features for 2-D and 3-D image analysis

3.1 Motivation

This chapter presents state of the art features for both 2-D and 3-D image analysis problems. The features explained here are used in later chapters and therefore worth a closer look.

3.2 Haar-like features

Haar-like features are weighted sums of simple box filters. Each box filter simply computes the sum of the image values inside the box. Although being simple, they are powerful because they can be computed very efficiently with the help of an integral image.

Haar-like features were introduced in [Viol 01] and there very successfully used for face detection. Figure 3.1 illustrates four different types of 2-D Haar features. Each feature is a scalar. The sum of the image intensities inside the dark boxes is subtracted from the sum of the image intensities inside the bright boxes. If a feature has an unequal number of black and white boxes, the boxes are weighted such that the sum of the weights of the white boxes equals the sum of the weights of the dark boxes. The features are computed inside a sliding window. The bounds of the window are illustrated by the four boxes. Different features can be generated by translating the boxes inside the sliding window, by scaling the boxes, or by changing the number and relative position of the boxes. Thus, easily a feature pool containing thousands of features can be generated.

3.2.1 Integral images

If it is implemented the straightforward way, computing the sum of pixels inside a box has a time complexity of $O(\text{number of pixels in the box})$. When the sums over a lot of boxes have to be computed for the same image, the computation time can be greatly reduced by using integral images that are also known as as summed-area tables in the computer graphics community [Crow 84]. The value of the pixel (p_1, p_2) of the integral image II of an image I is defined to be the sum

$$II(p_1, p_2) = \begin{cases} \sum_{i_1=1}^{p_1} \sum_{i_2=1}^{p_2} I(i_1, i_2) & \text{if } p_1 > 0 \wedge p_2 > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

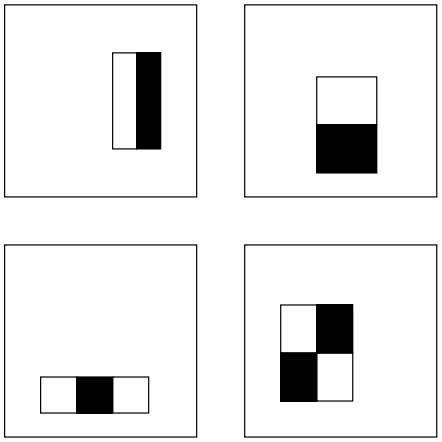


Figure 3.1: Four examples of 2-D Haar-like features as used in [Viol 01]. Dark boxes have a negative and bright boxes a positive weight.

of pixels in I inside the region that is bounded by the origin and (p_1, p_2). Computing the integral image has a complexity of $O(\text{number of pixels in the image})$. Once it is computed, the integrated intensity of a box bounded by the origin and a point in the image can be computed by simply looking up the value in the integral image. The integral over an arbitrary axis-aligned rectangle can be computed using two subtractions and one addition operation from four values of the integral image. This is illustrated in Figure 3.2. Thus, the time complexity of computing the integral over a box is constant (in $O(1)$) and does not depend on the size of the box.

3.2.2 Extension to 3-D

The concept of Haar features and integral images can be extended to three dimensions as described in [Tu 06]. Once the integral image has been computed, the integral over an axis aligned cuboid can be computed from eight values of the integral image. The 3-D Haar features proposed in [Tu 06] are simply rotated versions of the 2-D Haar features as used in [Viol 01] along with a new feature that consists of two nested boxes. These features are shown in Figure 3.3.

3.3 Steerable features

Haar-like features can be computed very efficiently and if the feature pool is large enough, they can also separate objects with a more complex appearance. However, a severe limitation of Haar-like features is that features that are rotated by an angle other than 90° cannot be efficiently extracted. That means, it is not possible to train a classifier using Haar-like features on a set of training images with a normalized orientation, and detect rotated objects in a test image using rotated features. On 2-D images, a possible workaround would be to simply generate rotated versions of the test image and generate an integral image for each angle. On 3-D images, this is however computationally intractable because 3-D rotations have three degrees of freedom in contrast to 2-D rotations that only have a single degree of freedom, and because 3-D images are typically much larger in size than 2-D images.

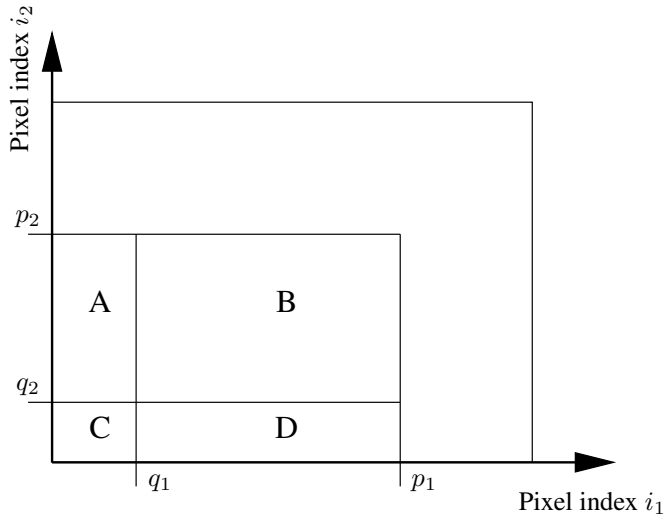


Figure 3.2: The pixel (q_1, q_2) of the integral image II contains the sum of the image pixels within the region C, $II(q_1, p_2) = A + C$, $II(p_1, q_2) = C + D$, and $II(p_1, p_2) = A + B + C + D$. Thus, the sum $B = II(p_1, p_2) - II(p_1, q_2) - II(q_1, p_2) + II(q_1, q_2)$ over an arbitrary rectangular axis-aligned region can be computed by accessing the integral image four times.

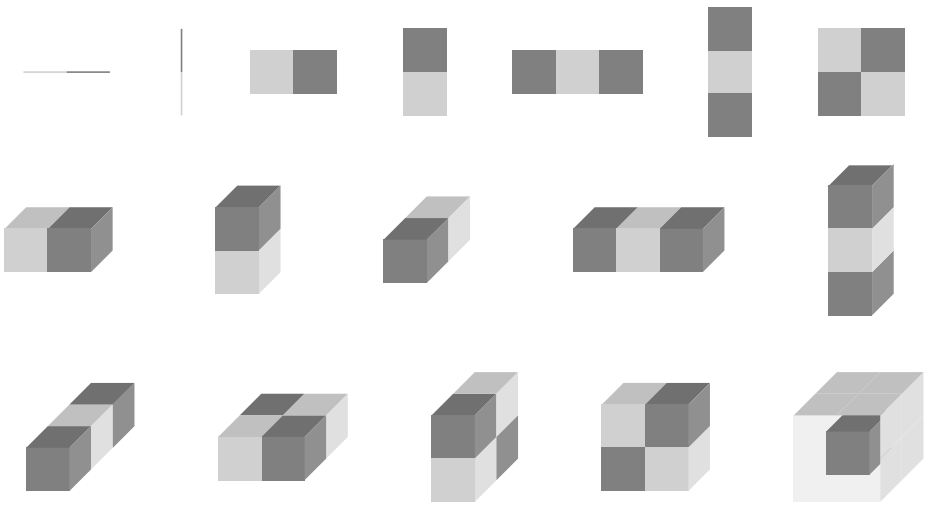


Figure 3.3: Illustration of 3-D Haar-like features as used in [Tu 06]. Dark boxes have a negative and bright boxes a positive weight.

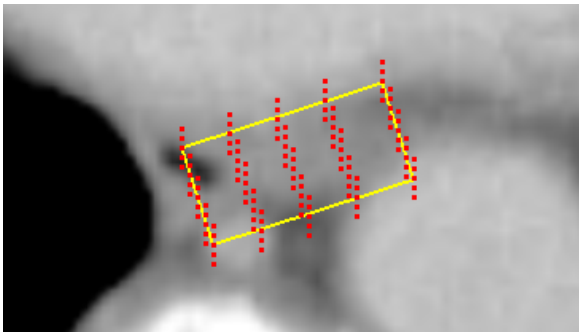


Figure 3.4: Example of a regular sampling pattern for the extraction of steerable features.

So-called "steerable features" that can be efficiently rotated and also scaled have been proposed in [Zhen 07]. These are a set of simple point features that are extracted on a sampling pattern. For each point (x, y, z) of the pattern, the image intensity $I(x, y, z)$, the partial derivatives $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$, $\frac{\partial I}{\partial z}$ in all directions, the gradient magnitude $\|\nabla I\|_2$, and nonlinear variations including I^2, I^3, $\log I$, $\sqrt{\|\nabla I\|_2}$, $\|\nabla I\|_2^2$, $\|\nabla I\|_2^3$, $\log \|\nabla I\|_2$ are used. In total, 24 scalar features are extracted for each point.

The sampling pattern cannot only be translated but also scaled and rotated. This makes steerable features particularly useful for finding an object that has an unknown orientation and scale.

A regular sampling pattern is a suitable choice for many applications and has been widely used in the literature [Iona 10, Vita 09, Wels 09, Seif 09], but it is also possible to use non-regular patterns that are specific to the problem. For instance, [Zhen 07] also proposed a sampling pattern that has the shape of the outline of the object being sought in order to get a high response when the sampling pattern matches the object.

Figure 3.4 illustrates how a regular sampling pattern is deformed to cover the object of interest, which is a section of the esophagus in this case.

3.4 Speeded-up robust features (SURF)

The "Speeded Up Robust Features" (SURF) [Bay 06, Bay 08] are popular in the computer vision community because they have good discriminative power, are robust and can be made invariant to rotation. They are similar to the older SIFT features [Lowe 99] which have been successfully used for various applications, for instance, scene classification tasks [Laze 06, Fei 05, Lowe 99], but can be computed faster.

To compute a standard SURF descriptor at a certain location $\boldsymbol{p} = (p_1, p_2)$ in a 2-D image $I(\boldsymbol{p})$, a regular sampling pattern of size 20×20 is placed on the image so that the center of the pattern is located at $\boldsymbol{p}$. For each point $\boldsymbol{r}$ of the pattern, the gradient $\nabla I(\boldsymbol{r})$ of the image is approximated with the responses (c_1, c_2) of two Haar filters, which are weighted with a 2-D Gaussian centered at $\boldsymbol{p}$. Both the Haar filters and the sampling pattern are shown in Figure 3.5. The sample spacing of the pattern is $1s$, and the size of the Haar filters is $2s$, where s is the scale of the descriptor. The response of a Haar filter is efficiently

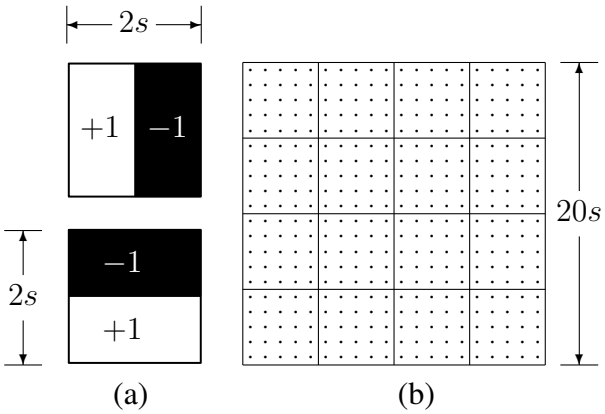


Figure 3.5: (a) The Haar filters for the 2-D case. (b) Sampling pattern for 2-SURF-4 (2-D with four bins b per dimension). The scale s of the descriptor equals the sample spacing and is half the size of the Haar filter.

computed with an integral image as described in section 3.2. The sampling pattern has 16 bins, each one containing 5×5 sample points. For each bin, a feature vector $\boldsymbol{v}$

$$\boldsymbol{v} = \left(\sum_{i=1}^{25} c_1^{(i)}, \sum_{i=1}^{25} c_2^{(i)}, \sum_{i=1}^{25} |c_1^{(i)}|, \sum_{i=1}^{25} |c_2^{(i)}| \right) \tag{3.2}$$

containing the summed and the summed absolute filter responses of the 25 samples of the bin is computed. The summation index i is the number of a sample inside a bin of the pattern. The feature vectors of all 16 bins are concatenated into a 64 dimensional descriptor.

3.5 Computational complexity of SURF

As the image gradient approximations can be computed in constant time with the help of an integral image, the time needed to compute a SURF descriptor linearly grows with the total number of sample points in the pattern. Note that is does not depend on the scale s of the descriptor.

3.6 Conclusion

This chapter presented three different feature sets that are popular for image analysis problems. While SURF have a very good descriptive power, Haar-like and steerable features have the advantage that single (scalar) components can be computed independently from each other. This makes them particularly suitable for boosting methods, where few features are selected from a large pool.

Chapter 4

Estimating the visible body region of 3-D CT scans

This work presented in this chapter was first published in [Feul 09b] and later in [Feul 11c] in an extended version.

4.1 Motivation

This chapter addresses the problem of determining which portion of the body is shown by a stack of axial CT image slices. For example, given a small stack of slices containing the heart region, one may want to automatically determine where in the human body it belongs.

Such a technique can be used in various applications such as attaching text labels to images of a database. A user may then search the database for volumes showing the heart. The DICOM protocol already specifies a flag "Body part examined", but this is imprecise as it only distinguishes 25 body parts. Moreover, the flag can often be wrong as reported by Gueld et al [Guel 02]. Or alternatively, our method may be used to reduce traffic load on medical image databases. Often physicians are only interested in a small portion of a large volume stored in the database. If it is known which parts of the body the large image shows, the image slices of interest showing e. g. the heart can be approximately determined and transferred to the user. Another possible application that is especially important for this work is the pruning of the search space of subsequent image analysis algorithms, like organ detectors.

The problem of estimating the body portion of a volume image is closely related to inter-subject image registration as it can be solved by registering the volume to an anatomical atlas. This is typically solved in two ways: By detection of anatomical landmarks in the volume image, or by intensity based non-rigid image registration. Landmark based registration may also be used as an initialization for non-rigid registration. However, a set of landmarks is required that covers all regions of the body and can be robustly detected. Intensity based registration tends to be slow, and because it is prone to getting stuck in local optima, it requires a good initialization. In many cases one is only interested in registration along the longitudinal (z) axis and a complete 3-D registration is not necessary.

Dicken et al. [Dick 08] proposed a method for recognition of body parts covered by CT volumes. An axial slice is described by a Hounsfield histogram with bins adapted to

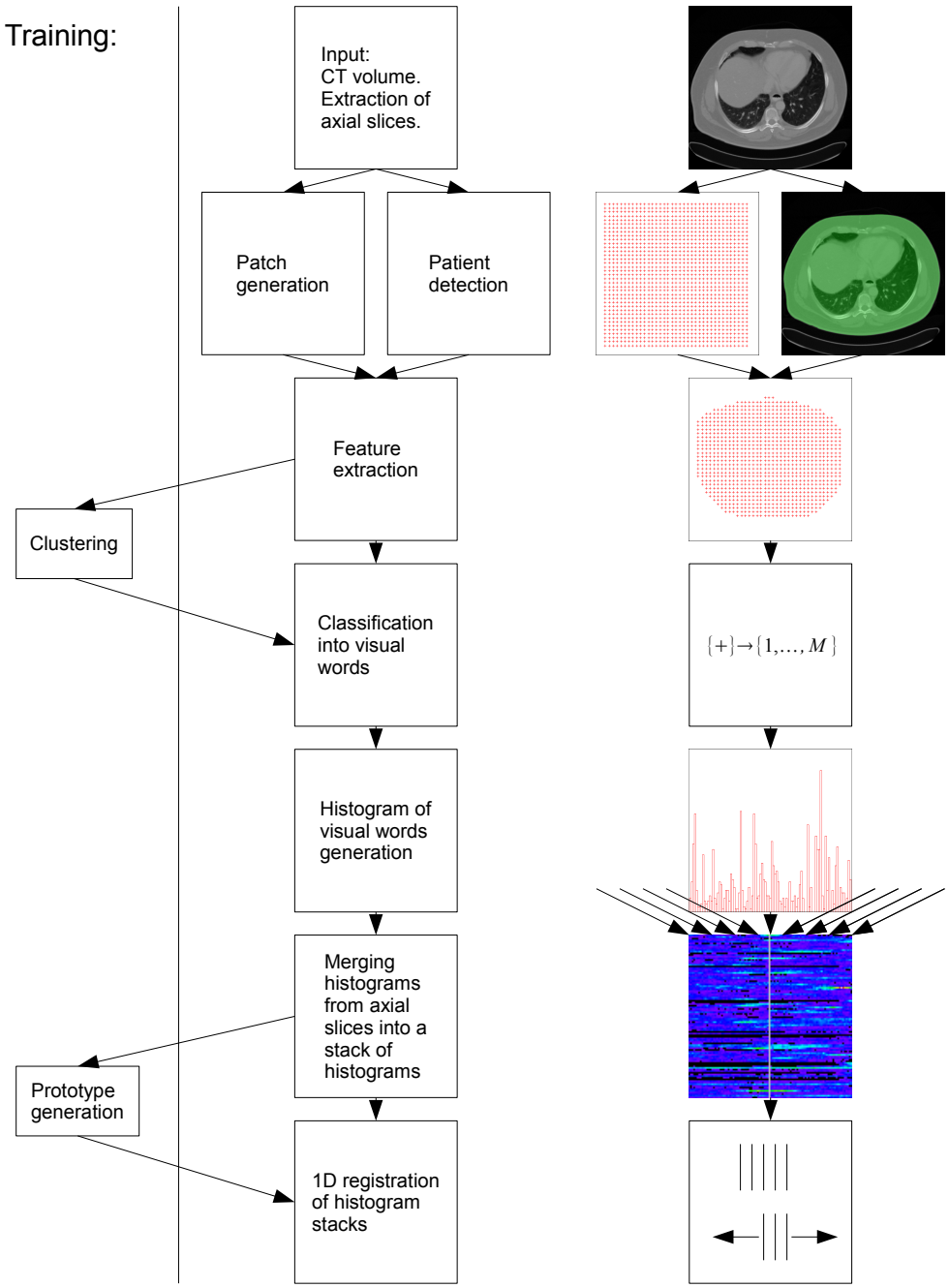


Figure 4.1: The proposed system for body portion estimation. The axial slices of a CT volume are first processed separately. sample positions are generated on a regular grid. For each sample position inside the patient, a SURF descriptor is computed from the local neighborhood called "patch". The descriptors are classified into visual words and accumulated in a histogram. The stack of histograms from the axial slices is registered with prototype histogram stacks to find the body portion.

the attenuation coefficient of certain organs. Derived values such as the spatial variance within the slice of voxels of a certain bin are also included into the descriptor. The stack of the N_D-dimensional axial slice descriptors is interpreted as a set of N_D 1-D functions whose domain is the (vertical) z level. Then five handcrafted rules are used to decide which of eight different body parts are visible. However, the results are imprecise because no quantitative estimation of the covered body region is performed. Furthermore, Dicken et al. report problems with short scan ranges.

In scene classification, it has recently become popular to measure the similarity of two images by extracting a „bag of features“ from both images. Grauman and Darrell [Grau 05] proposed a distance measure for feature bags that builds a pyramid of histograms of features. They then compare the two histogram pyramids. Lazebnik [Laze 06] adapted this distance measure by first classifying the feature vectors into visual words. The vocabulary is generated in advance by clustering feature vectors that have been extracted from a set of training images. Thus, a visual word corresponds to a class of image patches that have a similar descriptor and similar appearance. For example, a visual word may correspond to blobs, curved edges, or homogeneous regions. Then a spatial pyramid of histograms of the visual words is generated and used in comparing two images.

In [Feul 09b], we introduced the use of histograms of visual words to register stacks of CT image slices. Only the z axis of the volume is considered as it is sufficient for many applications and it leads to a small search space that even allows exhaustive search. The body region of a test volume is estimated by 1-D registration along the longitudinal axis to a set of prototype volumes with known body regions. In order to quantify body regions, patient-specific 1-D “body coordinates” (BC) are introduced. The origin is defined to be at the level of a landmark in the pelvis, and the unit length is set to the distance between a landmark at the clavicle and the pelvis landmark.

In [Feul 11c], we proposed an extension of [Feul 09b]. We introduced an extension of the SURF descriptor to higher dimensions. We also presented two methods for making such a descriptor rotation invariant.

Figure 4.1 shows an overview of the proposed system. For an incoming volume, first the skin of the patient is detected. Independent of this, the axial slices of the volume are regularly divided into small quadratic or cubic patches that also cover neighboring slices. In the next step, a feature vector is extracted from each patch, which is used to classify the patch into a visual word belonging to a predefined vocabulary. The feature vector is a combination of a 2-D or 3-D SURF descriptor and a histogram of the image values. Only patches inside the patient’s skin are considered so as to avoid getting confused by the environment, e.g. the table the patient lies on, or the air surrounding the patient. A spatial pyramid of histograms is then generated from the visual words detected in a slice of the volume. This pyramid serves as a descriptor of the slice and it is computed for all slices of the volume. Thus, the result is a stack of histograms. A set of training volumes with known annotations of the pelvis and clavicle landmarks are processed in the same way, resulting in a set of prototype histogram stacks. The vocabulary of visual words is generated in advance by clustering the feature vectors extracted from the training volumes. In the end, the body portion of the input volume is determined by 1-D registration of its histogram stack with respect to the prototype stacks with known body regions. Generally a single prototype would be enough, but using more than one leads to more robust results.

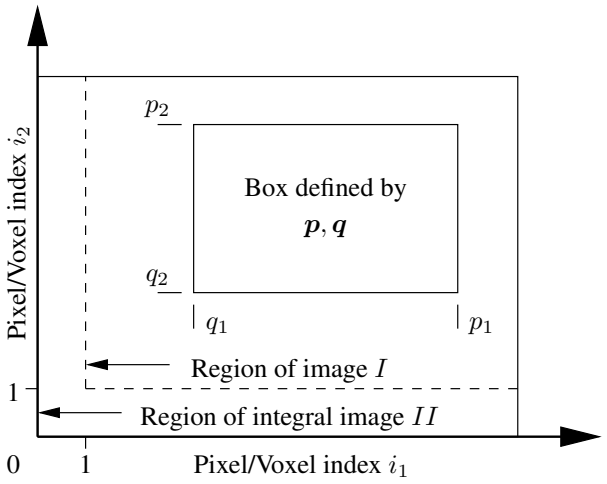


Figure 4.2: Illustration of an image region with upper bounds $\boldsymbol{p}$ and lower bounds $\boldsymbol{q}$. The regions on which image I and its integral image II are defined for two dimensions ($N = 2$) are also shown. Both $\boldsymbol{p}$ and $\boldsymbol{q}$ are pixel/voxel indices. The origin of the image is $(1, 1)$, while the origin of the integral image is $(0, 0)$.

The structure of the rest of this chapter is as follows: Sections 4.2 and 4.3 describe the extension of the SURF descriptor to N dimensions and two approaches to make it rotation invariant. In section 4.4 the extraction of visual words and the histogram generation are explained. Section 4.5 is on the registration of the histogram stacks. Section 4.6 describes experiments and presents results, and section 4.7 concludes the chapter.

4.2 N-SURF

When dealing with 3-D volumetric images, it is desirable to also use a 3-D descriptor. So far, SURF features have only been defined for two dimensions (see section 3.4). We propose an N-SURF descriptor, which is an extension of SURF to an arbitrary number N of dimensions. For $N = 2$, N-SURF simplifies to standard SURF. The formulation for an arbitrary N leads to a uniform notation and enables application for $N > 3$, for instance in case of 3-D+t temporal volumetric sequences.

N-D Haar filters

First, the concept of Haar filters and rectangle filters is generalized to N dimensions. As Haar filters are combinations of rectangle filters, an N-D Haar filter becomes a combination of N-D (hyper)-cuboid filters. When applying a hyper-cuboid-filter, we need to compute the integral C over an axis-aligned (hyper)-cuboid which is described by its upper bounds $\boldsymbol{p} = (p_1 \dots p_N)$ and its lower bounds $\boldsymbol{q} = (q_1 \dots q_N)$ with $p_i \geq q_i, i = 1 \dots N$. As we are dealing with discrete images, p_i and q_i are voxel indices of the i-th dimension. See Figure 4.2 for an example of a box described by $\boldsymbol{p}$ and $\boldsymbol{q}$ for $N = 2$. In this case, $\boldsymbol{p}$ is the

upper right corner of the box and $\boldsymbol{q}$ is the lower left corner. When I is an N-dimensional image, the sum of voxels C inside the hyper-box is

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{i_1=q_1+1}^{p_1} \sum_{i_2=q_2+1}^{p_2} \dots \sum_{i_N=q_N+1}^{p_N} I(\boldsymbol{i}) \tag{4.1}$$

with $\boldsymbol{i} = (i_1 \dots i_N)$.

Just like in the 2-D case, the sum can be efficiently computed with the help of an integral image II

$$II(\boldsymbol{p}) = \begin{cases} \sum_{i_1=1}^{p_1} \sum_{i_2=1}^{p_2} \dots \sum_{i_N=1}^{p_N} I(\boldsymbol{i}) & \text{if } p_j > 0 \, \forall j \in \{1 \dots N\} \\ 0 & \text{else} \end{cases} \tag{4.2}$$

$$= \begin{cases} C(\boldsymbol{p}, \boldsymbol{0}) & \text{if } p_j > 0 \, \forall j \in \{1 \dots N\} \\ 0 & \text{else.} \end{cases} \tag{4.3}$$

Each voxel $\boldsymbol{p}$ of this integral image contains the sum of voxels of the original image I that lie inside the axis-aligned (hyper)-cuboid that has the origin and $\boldsymbol{p}$ as two opposite corners.

Theorem 1. Let $T(N, d)$

$$T(N, d) = \left\{ \boldsymbol{t} \in \{0, 1\}^N \,\middle|\, \sum_{i=1}^{N} t_i = d \right\} \tag{4.4}$$

denote the set of permutations of an N-dimensional vector that contains d ones and $N - d$ zeros. Let $C_N(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q})$ be

$$C_N(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q}) = II \begin{pmatrix} (1 - t_1)p_1 + t_1 q_1 \\ \vdots \\ (1 - t_N)p_N + t_N q_N \end{pmatrix}, \tag{4.5}$$

where II denotes the integral image of image I. Then the sum $C(\boldsymbol{p}, \boldsymbol{q})$ of the image-values inside a hyper-box with upper bounds $\boldsymbol{p}$ and lower bounds $\boldsymbol{q}$ is

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{d=0}^{N} (-1)^d \sum_{\boldsymbol{t} \in T(N,d)} C_N(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q}). \tag{4.6}$$

This theorem is proven in appendix A.

Since the number of permutations is $|T(N, d)| = \binom{N}{d}$ and

$$\sum_{d=0}^{N} \binom{N}{d} = 2^N, \tag{4.7}$$

the sum $C(\boldsymbol{p}, \boldsymbol{q})$ can be computed with a complexity of $O(2^N)$. Though the sum grows exponentially in the number of dimensions, it does not depend on the size of the rectangle filter and is, therefore, very efficient for small N. The integral image II can be precomputed efficiently by first computing the integral images of all (hyper)-slices and then summing over the outermost dimension.

N-D descriptor

As in the 2-D case, the image is sampled on a regular grid around an interest point. The samples are split into b bins per dimension, resulting in b^N bins. For each sample, the gradient is approximated with N Haar filters $c_1 \dots c_N$, which are weighted with an N-D Gaussian centered at the interest point with $\sigma = 10s$. If σ is high, then the gradients computed at different sample points have a similar influence, meaning that there is no special focus on the center of the sampling pattern. If σ is low, only the gradients extracted close to the center of the sampling pattern have influence, and the remaining ones are effectively not used. A value of $10s$ is a reasonable choice. For each bin, a feature vector $\boldsymbol{v}$

$$\boldsymbol{v} = \left(\sum_{i=1}^{N_{SB}} c_1^{(i)}, \dots, \sum_{i=1}^{N_{SB}} c_N^{(i)}, \sum_{i=1}^{N_{SB}} |c_1^{(i)}|, \dots, \sum_{i=1}^{N_{SB}} |c_N^{(i)}| \right) \tag{4.8}$$

is extracted, and the final descriptor is generated by concatenating the vectors from all bins. In (4.8), N_{SB} denotes the number of samples in a bin of the pattern. The final descriptor has a dimension n of

$$n = 2Nb^N. \tag{4.9}$$

4.3 Rotation invariance

To make the descriptor invariant to rotation, standard SURF first assigns a canonical orientation to the interest point where the descriptor is extracted. The sampling pattern is rotated according to this orientation. For each sample point, the gradient is approximated using Haar filters. As the Haar filters can only be extracted efficiently in an axis-aligned orientation, they are computed upright, and the approximated gradient is rotated afterwards into the coordinate system of the sampling pattern. In the 2-D case, the canonical orientation is determined by generating a 1-D angle-histogram from gradients extracted inside a circular region around the interest point. This histogram is then filtered with a rectangle filter (sliding window), and the mode is used as dominant orientation.

This cannot be directly generalized to more than two dimensions, because the mode of the gradient directions fixes only $N - 1$ degrees of freedom (DOF), which is not enough for $N > 2$. In general, an N-D rotation has $\frac{(N-1)N}{2}$ DOF.

As a solution to this problem, we propose two methods for obtaining rotation invariance in three or more dimensions. In both cases, first gradient approximations $\boldsymbol{c}^{(i)}, i = 1 \dots G$ are extracted inside a (hyper-)spherical region with radius $r = 6s$ around the interest point like in the 2-D case. The orientation is then determined from this set of gradients. G is the number of sample points with spacing s that fit into the (hyper-)spherical region.

A convenient representation for an N-D rotation is a rotation matrix. An $N \times N$ matrix $\boldsymbol{R}$ is a rotation matrix if and only if $\det(\boldsymbol{R}) = 1$ and it is orthonormal, meaning that all columns have unit length and are orthogonal to each other.

4.3.1 Variant 1

For the first variant, it is assumed that the gradient vectors $\boldsymbol{c}^{(i)}, i = 1 \dots G$ are normally distributed. The principal component analysis (PCA) is computed on the gradient vec-

tors. Note that PCA analysis of gradients has similarities with the structure tensor for edge and corner detection as described in [Kthe 03], as the spatially averaged structure tensor is similar to the covariance matrix of the gradients if the gradients have zero mean. The eigenvectors $\boldsymbol{u}^{(i)}, i = 1 \dots N$ resulting from the PCA analysis are sorted in descending eigenvalue order. The columns of the rotation matrix $\boldsymbol{R}$ are generated from the eigenvectors $\boldsymbol{u}^{(i)}$. These are already orthogonal, but an eigenvector can point in either of the two directions of its principal axis. To standardize this direction, an eigenvector $\boldsymbol{u}^{(i)}$ is multiplied with -1 if the scalar product of $\boldsymbol{u}^{(i)}$ with the mean gradient $\overline{\boldsymbol{c}}$ is below zero. The eigenvectors with canonical direction are denoted as $\boldsymbol{u}_a^{(i)}$:

$$\boldsymbol{u}_a^{(i)} = \begin{cases} -\boldsymbol{u}^{(i)} & \text{if } \overline{\boldsymbol{c}}^T \boldsymbol{u}^{(i)} < 0 \\ \boldsymbol{u}^{(i)} & \text{else} \end{cases} \quad \text{with } \overline{\boldsymbol{c}} = \frac{1}{G} \sum_{i=1}^{G} \boldsymbol{c}^{(i)}. \tag{4.10}$$

They are normalized to unit length

$$\boldsymbol{u}_b^{(i)} = \frac{\boldsymbol{u}_a^{(i)}}{\left\| \boldsymbol{u}_a^{(i)} \right\|_2}. \tag{4.11}$$

Now the matrix

$$\boldsymbol{R}' = \left(\boldsymbol{u}_b^{(1)} \dots \boldsymbol{u}_b^{(N)} \right) \tag{4.12}$$

is orthonormal, but its determinant $\det(\boldsymbol{R}')$ can be either $+1$ or -1. As a rotation matrix must have a determinant of $+1$, the last column of $\boldsymbol{R}'$ is multiplied with -1 if necessary. The result is called $\boldsymbol{R}$ and is the final rotation matrix.

Note that $\boldsymbol{R}$ is always a valid rotation matrix for all possible gradients $\boldsymbol{c}^{(i)}, i = 1 \dots G$ because the covariance matrix is real and symmetric. Thus, there are N orthogonal eigenvectors, even if some or all of the eigenvalues are zero.

4.3.2 Variant 2

In variant 1, only the covariance matrix of the gradients is used in determining the axes of the rotated coordinate system. Since the gradients are converted to zero-mean as part of the process, the absolute gradient values are not taken into account, although they contain valuable information. For instance, if all gradients $\boldsymbol{c}^{(i)}$ happen to be the same, their covariance matrix is the zero matrix, and the resulting rotation matrix is valid but describes an arbitrary rotation, although different orientations will in general result in different descriptors.

In the second variant, this is solved by taking the normalized mean $\frac{\overline{\boldsymbol{c}}}{\|\overline{\boldsymbol{c}}\|_2}$ of the gradients into account. It is used as the first axis of the rotated coordinate system and as the first column of the rotation matrix $\boldsymbol{R}$. Then, all gradient vectors $\boldsymbol{c}^{(i)}$ are projected onto the $(N-1)$-D (hyper)-plane orthogonal to $\overline{\boldsymbol{c}}$. This cannot be continued in the same way for the remaining dimensions because the projected $(N-1)$-D gradient vectors, denoted by $\boldsymbol{c}_p^{(i)}$, always have zero mean. Therefore, they are now treated similarly to variant 1: The $(N-1)$-D PCA of the projected gradient vectors $\boldsymbol{c}_p^{(i)}$ is computed, and the principal axes are taken as the remaining axes of the rotated coordinate system. The eigenvectors are normalized to unit length. As before, the orientation of the eigenvectors with respect to

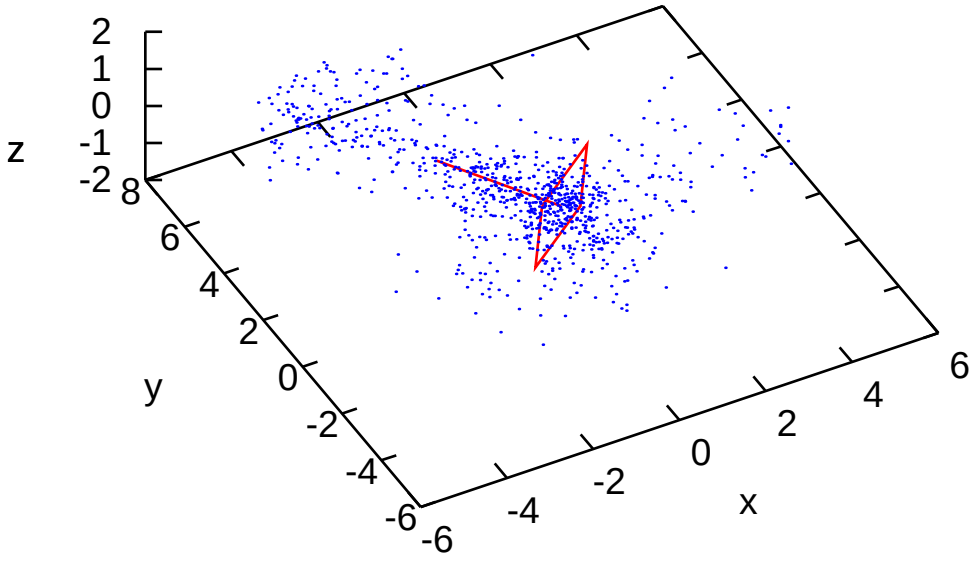


Figure 4.3: Gradient vectors (blue dots) extracted in a spherical image region with assigned orientation according to variant 2 in red. The single red line is the mean gradient vector. The gradients are projected onto the plane of the square. The single red line and the square are axis-aligned to the rotated coordinate system.

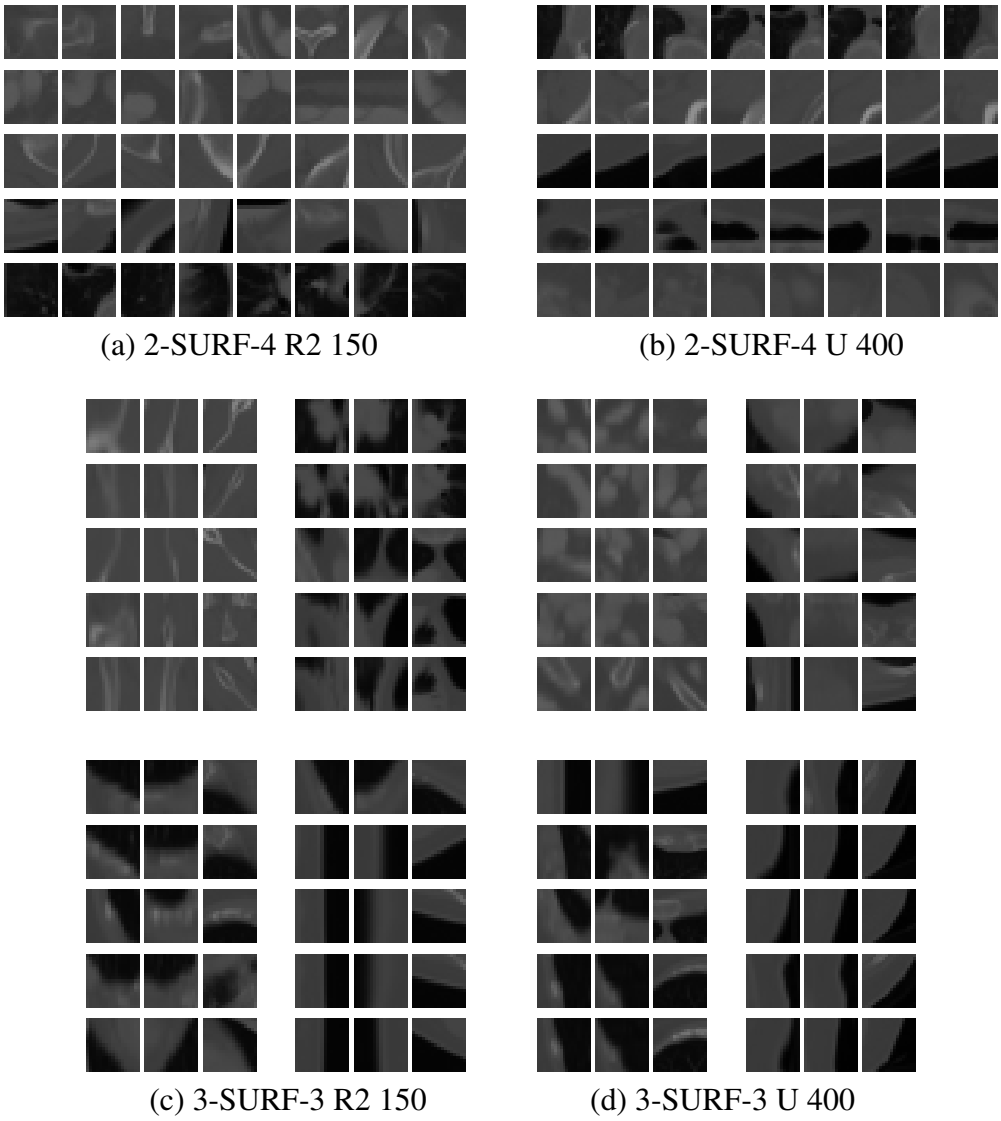
their principal axis can be positive or negative and must be standardized. We cannot use the mean gradient as reference like in variant 1 because it is zero. Instead, we use $\boldsymbol{c}_r$ with

$$\boldsymbol{c}_r = \sum_{i=1}^{G} \boldsymbol{c}^{(i)} \left\| \boldsymbol{c}^{(i)} \right\|_2^2 . \tag{4.13}$$

The columns $2 \ldots N$ of the rotation matrix $\boldsymbol{R}$ are formed by the eigenvectors. Again, an eigenvector is multiplied with -1 if its scalar product with $\boldsymbol{c}_r$ is less than zero, except for the last eigenvector, which is multiplied with -1 if the determinant of the rotation matrix $\boldsymbol{R}$ was -1 otherwise. Variant 2 is illustrated in Figure 4.3.

4.4 Histograms of visual words

The concept of describing images using a visual vocabulary has been successfully used in the past in data mining, scene classification and object recognition. The visual vocabulary consists of visual words, which are primitive patches used to characterize an ensemble of images. In practical applications, the visual words are learned from the image ensemble and often include straight lines, corners, uniform patches, holes or certain textures.



(a) 2-SURF-4 R2 150 (b) 2-SURF-4 U 400

(c) 3-SURF-3 R2 150 (d) 3-SURF-3 U 400

Figure 4.4: Example images shown for selected visual words taken from four different vocabularies. (a) and (b) show examples from 2-D vocabularies. Each row corresponds to one visual word. The vocabulary (a) was generated using a 2-D SURF descriptor with rotation invariance of type 2. (b) shows example images for five visual words with rotation invariance turned off ("upright"). (c) and (d) show each examples for four different visual words. Now the image patches are cubes instead of 2-D regions. A cubic image region is visualized by three axis-aligned cross-sections, displayed in a row. For (c), rotation invariance of type 2 was turned on. (d) was generated with an upright descriptor. The size of the vocabulary was 150 in the rotation invariant cases and 400 in the upright cases.

Bhattacharya et al. [Bhat 05] described retina images using a visual vocabulary. This description was used to distinguish image classes and to highlight parts of the image that are characteristic for their class. Duygulu et al. [Duyg 02] labeled image regions with keywords from a predefined vocabulary of nouns in order to automatically generate an image description and to recognize objects.

In this chapter, we use visual words to describe an axial CT slice in the form of a spatial pyramid of histograms of visual words. Two axial CT slices are compared by measuring the similarity of the two descriptors. This is done using a technique called *spatial pyramid matching*. In the remainder of this section, we explain the matching technique and how this descriptor is obtained from a CT slice.

4.4.1 Spatial pyramid match kernel

The *spatial pyramid matching* scheme that is used in this work to measure the similarity of CT slices was derived in [Laze 06] from the *pyramid match kernel* proposed in [Grau 05]. The pyramid match kernel was proposed as an efficient similarity measure for two feature sets, or point clouds in general. Let X and Y denote two sets of n-dimensional feature vectors. The pyramid match kernel places a pyramid of grids (or histograms) over the feature space, with a grid spacing that doubles from a finer to a coarser level. At each level of the pyramid, features are said to match if they fall into the same grid cell, and matches at finer level are weighted higher because feature vectors that are close do on average match at finer grid levels.

Let l denote the current pyramid level with $l \in \{0 \dots L\}$. The grid at level l has 2^l bins in each dimension and $D_l = 2^{nl}$ bins in total. Let further H_X^l and H_Y^l denote the histograms of the feature vectors in X and Y at level l, respectively, and $H_X^l(i)$ and $H_Y^l(i)$ the number of features in X and Y that fall into the ith bin of the histogram at level l. The number of matching features at level l is then given by the histogram intersection [Swai 91]

$$\mathcal{I}(H_X^l, H_Y^l) = \sum_{i=1}^{D_l} \min\left(H_X^l(i), H_Y^l(i)\right). \tag{4.14}$$

In the remainder of the section, $\mathcal{I}(H_X^l, H_Y^l)$ is abbreviated with $\mathcal{I}^l$.

The matches $\mathcal{I}^l$ at level l do always include the matches at finer levels. The number of new matches at level l with respect to the finer level $l+1$ are given by $\mathcal{I}^l - \mathcal{I}^{l+1}$ for $l \in \{0 \dots L-1\}$. New matches at level l are weighted with $\frac{1}{2^{L-l}}$, which is inversely proportional to the grid spacing.

The pyramid match kernel of [Grau 05] is then the weighted sum of new matches

$$\kappa^L(X,Y) = \mathcal{I}^L + \sum_{l=0}^{L-1} \frac{1}{2^{L-l}}(\mathcal{I}^l - \mathcal{I}^{l+1}) \tag{4.15}$$

$$= \frac{1}{2^L}\mathcal{I}^0 + \sum_{l=1}^{L} \frac{1}{2^{L-l+1}}\mathcal{I}^l. \tag{4.16}$$

In [Laze 06], it was proposed to first quantize the feature vectors into M classes. Then, pyramid matching is carried out in the image coordinate space in order to include spatial neighborhood. It is assumed that only features of a certain class can match. Pyramid

matching is then carried out M times, once for each class, or channel. Let X_i denote the set of 2-D vectors of the image coordinates of the features of class i in X, and let Y_i be defined analogously. Then the spatial pyramid match kernel is defined as

$$\mathcal{K}^L(X, Y) = \sum_{i=1}^{M} \kappa^L(X_i, Y_i). \tag{4.17}$$

By using that κ^L is a weighted sum of histogram intersections, and due to $c \min(a, b) = \min(ca, cb)$ for $c, a, b \geq 0$, (4.17) can be computed as the histogram intersection of two long histograms that are formed by concatenating weighted M dimensional histograms.

4.4.2 Sampling

In the first step of extracting such a spatial pyramid of quantized features from a slice, it is densely sampled on a regular grid with a sample spacing of 10mm. An alternative to a fixed sampling grid is to detect key locations in the image, for example minima and maxima in scale space as suggested by Lowe [Lowe 99]. However, according to Fei-Fei and Perona [Fei 05], better results have been reported for a regular dense sampling.

4.4.3 Patient detection

Since we are only interested in the patient and not the surrounding air or other objects like the table that is usually visible in a CT slice, we first run a simple detector that segments the patient in a slice. First, a binary mask that has the same dimensions like the slice is initialized with ones. Then, each row and each column is scanned from outside to inside, from both directions. A pixel is assumed to be the skin if a certain number n_s of successors and the pixel itself are above a threshold of -600 HU. n_s is set to 3mm divided by the voxel spacing for scans in dorsal direction, and to 10mm divided by the voxel spacing for scans in other directions. The reason for the difference is that sometimes the chest wall is thinner than 10mm. Pixels outside the skin are set to zero. The result is a mask that marks each voxel either as “patient” (1) or “environment”(0). This simple algorithm proved to be fast and effective for rejecting the air surrounding the patient and also the table s/he lies on.

4.4.4 Feature extraction

For all sample points inside the patient, a feature vector is computed, which consists of an eight bin histogram of the Hounsfield units and a 2-SURF or a 3-SURF descriptor. In the 2-D case, the descriptor is computed from the axial slice. In the 3-D case, voxels from neighboring slices are also considered.

As SURF descriptors were designed to be invariant to illumination changes that often cause problems in computer vision, they do not make use of absolute intensities. However, in CT images absolute intensities are reliable. In order to use this information, the N-SURF descriptor is extended with the Hounsfield histogram, which is scaled to fit the mean values of the N-SURF descriptor entries. Descriptors are computed at a fixed scale of $s = 1$, which corresponds to a descriptor window size of 20×20 pixel.

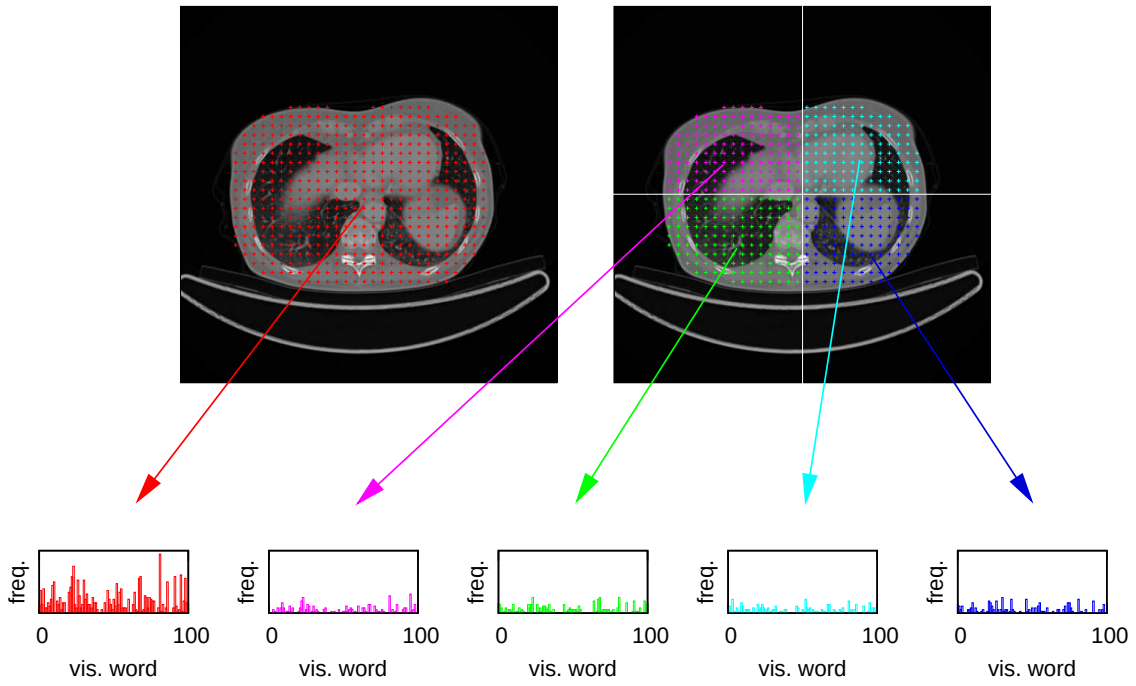


Figure 4.5: Illustration of the spatial pyramid of histograms used to describe an axial CT slice, here displayed with two levels. At the first level (left), a histogram of quantized features (visual words) is generated for the whole slice. Here, the vocabulary size is 100. At level two (right), the image slice is split into four parts, and for each one, a histogram is generated. The quantized features remain the same, meaning that the sum of the four right histograms equals the left (red) one.

4.4.5 Visual words

The extracted feature vectors are now quantized and classified into a set of visual words. The vocabulary is represented by a prototype feature vector for each word, and a nearest neighbor classifier maps a new feature vector to the nearest prototype. The distance of two feature vectors is measured using the L_2 norm. To generate the vocabulary, a random subset of feature vectors is extracted from a set of training images. The K-Means algorithm is used in finding clusters. The cluster centers are chosen as the vocabulary.

Figure 4.4 shows example images from four different vocabularies, generated using a 2-D or 3-D descriptor with rotation invariance turned on or off. In the 2-D case, image patches from five visual words are displayed for the rotation invariant case (a) and the upright case (b). Image patches in one row belong to the same visual word. With a rotation invariant descriptor (a), the orientation of the patches within a word is arbitrary, while in the upright case (b), patches of a word share a similar orientation. In Figure 4.4 (c,d), images patches from two 3-D vocabularies are visualized. One 5×3 block of images corresponds to one word. Each row shows an axial, coronal and sagittal cross-section of a cubic image patch. In (c) the descriptor was made rotation invariant using method 2, and in (d) an upright descriptor was used. Generally, the number of clusters in features space, which equals the vocabulary size, needs to be higher in the upright case in order to separate patches showing different tissue types.

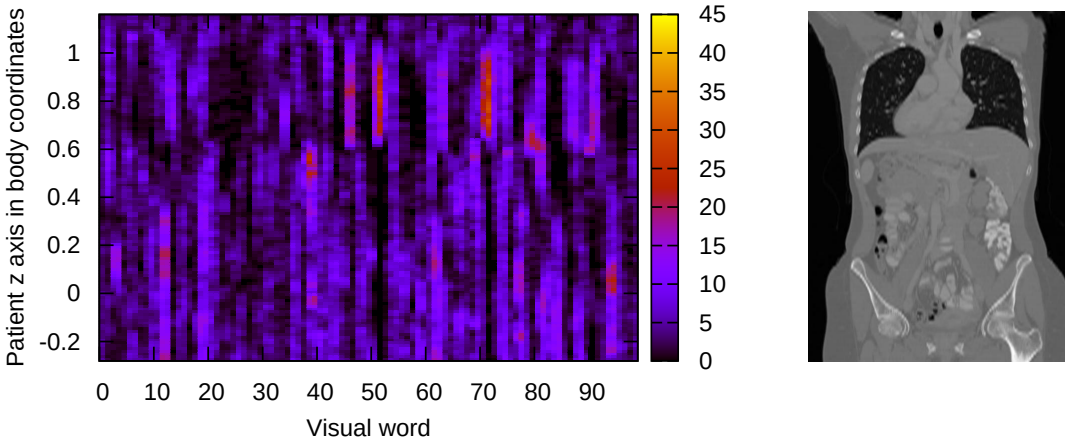


Figure 4.6: Histograms of visual words along with a coronal section of the volume it was generated from. Salient are especially the visual words that correspond to the lung region. The image is best viewed in color.

4.4.6 Histograms of visual words

The visual words (or quantized features) extracted from a slice are now aggregated in a spatial pyramid of histograms, which serves as a descriptor of the slice. Figure 4.5 illustrates a spatial pyramid with two levels. At the root level, a histogram of quantized features inside the patient body is computed for the entire slice (level $l = 0$). At the next level ($l = 1$), four histograms are computed for different parts. Note that the features only need to be computed and quantized once for each sample, independent from the number of pyramid levels. A slice is finally described by a concatenation of all the histograms. In the shown example, the vocabulary size, which is the histogram length, is 100, and the slice descriptor is therefore of dimension 500. The spatial pyramid match kernel weights deeper pyramid levels higher (4.16). In this work, we however use one level below the root ($L = 1$), and as the weights for $l = 0$ and $l = 1$ are the same, the weighting is omitted here. In Figure 4.6, a stack of histograms of visual words is shown together with a coronal section of the original volume (for only one pyramid level).

4.5 Histogram matching

Consider two 2-D images j and k that are axial slices taken from two 3-D volumes I_j and I_k at level z_j and z_k

$$j(x, y) = I_j(x, y, z_j) \tag{4.18}$$
$$k(x, y) = I_k(x, y, z_k). \tag{4.19}$$

Two slices j and k are now compared by matching their spatial pyramids of histograms according to (4.17). If H_j and H_k denote the corresponding concatenated histograms con-

sisting of $5M$ bins each ($L = 1$), the spatial pyramid match kernel can be computed (see section 4.4.1) as their histogram intersection

$$\mathcal{I}(H_j, H_k) = \sum_{i=1}^{5M} \min(H_j(i), H_k(i)). \tag{4.20}$$

For a fixed number of samples per image, $\mathcal{I}(H_j, H_k)$ is, up to a normalizing factor, equivalent to using one minus the sum of absolute differences

$$d(j, k) = \sum_{i=1}^{5M} |H_j(i) - H_k(i)| \tag{4.21}$$

that is used as a distance measure in this work. Other distance measures, such as the L_2 norm or the Kullback-Leibler divergence, are conceivable as well, although they do not fit as well into the spatial pyramid match kernel. We leave their evaluation for future work.

In the following subsection we compare two different methods for registering two slice stacks $J = j_1, \ldots, j_{n_J}$ and $K = k_1, \ldots, k_{n_K}$ along the z axis, which is discretized with a 4mm resolution.

4.5.1 Rigid matching

The first method is a rigid registration. An objective function $f(z)$ measures the average distance of the slices given a longitudinal offset z:

$$f(z) = \frac{1}{i_{\text{max}} - i_{\text{min}} + 1} \sum_{i=i_{\text{min}}}^{i_{\text{max}}} d(k_i, j_{i+z}). \tag{4.22}$$

In (4.22), $i_{\text{min}}, \ldots, i_{\text{max}}$ denotes the range of overlapping slices. The objective function is only evaluated in the z range that results in an overlap of at least 80% between the two stacks J and K.

Because a single evaluation of the objective function f is computationally inexpensive and the search space is only one-dimensional, exhaustive optimization is feasible. Figure 4.7 shows $f(z)$ for two test stacks $K_{1,2}$ of different size and four reference histogram stacks $J_{1\ldots4}$.

After exhaustive optimization, a set of candidates $C = \{c_1, c_2, \ldots, c_{|C|}\}$ is generated from f by finding local optima. The reason is that especially for volumes with a small number of slices, it occasionally happens that the global optimum is not the right solution. However, the correct solution is almost ever located in a valley. Thus, we associate a weight w_i with each candidate c_i. The weight w_i is computed from the objective function at c_i and its second derivatives:

$$w_i = 2 \left(\sum_{s=0}^{2} f(z - c_i) * g_s(z) \right) - f(c_i). \tag{4.23}$$

Here, $*$ denotes convolution, g_0 is a filter kernel to compute the second derivative, and $g_{s+1}(z) = g_s(\frac{z}{3})$ is scaled with a factor of 3 relative to g_s.

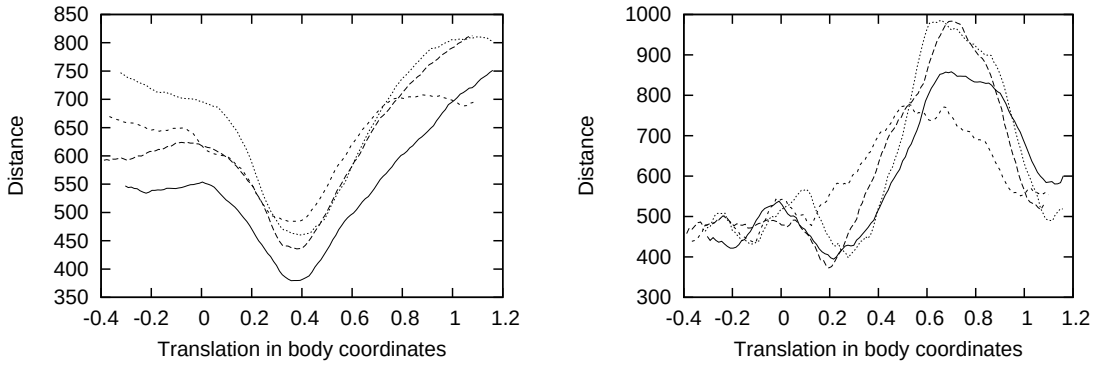


Figure 4.7: The objective function f of four different prototype volumes. Left: Test volume with 114 slices. Right: Test volume with 10 slices from the abdomen. For the large volume, one clear minimum exists. For the small stack identification of a minimum is more ambiguous. But still in 3 out of 4 cases, the global optimum is close to the correct location (at approx. 0.2BC).

In order to achieve robust results, a test volume is registered with several prototype volumes. As final registration result, the candidate with the best weight is selected. Note that, though the described method does not explicitly handle scale variations, it implicitly addresses the issue through the scale variations of the training data. For instance, a test volume of a tall patient will generally fit better to tall patients in the training set.

4.5.2 Non-rigid matching

For comparison, additionally to the rigid matching, non-rigid matching based on dynamic time warping (DTW) was used for registering a test volume with a prototype volume.

Now the objective function $f_d(z_0, z_1)$ takes two arguments, which are the longitudinal coordinates of the lower and the upper slice of the test stack. In each evaluation of f_d, the top and bottom slice remains fixed and only the positions of the intermediate slices of the test stack are varied. As before, the similarity of two slices j, k is measured using the distance function $d(j, l)$. The costs of the cheapest match of the intermediate slices is computed using dynamic time warping and returned by f_d. The objective function f_d is evaluated for every pair z_0, z_1 of upper and lower z-coordinates of the test stack, which are inside the z-range of the reference patient and satisfy

$$\left| \frac{z_1 - z_0 - \Delta_z}{\Delta_z} \right| < 0.15, \tag{4.24}$$

where Δ_z is the height of the test stack, measured in mm. This means that a test stack is never shrunk or enlarged more than by 15%. In Figure 4.8, the cheapest warp is visualized in the DTW cost matrix. The columns of the matrix are the slices of the test stack, and the rows correspond to the slices of the reference stack in the range between z_0 and z_1.

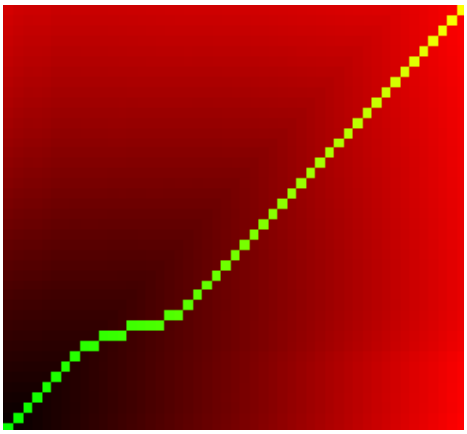


Figure 4.8: Example of a cost matrix for dynamic time warping. The horizontal axis corresponds to the test stack, and the vertical axis to the section of reference stack between z_0 and z_1. Black denotes low costs, red high costs. The cheapest warp that registers two slice stacks is shown in green.

	U	R1	R2
2-SURF-4	9.0	10.8	11.0
3-SURF-3	102.2	148.5	154.3

(a)

b	2	3	4
3-SURF-b U	53.7	102.2	236.2

(b)

Table 4.1: Computation times in seconds for different variants of the method. Top: Comparison between a 2-D and 3-D descriptor computed either upright (U), with rotation invariance of type 1 (R1) or type 2 (R2). Comparison of three upright 3-D descriptors with a different number of sub-bins b per dimension.

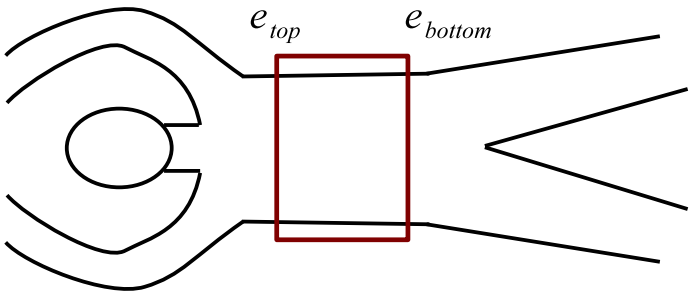


Figure 4.9: Illustration of the error measure used. For a single registration, the error was measured at the top and bottom of the test volume.

Num. partitions/ size in mm*	10/44	5/86	3/140
2-SURF-4 U 300	18.08 ± 25.81	15.18 ± 15.16	15.30 ± 15.06
3-SURF-3 U 400	18.74 ± 29.13	15.59 ± 14.82	15.57 ± 14.88
2-SURF-4 R1 300	20.33 ± 33.95	17.18 ± 19.86	15.38 ± 14.30
2-SURF-4 R2 150	20.13 ± 28.53	17.18 ± 20.11	15.14 ± 15.16
3-SURF-3 R1 150	20.01 ± 28.74	16.70 ± 16.22	15.43 ± 14.56
3-SURF-2 R2 100	21.79 ± 32.79	19.35 ± 23.10	15.12 ± 13.83
3-SURF-3 R2 150	20.86 ± 34.09	17.13 ± 20.65	15.46 ± 15.00
3-SURF-4 R2 200	19.44 ± 25.30	16.29 ± 16.35	14.55 ± 13.73
3-SURF-3 U 200 F	23.38 ± 48.28	18.22 ± 26.00	15.53 ± 14.00
3-SURF-3 U 200	19.69 ± 36.37	17.33 ± 28.14	14.61 ± 13.97
3-SURF-3 U 100 DTW	21.43 ± 30.80	19.25 ± 22.94	18.31 ± 16.41
3-SURF-3 U 100	19.79 ± 25.71	16.55 ± 16.21	15.42 ± 13.91
Hounsfield	48.02 ± 82.39	38.64 ± 69.64	35.54 ± 76.64

Num. partitions/ size in mm*	2/206	1/427	average
2-SURF-4 U 300	13.60 ± 9.06	15.46 ± 9.16	**15.52** ± 14.85
3-SURF-3 U 400	12.94 ± 8.95	14.68 ± 10.27	**15.50** ± 15.61
2-SURF-4 R1 300	13.30 ± 9.45	15.89 ± 10.65	**16.42** ± 17.64
2-SURF-4 R2 150	12.80 ± 9.47	15.20 ± 10.29	**16.09** ± 16.71
3-SURF-3 R1 150	12.99 ± 9.02	17.09 ± 12.12	**16.44** ± 16.13
3-SURF-2 R2 100	13.01 ± 9.48	16.50 ± 10.61	**17.15** ± 17.96
3-SURF-3 R2 150	13.58 ± 9.99	16.66 ± 11.49	**16.74** ± 18.24
3-SURF-4 R2 200	13.73 ± 9.49	17.12 ± 10.27	**16.23** ± 15.03
3-SURF-3 U 200 F	14.80 ± 12.88	15.07 ± 10.37	**17.40** ± 22.30
3-SURF-3 U 200	12.91 ± 9.34	15.57 ± 10.93	**16.02** ± 19.75
3-SURF-3 U 100 DTW	17.24 ± 10.71	20.12 ± 14.10	**19.27** ± 18.99
3-SURF-3 U 100	13.23 ± 8.00	15.94 ± 10.88	**16.19** ± 14.94
Hounsfield	28.99 ± 56.73	17.00 ± 10.43	**33.64** ± 59.17

Table 4.2: Results of accuracy evaluation. Each row corresponds to a different method. 2-SURF-4 means 2-D SURF with 4 sub-bins b per dimension. U means upright, R1 is the first approach for rotation invariance, R2 the second one. The final number is the number of clusters. A trailing F stands for a flat pyramid which has only one level, and DTW means that dynamic time warping was used for the registration. *Size of partition in mm is an approximate value, averaged over patients.

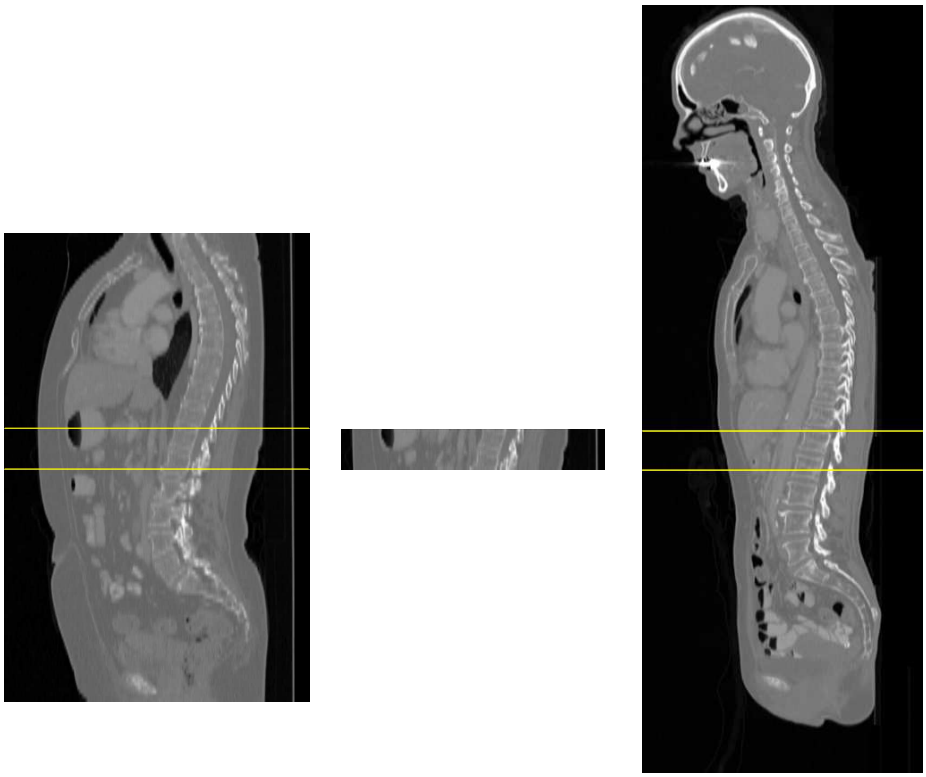


Figure 4.10: Example of a registration result. Middle: Sagittal slice through the test sub volume of which the body region is to be determined. It consists of 10 axial slices with a slice thickness of 5mm and shows a portion of the abdomen. Left: True position in the original volume from which the sub volume was cropped. Right: Sagittal slice through a volume with known body coordinates. The horizontal lines show the estimated body region covered by the test sub volume.

4.6 Results

Registration accuracy was evaluated using 84 CT volume scans of the thoractic and abdominal region. For all datasets, annotations of landmarks at the clavicle and the pelvis were available. They served as ground truth for the body coordinate system, marking the levels zero and one. In between, linear interpolation was used to generate ground truth values for the body coordinates. All datasets were resampled to an isotropic resolution of $2 \times 2 \times 2\,\mathrm{mm}^3$ and descriptors were generated for every second axial image slice.

Three fold cross validation was used to separate the datasets in test and prototype volumes. Registration was performed with slice stacks of five different sizes: A test stack was always partitioned into ten, five, three, two and one pieces, resulting in $10+5+3+2+1 = 21$ registrations per fold and test volume.

The error of a single registration was measured at the top and the bottom of the test volume (see Figure 4.9). The average of the absolute error values

$$e = \frac{1}{2}\left(|e_{\text{top}}| + |e_{\text{bottom}}|\right) \tag{4.25}$$

was taken as the final error e. Table 4.2 shows the results of the cross validation. The columns show the registration accuracy for the five different test volume heights. Each row

shows results for a different method. The mean error in mm is displayed along with the standard deviation. Cross-validation was run for 2-D and 3-D descriptors with two, three and four sub-bins per dimension (see Figure 3.5 (b)), with rotation invariance of type 1, type 2 or with an upright descriptor, for 50, 100, 150, 200, 300, 400 and 800 clusters. In rows one to eight, the number of clusters that gave best results is displayed. For example, a 2-SURF-4 U descriptor worked best with 300 clusters. Results for 50 and 800 clusters don't show up in the table because they were never among the best performers.

Comparing the 2-D with the 3-D descriptor, average accuracy was slightly better for 3-D in the upright case (15.50mm for 3-SURF-4 U vs. 15.52mm for 2-SURF-4 U). While the 2-D descriptor worked better for smaller test stacks of 4.4cm and 8.4cm, the 3-D descriptor performed better for larger test stacks of 20.6cm and 42.7cm. A possible explanation is that the 3-D descriptor takes into account the neighboring slices, which makes it more descriptive but lowers the resolution in z direction. When rotation invariance was turned on, the 2-D descriptor performed better.

The upright descriptors performed clearly better than the rotation invariant ones (rows 1–2 vs. 3–8). The reason is probably that patients are almost always lying in the same position on their back and thus the orientation of an image patch contains valuable information which is lost when the descriptor is made rotation invariant. But we see that the upright descriptors require more clusters in the feature space: They performed best with 300–400 clusters (rows 1–2), while the rotation invariant descriptors performed best with 100–300 clusters (rows 3–8). This means that the vocabulary of visual words is longer and therefore also the concatenated histograms which describe an image slice. Comparing the two approaches to make a descriptor rotation invariant, the second one (R2) worked better in the 2-D case, and the first one (R1) was more accurate in 3-D. A possible explanation is that in 2-D, a rotation has only one degree of freedom, and therefore the mean gradient used by R2 suffices for determining the angle.

In rows 6–8, the number of bins b per dimension of the SURF sampling pattern is varied (see Figure 3.5). For $b = 2, 3, 4$, according to (4.9), the length of the 3-D descriptor is 48, 162 and 384, respectively. The mean error decreased for higher b. While the error was 17.15mm for $b = 2$, it dropped to 16.74mm for $b = 3$ and to 16.23mm for $b = 4$. However, the time needed to extract a descriptor is in $O(b^N)$, which means it is more than twice as expensive to compute a 3-SURF-4 instead of a 3-SURF-3 descriptor.

The accuracy, depending on whether a spatial pyramid is used for the matching or not, is shown in rows 9–10. In the flat case, denoted with a trailing F, an image slice is not split into four subregions. The average mean error dropped from 17.40mm to 16.02mm when a spatial pyramid was used. While the difference is small and the flat approach is even slightly better for larger test volumes of 42.7cm height, the spatial pyramid based approach works considerably better for smaller test volumes of 4.4cm height. Here, the mean error dropped from 23.28mm to 19.69mm. In the results presented so far, two volume images were always registered rigidly. Lines 11–12 compare the rigid registration with the non-rigid version which is based on dynamic time warping, denoted with DTW. In the experiments, the rigid registration worked better than the non-rigid independent of the test volume size. The problem with dynamic time warping is that it often generates unnatural warps in order to match axial slices that happen to have similar descriptors but belong to different body regions. For instance, only the abdominal region is stretched, and the

remaining regions are unchanged. However, such nonlinear deformations are rare in nature and the missing constraint leads to false matches.

For comparison, accuracy was also measured for an approach that simply takes a 1024-bin histogram of the Hounsfield intensities as a descriptor of an axial slice. The results are shown in the last row of Table 4.2. The visual word based approach performed clearly better than the method using an intensity histogram.

Figure 4.10 shows an example of the algorithm's output. The input is a portion of the abdomen of 10cm height. To visualize the result, another volume shown at the right side was annotated with body coordinates. The horizontal lines on the right indicate the estimated body region. The horizontal lines on the left show the true position in the original volume.

As the proposed algorithm is deterministic, its computation time was only benchmarked on a single dataset of 100 slices and using 28 prototype volumes. Results measured on a standard PC with 2.2GHz CPU are shown in Table 4.1. Displayed is the total time needed in seconds for different descriptors. The values include 23ms needed for patient detection and 2.07s for exhaustive optimization, which are both independent of the descriptor. The 2-D descriptors can be computed fast. When using a 2-D upright descriptor, the algorithm takes 9s in total to estimate the portion of the body. With a rotation invariant descriptor, it takes 2s longer. The 3-D descriptors are considerably more expensive to compute. Here, the algorithm takes between 102.2s and 154.3s, depending on whether rotation invariance is turned on. In Table 4.1 (b), the computation time is shown for 3-D upright descriptors of different dimensions, which depends on the number b of sub bins per dimension. For a 48-dimensional descriptor ($b = 2$), the algorithm takes 53.7s, while for 384 dimensions ($b = 4$), it takes more than four times longer (236.2s). Parallelization of the algorithm is straightforward. We leave this for future work.

4.7 Conclusion

This chapter presents a method for estimating the body region of a CT volume image. It is based on 1-D registration of histograms of visual words, which serve as a description of a CT slice.

As part of this work, the SURF descriptor was generalized to N dimensions. It was used in generating the vocabulary of visual words. Different variants of the descriptor were compared. Results show that upright descriptors perform better than rotation invariant ones. 2-D and 3-D upright descriptors perform equally well. As 2-D descriptors are simpler and can be more efficiently computed, we propose the use of 2-D upright SURF descriptors for estimating the body region. In such a setup, an estimation with an average error of 15.5mm can be computed in 9s. This error can be considered as a good result. As we are registering different subjects with each other, we have to deal with considerable anatomical inter-patient variations that exist in the thoracic and abdominal regions. This limits the accuracy because finding a level along the longitudinal axis of a patient that corresponds to a certain level in another patient may become ambiguous.

Besides automatic initialization of further processing steps as presented in the next chapters, possible applications are also automatic labeling of images for the purpose of semantic image search. The 3-D descriptors as described here may also be used for point matching tasks, which is the classical application for SURF and SIFT. 3-D and 4-D SURF

descriptors may be especially useful for finding point correspondences in 2-D sequential data, volumetric data, or even sequential 3-D volumetric data.

Chapter 5

Automatic segmentation of the esophagus in 3-D CT scans

The work presented in this chapter was published by the author of this thesis first in [Feul 09a, Feul 10b] and later as an extended version in [Feul 11a].

5.1 Motivation

During oncological examinations of the chest, radiologists are particularly interested in the region around the trachea and the esophagus [Duwe 05]. These are natural gateways into the body and therefore often surrounded by lymph nodes, which need to be examined for all types of cancer. CT scans of the thorax are common practice for diagnosis and in order to assess whether treatment is effective. While the trachea is clearly visible in CT, the esophagus is much harder to see and can easily be confused with other structures, which is one reason that makes the interpretation of the CT images tedious. An automatic segmentation can serve as a guideline and provide valuable overview to a physician.

A segmentation is also useful for therapy planning. Atrial fibrillation, which is a major cause of stroke, can be treated with an ablation therapy in the heart. During this intervention, however, there is a small risk of an atrial-esophageal fistula. Then, air from the esophagus can enter the left atrium, which normally leads to the death of the patient [Papp 04]. A preoperative segmentation of the esophagus can be useful for intervention planning.

Automatic esophagus segmentation is a challenging problem. The wall of the esophagus consists of muscle tissue, which has even on contrasted CT scans a low contrast to vessels, other muscles and lymph nodes. Shape and appearance can vary a lot. It appears solid if it is empty, but it can also be filled with air, remains of orally given contrast agent, or both. Even for a human, it is often impossible to accurately delineate the boundaries given only a single slice. Figure 5.1 shows two examples along with manual ground truth segmentation.

In [Rous 06], a method is described which combines a spatial prior of the esophagus centerline with a histogram based appearance model. The centerline is extracted using a shortest path algorithm. Then, ellipses are fitted into axial slices by optimizing an energy function that is again histogram based and also has a regularization term for smooth transitions between neighboring slices. The method is semiautomatic and requires two manually placed points on the centerline and also a segmentation of the left atrium and the aorta as

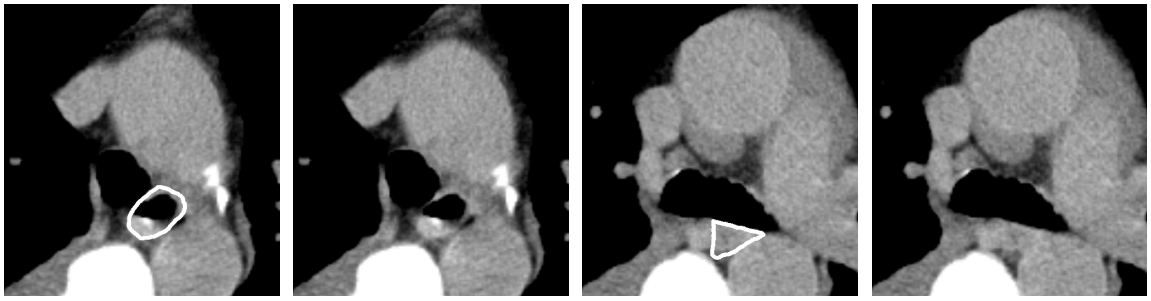


Figure 5.1: Two axial slices with and without manual ground truth segmentation displayed as white contours. In the right example, it is hardly possible to accurately delineate the boundary given only one slice.

input. In [Kuru 10], another semiautomatic segmentation method is proposed which also uses a spatial prior of the esophagus centerline. The prior is estimated relative to a set of axial 2-D contours of vertebrae, the trachea, the left main bronchi, the aorta and the heart that were segmented manually in seven reference slices. This is combined with a level set segmentation, which is initialized with the detected centerline. In [Fies 08a], contour lines that were manually drawn in axial slices are interpolated in the frequency domain without using the image itself. In [Huan 06], the user draws one contour in an axial slice, and registration based on optical flow is used to propagate the contour to neighboring slices. The segmentation error was not evaluated quantitatively.

In the last years, discriminative learning has become increasingly popular for object detection [Viol 01, Tu 05, Zhen 07]. In [Feul 09a, Feul 10b, Feul 11a], we proposed a model-based approach for esophagus segmentation which consists of multiple sub-steps:

- A region of interest (ROI) containing the esophagus is detected. In order to make sure that the esophagus is always inside, the ROI has to be made relatively large, although a larger ROI makes the detection problem harder because of the increased search space and more clutter that is visible in the ROI.

- Elliptical candidates of the esophagus contour are generated for each axial slice of the ROI using a cascade of detectors based on discriminative learning techniques. Ellipse detection is supported by an explicit model of respiratory and esophageal air, because we noticed that esophageal air holes rather distract the detector, even though they are a clear hint to a human observer.

- This is combined with prior knowledge of the esophagus shape which is modeled in two alternative ways, a Markov chain and a new variant of the Condensation algorithm. This allows to efficiently infer the most likely path through the axial slices.

- The path is converted into a surface representation and further refined using a detector that was trained to find the esophagus boundary.

We evaluated our method on 217 CT scans. Manual segmentations served as ground truth. Besides evaluating the influence of model parameters, we also investigated the inter observer variability of manual segmentations.

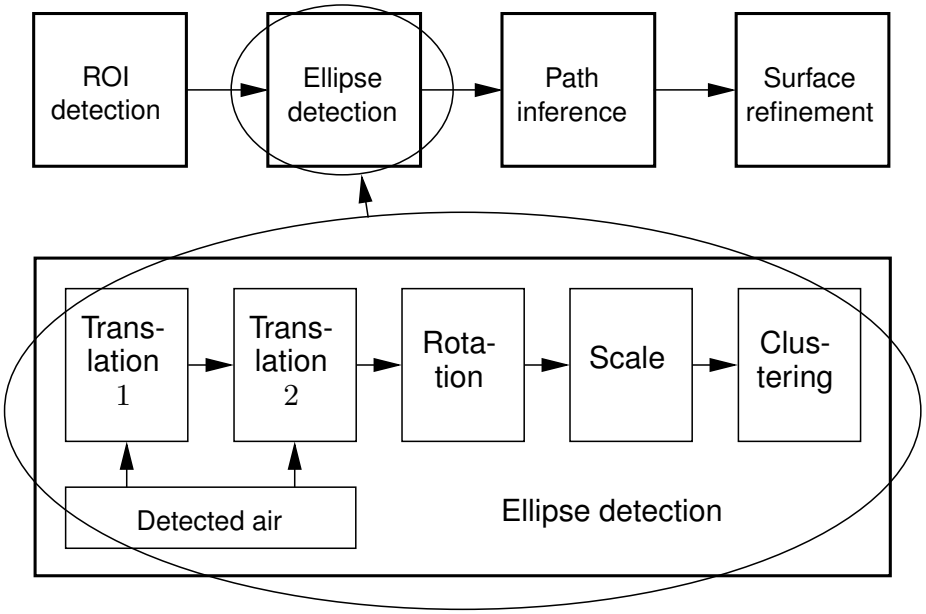


Figure 5.2: Overview of the system and the steps involved in ellipse detection.

The chapter is structured as follows: Section 5.2 describes the region of interest detection, ellipse candidate detection, path inference and surface generation steps of our method. Section 5.3 presents experiments and results, and section 5.4 concludes the chapter.

5.2 Esophagus segmentation

5.2.1 Region of interest detection

CT scans containing the thoracic esophagus typically show at least the hole thorax and often also the abdominal region. To simplify and accelerate the actual segmentation, a region of interest (ROI) is automatically detected. The ROI is an axis aligned cuboidal region that is rigidly attached to an anatomic landmark. As landmark, the bifurcation of the trachea is used because it is close to the esophagus and can be detected robustly in CT as it is unique and rich in contrast. We follow the approach of [Seif 09] to find this landmark. Instead of directly constructing a single detector, a network of detectors for salient axial slices and landmarks is used to constrain the results to be anatomically reasonable. This helps to resolve ambiguities and improves robustness. The size of the ROI and the offset from the landmark was selected such that the esophagus was always inside in all datasets that were available for evaluation with a minimum margin in x and y direction that was set to 3 cm, where the x axis points to the left and the y axis to the back. The resulting cuboid has a cross section of $13.3 \times 15.6\ \mathrm{cm}^2$. Along the z axis, which points upwards, the size is set to 26 cm. This ROI is relatively large which assures that the esophagus is not missed, but which also makes the detection problem harder as the region contains more clutter.

5.2.2 Ellipse detection

In the first steps of our method, the contour of the esophagus in an axial slice is approximated using an ellipse with parameters $\boldsymbol{e}$

$$\boldsymbol{e} = (\boldsymbol{t}, \theta, \boldsymbol{s})^T \in \mathbb{R}^5, \tag{5.1}$$

where $\boldsymbol{t} = (x, y)^T$ is the center, θ is the rotation angle within the slice, and $\boldsymbol{s} = (a, b)^T$ are the lengths of the semi major and semi minor axes ($a \geq b$). Using ellipses, we can get a good approximation of the contour with only five degrees of freedom.

For each axial slice, we want to find a set of ellipse candidates $\boldsymbol{e}^{(i)}$, $i = 1 \ldots N$, which are hypothesis of the true esophagus contour. Figure 5.2 gives an overview of the ellipse detection process. Instead of searching the five dimensional search space directly, we use a technique called marginal space learning which has been proposed in [Zhen 07]. The idea is to prune large portions of the search space using classifiers that were trained on marginal spaces, which are translation $\boldsymbol{t}$ only and translation together with orientation $(\boldsymbol{t}, \theta)^T$ in this case. The classifiers form a cascade. At each level, candidates with a poor detection score are rejected, and the remaining ones are propagated to the next level and augmented if the dimension increases. Translation is detected in steps $T1$ and $T2$ by two classifiers that differ in the examples they are trained with. They also take into account the distribution of respiratory and esophageal air, which is further explained in section 5.2.2. Finally, the set of candidates is reduced in a clustering step. As classifiers, we use probabilistic boosting trees (PBT) as explained in section 2.5

In step $T1$, we only consider translation and train a classifier to learn the probability

$$p(m = 1|\boldsymbol{H}(\boldsymbol{t})) \tag{5.2}$$

of whether there is an ellipse model instance with center $\boldsymbol{t}$ given a feature vector $\boldsymbol{H}(\boldsymbol{t})$ that was extracted at position $\boldsymbol{t}$. Here, m is the binary class label which is either one for "true" or zero for "false", and $\boldsymbol{H}$ are 3-D Haar-like features as described in section 3.2.2. Although being simple, these features are powerful because they can be computed very efficiently with the help of an integral image, which allows to search the whole volume exhaustively. Training requires a set containing positive and a set containing negative examples. The positive examples come from manual annotations, and the negative examples are generated by random sampling and rejecting samples too close to the positive samples. A set of translation candidates $C_{T1} = \{\boldsymbol{t}_1 \ldots \boldsymbol{t}_{N_{T1}}\}$ is generated from the N_{T1} positions with best detection score $p(m = 1|\boldsymbol{H}(\boldsymbol{t}))$.

In step $T2$, a second classifier is trained to also learn (5.2). In contrast to step $T1$, the negative training examples of this second classifier are not randomly drawn from the image, but set to the false positives of the first one. The second classifier only considers the set C_{T1} and generates a new set C_{T2} containing the N_{T2} top position candidates, where $N_{T1} > N_{T2}$. This significantly improves the accuracy because the second one gets specialized on the difficult cases [Feul 10b].

In the third step, a classifier is trained to learn the probability

$$p(m = 1|\boldsymbol{S}(\boldsymbol{t}, \theta)) \tag{5.3}$$

of a model instance given a vector $\boldsymbol{S}$ of steerable features (see section 3.3) that depend on rotation and translation. These are point features like the image intensity, the gradient, and

nonlinear combinations of them which are evaluated on a regular grid of size $7 \times 7 \times 3$ that is placed at position $\boldsymbol{t}$ and rotated according to θ. While these features are more expensive to compute compared to the Haar-like features, rotation detection is still efficient because only the set C_{T2} of position candidates needs to be considered. The result is a set containing N_{TR} rotation and translation candidates.

A fourth classifier is trained on the full search space to learn

$$p(m = 1|\boldsymbol{S}(\boldsymbol{t}, \theta, \boldsymbol{s})), \tag{5.4}$$

again using steerable features, but now the sampling pattern is also scaled according to $\boldsymbol{s}$. It generates the final set of ellipse candidates $C = \{\boldsymbol{e}_1 \dots \boldsymbol{e}_N\}$. Also for the last two classifiers, the negative training examples are generated from the false positives of the previous one.

Incorporating the distribution of air

In section 5.2.2, the detection only relies on local features. As described in [Feul 10b], this can lead to ambiguities in the presence of air bubbles in the esophagus. To a human, air bubbles, which are easy to see in CT, are a clear hint for the esophagus. But we observed that instead of learning a correlation between air bubbles and the esophagus, the classifier is rather distracted by esophageal air. The reason is that locally, esophageal air looks similar to respiratory air, which is a priori much more likely because the lung and the trachea cover a larger volume. A human, however, easily recognizes and excludes the respiratory organs.

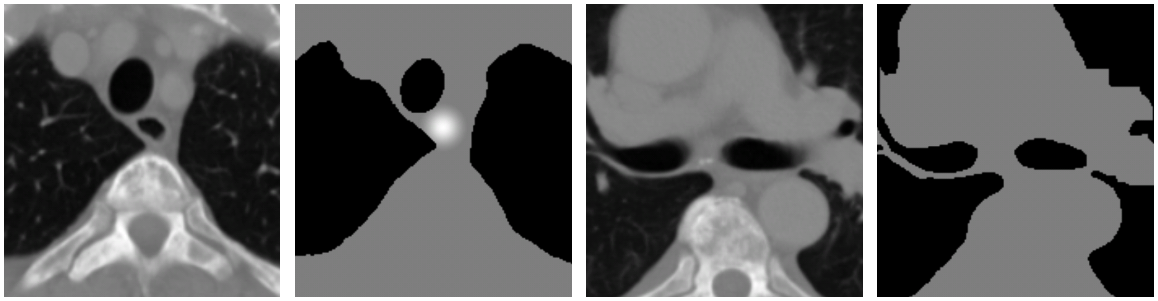
We found that the detection performance can be improved by adding the knowledge that respiratory air cannot belong to the esophagus, while air elsewhere most likely does. This is modeled by a binary mask $B(\boldsymbol{t})$

$$B(\boldsymbol{t}) = \begin{cases} 0 \ : \ \boldsymbol{t} \text{ belongs to a respiratory organ} \\ 1 \ : \ \text{otherwise} \end{cases} \tag{5.5}$$

and a probability map $A(\boldsymbol{t})$ of the esophagus that is generated from detected air holes. Respiratory air can be segmented easily in CT because it forms one connected region. In the first step, voxels with an attenuation coefficient below -625HU are labeled as air. To also include vessels and airways inside the lung, 2-D connected components in axial, sagittal and coronal slices with an area below 9 cm^2 are labeled as air as well. Now all 3-D connected components marked as air that touch the left, right, front or back border of the region of interest are removed. The removed regions are labeled as 0 in B. Elsewhere, B is set to 1. Regions filled with air that were not removed probably belong to the esophagus. These regions are marked in a second binary mask $E(\boldsymbol{t})$

$$E(\boldsymbol{t}) = \begin{cases} 1 \ : \ \text{esophageal air at } \boldsymbol{t} \\ 0 \ : \ \text{otherwise.} \end{cases} \tag{5.6}$$

A similar method to detect air holes in the esophagus is described in [Fies 08b]. If now an axial binary slice of E contains exactly one connected region marked as esophageal



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

Figure 5.3: Two examples of CT slices (a,c) along with their combined probability map $C(\boldsymbol{t})$ (b,d) generated from the distribution of air inside the volume. Left: The air hole is a clear hint for the esophagus. Right: No air hole is present, but respiratory air can be excluded.

air, this is a very strong hint for the esophagus. Then, the corresponding axial slice of the probability map A is set to

$$A(\boldsymbol{t}) = g(\|\boldsymbol{t} - \boldsymbol{p}\|_2) \tag{5.7}$$

$$g(r) = \max\left(\frac{e^{-\frac{r^2}{\sigma_a^2}} - e^{-\frac{w^2}{\sigma_a^2}}}{1 - e^{-\frac{w^2}{\sigma_a^2}}}, 0\right), \tag{5.8}$$

where $\boldsymbol{p}$ denotes the centroid of the region marked as esophageal air within the slice and g is a Gaussian with standard deviation σ_a that is deformed to have a maximum of 1 and limited support in $[-w, w]$. We selected a value of 7 mm as σ_a and 10 mm as w, which is a good compromise between accuracy (low σ_a and w) and robustness (high σ_a and w).

We now define a combined probability map $C(\boldsymbol{t})$ as

$$C(\boldsymbol{t}) = \frac{B(\boldsymbol{t}) + A(\boldsymbol{t})}{2} \tag{5.9}$$

and model the probability $p(m = 1|C(\boldsymbol{t}))$ of observing the esophagus at position $\boldsymbol{t}$ given the global distribution of air as being proportional to $C(\boldsymbol{t})$:

$$p(m = 1|C(\boldsymbol{t})) \propto C(\boldsymbol{t}). \tag{5.10}$$

During the position detection step, we are finally interested in the probability $p(m = 1|\boldsymbol{H}(\boldsymbol{t}), C(\boldsymbol{t}))$ of observing the esophagus at a certain location $\boldsymbol{t}$ given the Haar-like feature response $\boldsymbol{H}(\boldsymbol{t})$ and the information from the global distribution of air $C(\boldsymbol{t})$. In order to simplify the notation, we will omit the argument $\boldsymbol{t}$ in the remainder of this section. Using Bayes' rule, this can be rewritten as

$$p(m = 1|\boldsymbol{H}, C) = \frac{p(\boldsymbol{H}, C|m = 1)p(m = 1)}{p(\boldsymbol{H}, C)}. \tag{5.11}$$

Now we assume that the feature vector $\boldsymbol{H}$ is statistically independent from the distribution of air C, and we further assume that this independence does also hold under the condition $m = 1$. This is of course a simplifying assumption. The feature vector $\boldsymbol{H}$ is affected by the

presence of air, and therefore $\boldsymbol{H}$ and C are to some extent statistically dependent. But this dependency is not very strong because $\boldsymbol{H}$ is extracted from a local neighborhood, while C captures the global distribution of air. Locally, respiratory and esophageal air look very similar, but globally, they can be well distinguished. With this assumption, (5.11) can be transformed into

$$p(m=1|\boldsymbol{H},C) = \frac{p(\boldsymbol{H}|m=1)p(C|m=1)p(m=1)}{p(\boldsymbol{H})p(C)} \tag{5.12}$$

$$= \frac{p(m=1|\boldsymbol{H})p(m=1|C)}{p(m=1)}, \tag{5.13}$$

which is according to (5.10) proportional to the product $p(m=1|\boldsymbol{H})C$ of the classifier output and the probability map C. This means we can integrate C into a translation detector simply by multiplying it with the detection score. This is done for both detectors $T1$ and $T2$. In Figure 5.3, the probability map C is visualized for two axial CT slices.

Regions filled with respiratory air are not considered by the detector. Therefore, we also do not generate negative training examples from these regions. This makes the learning problem easier because now air is a priori more likely to be part of the esophagus.

The final detection score of a candidate $\boldsymbol{e}$ of the elliptical esophagus contour is now modeled as being proportional to the product

$$\begin{aligned} p(m=1|\boldsymbol{e}) \propto\ & p(m=1|\boldsymbol{H}(\boldsymbol{t}),C(\boldsymbol{t})) \\ & \cdot p(m=1|\boldsymbol{S}(\boldsymbol{t},\theta))p(m=1|\boldsymbol{S}(\boldsymbol{t},\theta,\boldsymbol{s})) \end{aligned} \tag{5.14}$$

of the translation, the rotation and the scale detection score. Here, we only take into account the score of the second translation detector $T2$ in order not to overemphasize translation. Detector $T1$ is only used to reject most samples at an early stage.

Clustering

In order to reduce subsequent search effort and to detect modes in the distribution of the candidates $\{\boldsymbol{e}^{(1)} \dots \boldsymbol{e}^{(N)}\}$, they are spatially clustered using an agglomerative hierarchical average-linkage clustering algorithm until a distance threshold θ_d is reached. Two clusters are merged if the mean radius of inter-cluster pairs of points is below θ_d. The result is a set of cluster centers $\{\boldsymbol{c}^{(1)} \dots \boldsymbol{c}^{(K)}\}$, $\boldsymbol{c}^{(i)} \in \mathbb{R}^2 \,\forall\, i \in \{1 \dots K\}$, with scores $\{\gamma^{(1)} \dots \gamma^{(K)}\}$, where the score $\gamma^{(k)}$ of cluster center k is the sum of detections scores $p(m=1|\boldsymbol{e})$ of its members.

5.2.3 Path inference

First order Markov model

So far, the axial slices of the volume image were treated separately. Shape knowledge is incorporated into a Markov chain model [Kind 80] of the esophagus and used to infer the most likely path through the axial slices. A factor graph [Ksch 01] of the Markov model used here is depicted in Figure 5.4. The variables $\boldsymbol{c}_1 \dots \boldsymbol{c}_T$ correspond to the axial slices of the image. Possible states of a variable $\boldsymbol{c}_t$ are the ellipses corresponding to the cluster centers $\boldsymbol{c}_t^{(k)}, k = 1 \dots K_t$ of slice t. Note that the slice index t and the 2-D ellipse center

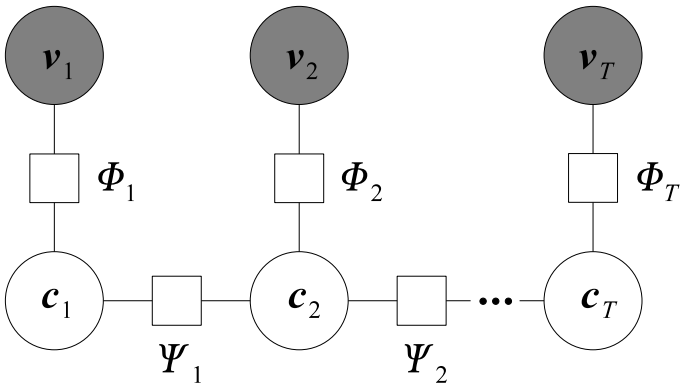


Figure 5.4: Factor graph of the Markov chain model.

$\boldsymbol{t}$ are different variables. Each state variable $\boldsymbol{c}_t$ is conditioned on the observed image slice $\boldsymbol{v}_t$. The clique potentials (or factors) of the observation cliques are denoted with Φ_t. They are set to the scores of the cluster centers:

$$\Phi_t(\boldsymbol{c}_t^{(k)}, \boldsymbol{v}_t) = \gamma_t^{(k)}. \tag{5.15}$$

The clique potentials Ψ_t between adjacent state variables $\boldsymbol{c}_t, \boldsymbol{c}_{t+1}$ represent the prior shape knowledge. They are set to the transition distribution from one slice to another:

$$\Psi_t(\boldsymbol{c}_t, \boldsymbol{c}_{t+1}) = p(\boldsymbol{c}_{t+1}|\boldsymbol{c}_t). \tag{5.16}$$

To simplify the transition distribution, it was assumed that the transition of the translation parameters $\boldsymbol{t}$ is statistically independent from the other parameters $\boldsymbol{s}$ and θ. Likewise, independence was assumed between the scale parameters $\boldsymbol{s}$ and the remaining parameters $\boldsymbol{t}$ and θ. As the rotation parameter θ is not well defined for approximately circular ellipses, the transition of rotation θ also depends on the scale $\boldsymbol{s}$, but independence was assumed between translation $\boldsymbol{t}$ and scale $\boldsymbol{s}$. With these assumptions, the transition distribution can be factorized and becomes

$$\begin{aligned} p(\boldsymbol{c}_{t+1}|\boldsymbol{c}_t) =& p(\boldsymbol{t}_{t+1}|\boldsymbol{t}_t) && (5.17)\\ & \cdot p(\theta_{t+1}|\theta_t, \boldsymbol{s}_t) && (5.18)\\ & \cdot p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t). && (5.19) \end{aligned}$$

The translation transition (5.17) is modeled as a 2-D normal distribution

$$p(\boldsymbol{t}_{t+1}|\boldsymbol{t}_t) = \mathcal{N}(\Delta\boldsymbol{t}_t|\boldsymbol{\Sigma}_p, \boldsymbol{m}_p) \tag{5.20}$$

with $\Delta\boldsymbol{t}_t = \boldsymbol{t}_{t+1} - \boldsymbol{t}_t$, and the transition (5.19) of scale $\boldsymbol{s} = (a, b)$ as a 4-D normal distribution

$$p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t) = \mathcal{N}\left((\boldsymbol{s}_{t+1}, \boldsymbol{s}_t)^T|\boldsymbol{\Sigma}_s, \boldsymbol{m}_s\right). \tag{5.21}$$

In (5.20) and (5.21), $\boldsymbol{\Sigma}_p \in \mathbb{R}^{2\times2}$ and $\boldsymbol{\Sigma}_s \in \mathbb{R}^{4\times4}$ are the covariance matrices, and $\boldsymbol{m}_p \in \mathbb{R}^2$ and $\boldsymbol{m}_s \in \mathbb{R}^4$ are the mean vectors of the two normal distributions. We chose to model $p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t)$ as a 4-D normal distribution and not as a 2-D normal distribution of the difference $\Delta\boldsymbol{s}_t = \boldsymbol{s}_{t+1} - \boldsymbol{s}_t$, because some constellations of $\boldsymbol{s} = (a, b)^T$ such as $a \gg b$ are unlikely, which cannot be captured by the 2-D distribution.

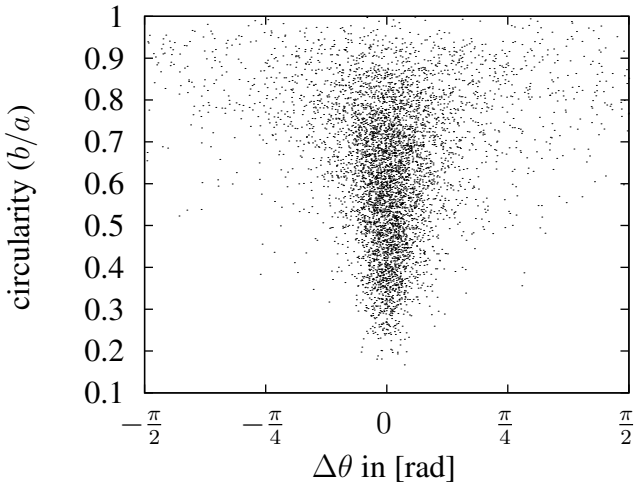


Figure 5.5: Samples of rotation transitions from one axial slice to another of the ellipses fitted to the ground truth annotations. For a fixed circularity value $\frac{b}{a}$, the 1-D conditional probability density $p(\Delta\theta|\frac{b}{a})$ can be well approximated with a normal distribution. The standard deviation is higher for high circularity values. For $\frac{b}{a} = 1$, the distribution is uniform in $[-\frac{\pi}{2}, \frac{\pi}{2})$.

The variance of the rotation transition is small for an elongated ellipse because the esophagus is usually smooth and not heavily twisted. However, the variance highly increases for more circular ellipses. The reason is that θ takes arbitrary values for a circle. This may results in big jumps of θ from slice to slice even though the shape of the esophagus contour hardly changes.

This is handled by modeling (5.18) with different normal distributions for elongated and more circular ellipses. In total, we use ten 1-D normal distributions of $\Delta\theta_t = \theta_{t+1} - \theta_t$, one for a certain interval of circularity, which is measured by the ratio $\frac{b}{a}$ of the length of the semi minor and the semi major axis:

$$p(\theta_{t+1}|\theta_t, a_t, b_t) \approx \mathcal{N}\left(\Delta\theta_t \middle| \sigma_r\left(\frac{b_t}{a_t}\right), m_r\left(\frac{b_t}{a_t}\right)\right). \tag{5.22}$$

Here, $\sigma_r(\frac{b}{a})$ is the standard deviation and $m_r(\frac{b}{a})$ the mean of the normal distribution that corresponds to the circularity value $\frac{b}{a}$. Figure 5.5 shows samples of rotation transitions along with the circularity. It illustrates that (5.18) can be represented with a Gaussian for a fixed circularity value.

This is an approximation because a normal distribution has only one mode and unlimited support, but a rotation by $180°$ results into the same ellipse. It is, however, enough to only consider the range of $\Delta\theta$ between $-90°$ and $90°$, where the approximation works well.

The parameters of all normal distributions were estimated from manually annotated data. The annotations are contours of the esophagus drawn in each axial slice. For each slice, an ellipse is fitted into the contour points. We use the method described in [Fitz 99] to non-iteratively compute the least squares solution. The transitions from one slice to the next are treated as samples and used to compute the mean vectors and covariance matrices.

The a posteriori joint distribution of all states $p(\boldsymbol{c}_{1:T}|\boldsymbol{v}_{1:T})$ is now given by the product of all factors of the factor graph. The maximum a posteriori (MAP) estimate

$$\begin{aligned}\hat{\boldsymbol{c}}_{1:T}^{(\text{MAP})} &= \underset{\boldsymbol{c}_{1:T}}{\operatorname{argmax}}\ p(\boldsymbol{c}_{1:T}|\boldsymbol{v}_{1:T}) \\ &= \underset{\boldsymbol{c}_{1:T}}{\operatorname{argmax}}\ \Phi_1(\boldsymbol{c}_1, \boldsymbol{v}_1) \prod_{t=2}^{T} \Phi_t(\boldsymbol{c}_t, \boldsymbol{v}_t)\Psi_{t-1}(\boldsymbol{c}_{t-1}, \boldsymbol{c}_t) \end{aligned} \tag{5.23}$$

can be computed efficiently using dynamic programming.

Second order Markov model

Alternatively, the shape prior was modeled with a Markov chain that assumes a Markov order of two for the transition of translation. In general, a higher Markov order makes a model more accurate, but also requires far more examples during training because higher dimensional probability density functions need to be estimated. The transition of rotation and scale was handled as described in section 5.2.3 because we observed that the translation parameters are more continuous compared to rotation and scale, and therefore the shape prior should benefit most from a second order assumption here. Furthermore, generalizing (5.21) and (5.22) to a second order model leads to problems because of limited training data. Now, the factor Ψ_t corresponding to the state transition probability (5.16) becomes

$$\begin{aligned}\Psi_t(\boldsymbol{c}_{t-1}, \boldsymbol{c}_t, \boldsymbol{c}_{t+1}) =& p(\boldsymbol{c}_{t+1}|\boldsymbol{c}_t, \boldsymbol{c}_{t-1}) && (5.24)\\ =& p(\boldsymbol{t}_{t+1}|\boldsymbol{t}_t, \boldsymbol{t}_{t-1}) && (5.25)\\ \cdot& p(\theta_{t+1}|\theta_t, \boldsymbol{s}_t) && (5.26)\\ \cdot& p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t). && (5.27)\end{aligned}$$

The translation transition is again modeled using a normal distribution, but now, also the second derivative

$$\Delta\Delta\boldsymbol{t}_t \in \mathbb{R}^2 \ \text{ with } \ \Delta\Delta\boldsymbol{t}_t = \Delta\boldsymbol{t}_t - \Delta\boldsymbol{t}_{t-1} \tag{5.28}$$

of $\boldsymbol{t}$ with respect to z is considered, which corresponds to the curvature of the esophagus:

$$p(\boldsymbol{t}_{t+1}|\boldsymbol{t}_t, \boldsymbol{t}_{t-1}) = \mathcal{N}\left((\Delta\boldsymbol{t}_t, \Delta\Delta\boldsymbol{t}_t)^T|\boldsymbol{\Sigma}_{p_2}, \boldsymbol{m}_{p_2}\right). \tag{5.29}$$

The MAP estimate

$$\begin{aligned}\hat{\boldsymbol{c}}_{1:T}^{(\text{MAP})} = \underset{\boldsymbol{c}_{1:T}}{\operatorname{argmax}} \Bigg(& \Phi_1(\boldsymbol{c}_1, \boldsymbol{v}_1)\Phi_2(\boldsymbol{c}_2, \boldsymbol{v}_2)\Psi_1(\boldsymbol{c}_1, \boldsymbol{c}_2) && (5.30)\\ & \cdot \prod_{t=3}^{T} \Phi_t(\boldsymbol{c}_t, \boldsymbol{v}_t)\Psi_{t-1}(\boldsymbol{c}_{t-2}, \boldsymbol{c}_{t-1}, \boldsymbol{c}_t) \Bigg) && (5.31)\end{aligned}$$

can be computed in the same way as in the single order case.

Particle filtering

We furthermore investigated to model and infer the esophagus path with a particle filter approach [Isar 98b] instead of using a Markov chain. Particle filtering, which is also known

as Condensation, is popular for tracking applications. Probability distributions are represented non-parametrically with weighted samples which are called particles. In contrast to Kalman or extended Kalman filtering, the distributions may be multimodal, which allows to model different hypothesis of the true state. It is also becoming popular for tracking of tubular structures [Flor 05, Scha 07].

We formulate the problem of inferring the esophagus shape in the framework of dynamic state estimation. Though the problem is not dynamic, we treat the vertical axis z of the volume image as time t. As before, an axial slice becomes the observation $\boldsymbol{v}_t$. The unknown state at time t is the ellipse parameter vector $\boldsymbol{e}_t$. Given the observation density $p(\boldsymbol{v}_t|\boldsymbol{e}_t)$ and the state transition density $p(\boldsymbol{e}_{t+1}|\boldsymbol{e}_t)$, the probability density $p(\boldsymbol{e}_{t+1}|\boldsymbol{v}_{1:t+1})$ of the state at time $t+1$ given all previous observations can be computed recursively as

$$p(\boldsymbol{e}_{t+1}|\boldsymbol{v}_{1:t+1}) = \frac{p(\boldsymbol{v}_{t+1}|\boldsymbol{e}_{t+1}) \int p(\boldsymbol{e}_{t+1}|\boldsymbol{e}_t) p(\boldsymbol{e}_t|\boldsymbol{v}_{1:t}) d\boldsymbol{e}_t}{p(\boldsymbol{v}_{t+1}|\boldsymbol{v}_{1:t})} \tag{5.32}$$

for a Markov order of one. Here, we use the notation $\boldsymbol{v}_{1:t}$ for the sequence $\boldsymbol{v}_1 \dots \boldsymbol{v}_t$. This is the core equation of probabilistic dynamic state estimation. In the Condensation algorithm [Isar 98a], the integral of (5.32) is computed by drawing samples from the probability density $p(\boldsymbol{e}_t|\boldsymbol{v}_{1:t})$ that is represented by a set $\mathcal{S}_t$ of particles

$$\mathcal{S}_t = \left\{ \left(\boldsymbol{e}_t^{(i)}, P_t^{(i)} \right), i = 1 \dots I \right\}. \tag{5.33}$$

Each particle consists of a sample $\boldsymbol{e}_t^{(i)}$ and a weight $P_t^{(i)}$. The probability density $p(\boldsymbol{e}_t|\boldsymbol{v}_{1:t})$ is approximated as

$$p_{\mathcal{S}_t}(\boldsymbol{e}_t|\boldsymbol{v}_{1:t}) = \sum_{i=1}^{I} P_t^{(i)} \delta(\boldsymbol{e}_t - \boldsymbol{e}_t^{(i)}), \tag{5.34}$$

where δ is a window function. For particle filtering, it is common practice to simply use the Dirac window δ because the empiric probability density $p_{\mathcal{S}_t}$ is not evaluated but only used for drawing samples or computing moments.

In the Condensation algorithm [Isar 98a], the samples are noisily propagated to the next time step using $p(\boldsymbol{e}_{t+1}|\boldsymbol{e}_t)$ and then weighted according to whether they fit to the new observation $\boldsymbol{v}_{t+1}$ using $p(\boldsymbol{v}_{t+1}|\boldsymbol{e}_{t+1})$. Figure 5.6 depicts this algorithm.

We found that the idea of marginal space learning can be nicely integrated into the particle filtering framework by factorizing both the observation density $p(\boldsymbol{v}_t|\boldsymbol{e}_t)$ and the state transition density $p(\boldsymbol{e}_{t+1}|\boldsymbol{e}_t)$, and expressing the factors of the observation density using classifiers that were trained on marginal spaces.

The observation density $p(\boldsymbol{v}|\boldsymbol{e})$ (to improve the readability, the subscript t is omitted in this section if it is the same for all variables) requires a generative model which is often not available in the context of object detection in images. Using Bayes' rule

$$p(\boldsymbol{v}|\boldsymbol{e}) = \frac{p(\boldsymbol{e}|\boldsymbol{v}) p(\boldsymbol{v})}{p(\boldsymbol{e})}, \tag{5.35}$$

it can be transformed to a discriminative model $p(\boldsymbol{e}|\boldsymbol{v})$ and two priors for the image $\boldsymbol{v}$ and the parameter vector $\boldsymbol{e}$. By using Bayes' rule once again, $p(\boldsymbol{e}|\boldsymbol{v}) = p(\boldsymbol{t}, \theta, \boldsymbol{s}|\boldsymbol{v})$ can be factorized into

$$p(\boldsymbol{e}|\boldsymbol{v}) = p(\boldsymbol{t}|\boldsymbol{v}) p(\theta|\boldsymbol{t}, \boldsymbol{v}) p(\boldsymbol{s}|\theta, \boldsymbol{t}, \boldsymbol{v}). \tag{5.36}$$

For the state transition density $p(\boldsymbol{e}_{t+1}|\boldsymbol{e}_t)$, we use the factorization of (5.17). Together with (5.36), (5.32) can be rewritten as

$$\begin{aligned} p(\boldsymbol{e}_{t+1}|\boldsymbol{v}_{1:t+1}) \propto \\ \frac{1}{p(\boldsymbol{e}_{t+1})} p(\boldsymbol{s}_{t+1}|\theta_{t+1}, \boldsymbol{t}_{t+1}, \boldsymbol{v}_{t+1}) \int_{\boldsymbol{s}_t} p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t) \\ p(\theta_{t+1}|\boldsymbol{t}_{t+1}, \boldsymbol{v}_{t+1}) \left[\int_{\theta_t} p(\theta_{t+1}|\theta_t, \boldsymbol{s}_t) \right. \\ p(\boldsymbol{t}_{t+1}|\boldsymbol{v}_{t+1}) \left(\int_{\boldsymbol{t}_t} p(\boldsymbol{t}_{t+1}|\boldsymbol{t}_t) p(\boldsymbol{e}_t|\boldsymbol{v}_{1:t}) \right. \\ \left. \left. \left. d\boldsymbol{t}_t \right) d\theta_t \right] d\boldsymbol{s}_t. \end{aligned} \tag{5.37}$$

Because $p(\boldsymbol{v})$ and $p(\boldsymbol{v}_{t+1}|\boldsymbol{v}_{1:t})$ do not depend on the parameters $\boldsymbol{e}$ that are to be estimated, they can be treated like constants. In (5.37), the integral of (5.32) over the state space was replaced by three nested integral over the scale, rotation and the translation subspace. Note that each weighted integral is very similar to the weighted integral in (5.32). Like in one iteration of the Condensation algorithm, it can be carried out by drawing, propagating and weighting samples. But instead of propagating the samples across time, they are propagated across dimension of the state space within a single time step. This is especially useful for higher dimensional state spaces, like 5-D in this case: Filling a 5-D state space with particles would require a high number of particles. Here, we can reject particles already after the first 2-D integration if the translation parameters do not fit the new observation and concentrate on particles with promising translation parameters. The same can be done for the rotation parameters

Figure 5.7 depicts the algorithm we used to compute (5.37). It is formulated to solve our specific problem, but the principle is always applicable if the state transition probability density (5.17) and the observation probability density (5.35) can be factorized and the factors are available.

The factors of (5.36) are now modeled as being proportional to the scores of the three detectors trained on the marginal spaces of translation, rotation, and scale

$$p(\boldsymbol{t}|\boldsymbol{v}) \propto p(m = 1|\boldsymbol{H}(\boldsymbol{t}), C(\boldsymbol{t})) \tag{5.38}$$

$$p(\theta|\boldsymbol{t}, \boldsymbol{v}) \propto p(m = 1|\boldsymbol{S}(\boldsymbol{t}, \theta)) \tag{5.39}$$

$$p(\boldsymbol{s}|\theta, \boldsymbol{t}, \boldsymbol{v}) \propto p(m = 1|\boldsymbol{S}(\boldsymbol{t}, \theta, \boldsymbol{s})), \tag{5.40}$$

and the prior $p(\boldsymbol{e})$ is assumed to be uniform. In contrast to the Markov chain approach, only a single translation detector is used here. A second one could be integrated by adding a fourth integral in equation (5.37).

Finally, we are interested in the MAP estimate

$$\hat{\boldsymbol{e}}_{1:T}^{(\text{MAP})} = \underset{\boldsymbol{e}_{1:T}}{\operatorname{argmax}}\ p(\boldsymbol{e}_{1:T}|\boldsymbol{v}_{1:T}). \tag{5.41}$$

This estimate can be easily obtained [Isar 98b] from $p(\boldsymbol{e}_T|\boldsymbol{v}_{1:T})$ by finding the history $\boldsymbol{e}_{1:T}$ of the particle

$$\hat{\boldsymbol{e}}_T^{(\text{MAP})} = \underset{\boldsymbol{e}_T}{\operatorname{argmax}}\ p(\boldsymbol{e}_T|\boldsymbol{v}_{1:T}) \tag{5.42}$$

1: **Input**: Old particle set

$$\mathcal{S}_t = \left\{ \left(\boldsymbol{e}_t^{(i)}, P_t^{(i)} \right), i = 1 \dots I \right\}$$

of time step t

2: Construct the new particle set

$$\mathcal{S}_{t+1} = \left\{ \left(\boldsymbol{e}_{t+1}^{(i)}, P_{t+1}^{(i)} \right), i = 1 \dots I \right\}$$

of the next time step from $\mathcal{S}_t$ as follows:

3: **for** i=1...I **do**

4: Construct the particle $\left(\boldsymbol{e}_{t+1}^{(i)}, P_{t+1}^{(i)} \right)$:

5: **Select** a sample $\boldsymbol{e}_t^{(j)}$ with probability $P_t^{(j)}$

6: **Predict** by sampling from $p(\boldsymbol{e}_{t+1}|\boldsymbol{e}_t^{(j)})$ to choose the sample $\boldsymbol{e}_{t+1}^{(i)}$ of the next time step

7: **Weight** the new sample by setting

$$P_{t+1}^{(i)} = p(\boldsymbol{v}_{t+1}|\boldsymbol{e}_{t+1}^{(i)})$$

8: **end for**

9: Normalize the weights s. th. $\sum_{i=1}^{I} P_{t+1}^{(i)} = 1$

10: **Output**: New particle set $\mathcal{S}_{t+1}$

Figure 5.6: One iteration of the original Condensation algorithm [Isar 98a].

Input: Old particle set

$$\mathcal{S}_t = \left\{ \left(\boldsymbol{e}_t^{(i)} = (\boldsymbol{t}_t^{(i)}, \theta_t^{(i)}, \boldsymbol{s}_t^{(i)}), P_t^{(i)} \right), i = 1 \ldots I \right\}$$

of time step t

STEP 1: Construct a first intermediate set of sub-particles

$$\mathcal{S}_{t+1,1} = \left\{ \left(\boldsymbol{t}_{t+1}^{(i)}, P_{t+1,1}^{(i)} \right), i = 1 \ldots I \right\}$$

from the old particle set $\mathcal{S}_t$:
for i=1...I **do**
  Construct the sub-particle $\left(\boldsymbol{t}_{t+1}^{(i)}, P_{t+1,1}^{(i)} \right)$:
  Select a sample $\boldsymbol{e}_t^{(j)}$ with probability $P_t^{(j)}$
  Predict by sampling from $p(\boldsymbol{t}_{t+1}|\boldsymbol{e}_t^{(j)}) = p(\boldsymbol{t}_{t+1}|\boldsymbol{t}_t^{(j)})$ to choose $\boldsymbol{t}_{t+1}^{(i)}$
  Weight the new sample by setting $P_{t+1,1}^{(i)} = p(\boldsymbol{t}_{t+1}^{(i)}|\boldsymbol{v}_{t+1})$
end for
Normalize the weights s. th. $\sum_{i=1}^{I} P_{t+1,1}^{(i)} = 1$

STEP 2: Construct a second intermediate set of sub-particles

$$\mathcal{S}_{t+1,2} = \left\{ \left((\boldsymbol{t}_{t+1}^{(i)}, \theta_{t+1}^{(i)}), P_{t+1,2}^{(i)} \right), i = 1 \ldots I \right\}$$

from the first intermediate set $\mathcal{S}_{t+1,1}$:
for i=1...I **do**
  Construct the sub-particle $\left((\boldsymbol{t}_{t+1}^{(i)}, \theta_{t+1}^{(i)}), P_{t+1,2}^{(i)} \right)$:
  Select a sample $\boldsymbol{t}_{t+1}^{(j)}$ with probability $P_{t+1,1}^{(j)}$
  Predict by first finding the full particle predecessor $\boldsymbol{e}_t^{(k)}$ of sub-particle $\boldsymbol{t}_{t+1}^{(j)}$ and then sampling from $p(\theta_{t+1}|\boldsymbol{e}_t^{(k)}) = p(\theta_{t+1}|\theta_t^{(k)}, \boldsymbol{s}_t^{(k)})$ to find $\theta_{t+1}^{(i)}$
  Weight with $P_{t+1,2}^{(i)} = p(\theta_{t+1}^{(i)}|\boldsymbol{t}_{t+1}^{(i)}, \boldsymbol{v}_{t+1})$
end for
Normalize the weights s. th. $\sum_{i=1}^{I} P_{t+1,2}^{(i)} = 1$

Figure 5.7: One iteration of the proposed variant of the Condensation algorithm for the problem of ellipse tracking. This variant was first published in [Feul 11a]. **Continued in Figure 5.8**.

1: **STEP 3:** Construct the particle set of the next time step

$$\mathcal{S}_{t+1} = \left\{ \left(\boldsymbol{e}_{t+1}^{(i)}, P_{t+1}^{(i)} \right), i = 1 \dots I \right\}$$

from the second intermediate set $\mathcal{S}_{t+1,2}$:
2: **for** i=1…I **do**
3: Construct the particle $\left(\boldsymbol{e}_{t+1}^{(i)}, P_{t+1}^{(i)} \right)$:
4: **Select** a sample $(\boldsymbol{t}_{t+1}^{(j)}, \theta_{t+1}^{(j)})$ with probability $P_{t+1,2}^{(j)}$
5: **Predict** by first finding the full particle predecessor $\boldsymbol{e}_t^{(k)}$ of sub-particle $(\boldsymbol{t}_{t+1}^{(j)}, \theta_{t+1}^{(j)})$ and then sampling from $p(\boldsymbol{s}_{t+1}|\boldsymbol{e}_t^{(k)}) = p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t^{(k)})$ to find $\boldsymbol{s}_{t+1}^{(i)}$
6: **Weight** with $P_{t+1}^{(i)} = p(\boldsymbol{s}_{t+1}^{(i)}|\theta_{t+1}^{(i)}, \boldsymbol{t}_{t+1}^{(i)}, \boldsymbol{v}_{t+1})$
7: **end for**
8: Normalize the weights s. th. $\sum_{i=1}^{I} P_{t+1}^{(i)} = 1$
9: **Output**: New particle set $\mathcal{S}_{t+1}$.

Figure 5.8: One iteration of the proposed variant of the Condensation algorithm for the problem of ellipse tracking. This variant was first published in [Feul 11a]. **Continued from Figure 5.7**.

with the highest weight in the last time step T.

5.2.4 Surface generation

After the MAP estimate of the path has been detected, the sequence of ellipses is converted into a triangular mesh representation by sampling the ellipses and connecting neighboring point sets with a triangle strip.

The cross-section of the esophagus is generally not elliptic, and the path obtained in section 5.2.3 often contains some inaccuracies. Therefore, the mesh model is further refined to better fit the surface of the organ.

A PBT classifier was trained to learn the boundary of the esophagus. The classifier uses steerable features as proposed in [Zhen 07]. As for ellipse detection, the steerable features are sampled on a regular grid, but now with a size of $5 \times 5 \times 9$. For each mesh vertex, the sampling pattern is placed so that the vertex is in the center of the pattern and the longest axis points in direction of the mesh normal. Now the pattern is moved along the normal to find the maximal detector response and the new position of the vertex. Finally, the surface is passed through a Laplacian smoothing filter that replaces each vertex with the average of its neighbors. Smoothing is necessary because the vertices are displaced independently from each other. This process of deformation and smoothing is repeated for a certain number of iterations that is varied in the experiments.

Figure 5.9 shows the first iteration of boundary refinement for a section of an inferred path.

Figure 5.10 summarizes the detection pipeline and shows example output for each step.

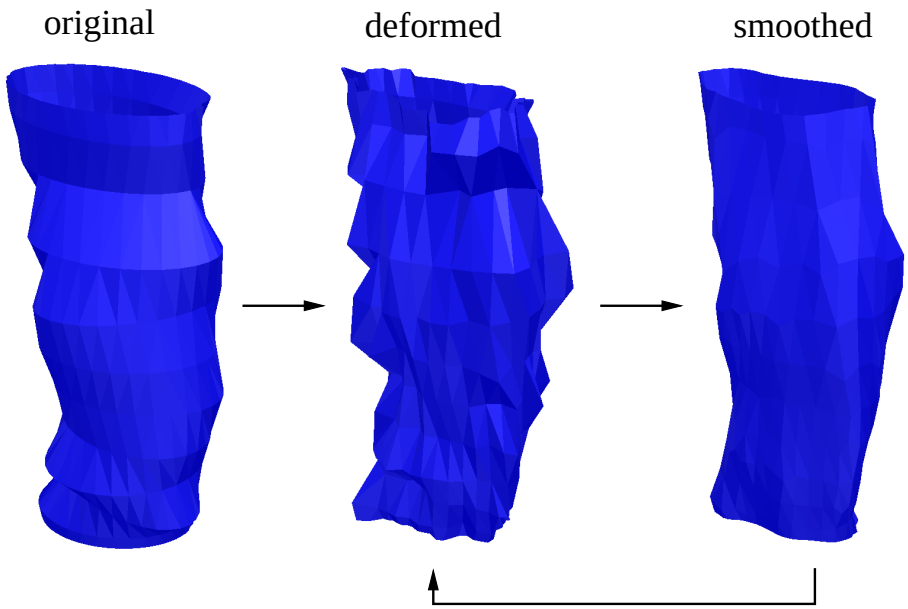


Figure 5.9: Example of one boundary refinement iteration shown for a section of the esophagus. Left: The ellipses obtained in the path inference step, connected with triangle strips. Middle: Surface after displacing the vertices along their normals according to the classifier output. Right: Smoothed surface.

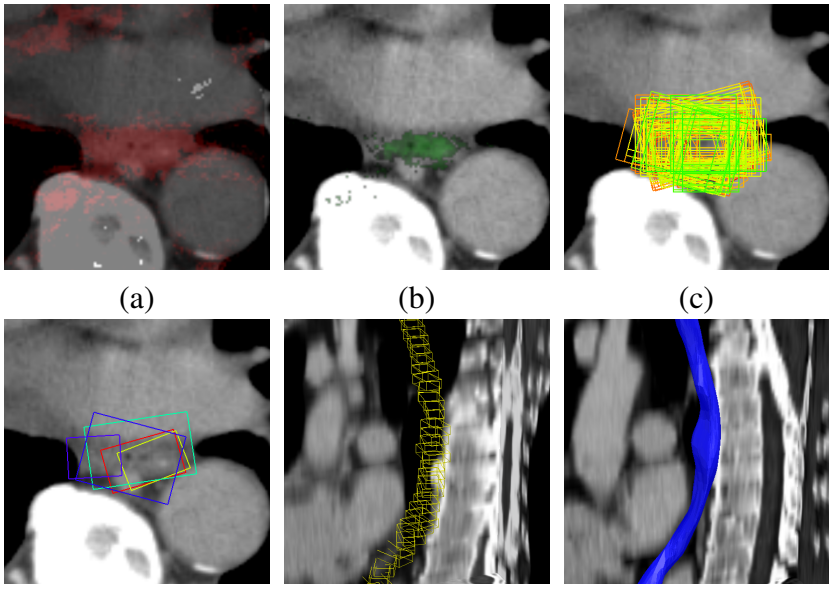


Figure 5.10: Example output for each step of the detection pipeline. (a): Score $p(m = 1|\boldsymbol{H})C$ of the first classifier for position multiplied with the probability map C. (b): Score $p(m = 1|\boldsymbol{H})C$ of the second classifier for position multiplied with the probability map C, evaluated for the best candidates from stage (a). (c): Candidate boxes after rotation and scale detection. The confidence of a box is color coded in HSV color space. Violet is lowest, red is highest score. (d): Cluster centers after clustering and merging. (e): Result of the path inference step. (f): Final surface after refinement.

Scanner type	Number of datasets
Biograph 64	1
Definition AS+	1
Sensation 10	25
Sensation 16	6
Sensation 64	110
Volume Zoom	1

Table 5.1: CT scanners used for data acquisition.

Filter kernel	kernel description	Number of datasets
B30f	medium smooth	2
B31f	medium smooth +	24
B31s	medium smooth +	1
B40f	medium	6
B41f	medium +	110
B70f	very sharp	1

Table 5.2: Filter kernels used for image reconstruction.

5.3 Results

5.3.1 Results of cross-validation

The method has been evaluated on 144 CT scans of the thoracic or the thoracic and abdominal region. No patient was included twice. The voxel spacing in x and y direction was in the range of 0.7 mm to 0.8 mm. The spacing in (longitudinal) z direction was 5 mm. After ROI detection, the volumes were resampled to a voxel spacing of $0.7 \times 0.7 \times 5\mathrm{mm}^3$. The data was acquired using six different CT scanner types listed in Table 5.1. Out of the 144 datasets, 143 were reconstructed using filter kernels for soft tissue, and one was reconstructed using a lung kernel (B70f). The filter kernels are listed in Table 5.2. Further details on the kernels can be found in [Soma 06]. The accelerating voltage was 120kV in all cases, and the tube current ranged from 94mA to 575mA with a mean and standard deviation of 293.79 ± 87.25 mA.

Manual segmentations were available for all datasets. The data was segmented by non-experts, but difficult cases were reviewed by a radiologist. The segmentations typically ranged from the thyroid gland down to a level below the left atrium.

Among the scans, 34 were taken from patients suffering from lymphoma, which often causes enlarged lymph nodes in the mediastinal region. In some datasets, the esophagus contained remains of orally given contrast agent.

The accuracy was measured using threefold cross-validation. For each fold, all five classifiers for translation ($2\times$), orientation, scale and surface were trained on two thirds of the data, and the parameters of the Markov model were estimated from the same two thirds of the data. The remaining third of the data was used for testing. There was no overlap between training and testing data. For evaluation, the detector was run in z direction on the same interval covered by the manual annotation in order not to introduce artificial errors because of different lengths of the segmentations.

Classifier	tree levels	weak classifiers	candidates
Translation 1	2	20	400
Translation 2	2	20	120
Rotation	2	20	50
Scale	2	20	200
Surface	5	20	n/a

Table 5.3: Parameter settings for the five classifiers of the detection pipeline: The number of levels in the PBT classifier, the number of weak classifiers per AdaBoost node, and the number of candidates generated.

Method		mean err. in mm	Hausdorff dist. in mm
P	Proposed method	1.80 ± 1.17	12.62 ± 7.01
PB	Proposed method, best 80%	1.34 ± 0.31	9.65 ± 3.07
NS	No surface refinement	2.24 ± 1.08	12.93 ± 7.16
B	Only binary air model $B(\boldsymbol{t})$	1.88 ± 1.24	13.00 ± 7.88
NA	No air model	1.94 ± 1.39	13.06 ± 7.21
ST	Single translation class.	2.07 ± 1.47	14.50 ± 8.92
NAT	No air mdl., single trnsl. cls.	2.32 ± 1.87	15.02 ± 9.83
M0	Markov order 0	2.30 ± 1.49	17.29 ± 11.42
M2	Markov order 2	1.80 ± 1.15	12.65 ± 6.92
PF	Particle filtering	5.39 ± 3.08	22.32 ± 7.97
IOV	Inter observer variability	0.78 ± 0.17	7.29 ± 2.22

Table 5.4: Results of performance evaluation. Shown is the mean error and the mean Hausdorff distance along with the corresponding standard deviations. First row: The proposed method uses the first order Markov chain approach. Rows 2-10: Proposed method with at least one parameter or experimental setting altered. Last row: Inter observer variability measured on ten of the 144 datasets.

ROI detection succeeded in all of the 144 datasets, meaning that the bifurcation of the trachea was always detected with a reasonable accuracy. Due to the large ROI, the segmentation method can tolerate normal anatomical variations and detection errors.

Unless otherwise stated, parameters of the classifiers were set to the values displayed in Table 5.3, the distance threshold θ_d was set to 8 mm, and surface refinement was iterated two times.

Table 5.4 and Figure 5.11 show the results of performance evaluation. Two error measures were computed: The mean symmetric point-to-mesh distance, and the maximum symmetric point-to-mesh distance, which is also known as Hausdorff distance. Symmetric means that the distance between two meshes remains the same if the meshes are swapped.

Our proposed method (P), which uses the Markov chain model for path inference with a Markov order of one, segments the esophagus with a mean error of 1.80 mm and a standard deviation of 1.17 mm. The data used for evaluation also contains difficult and extreme

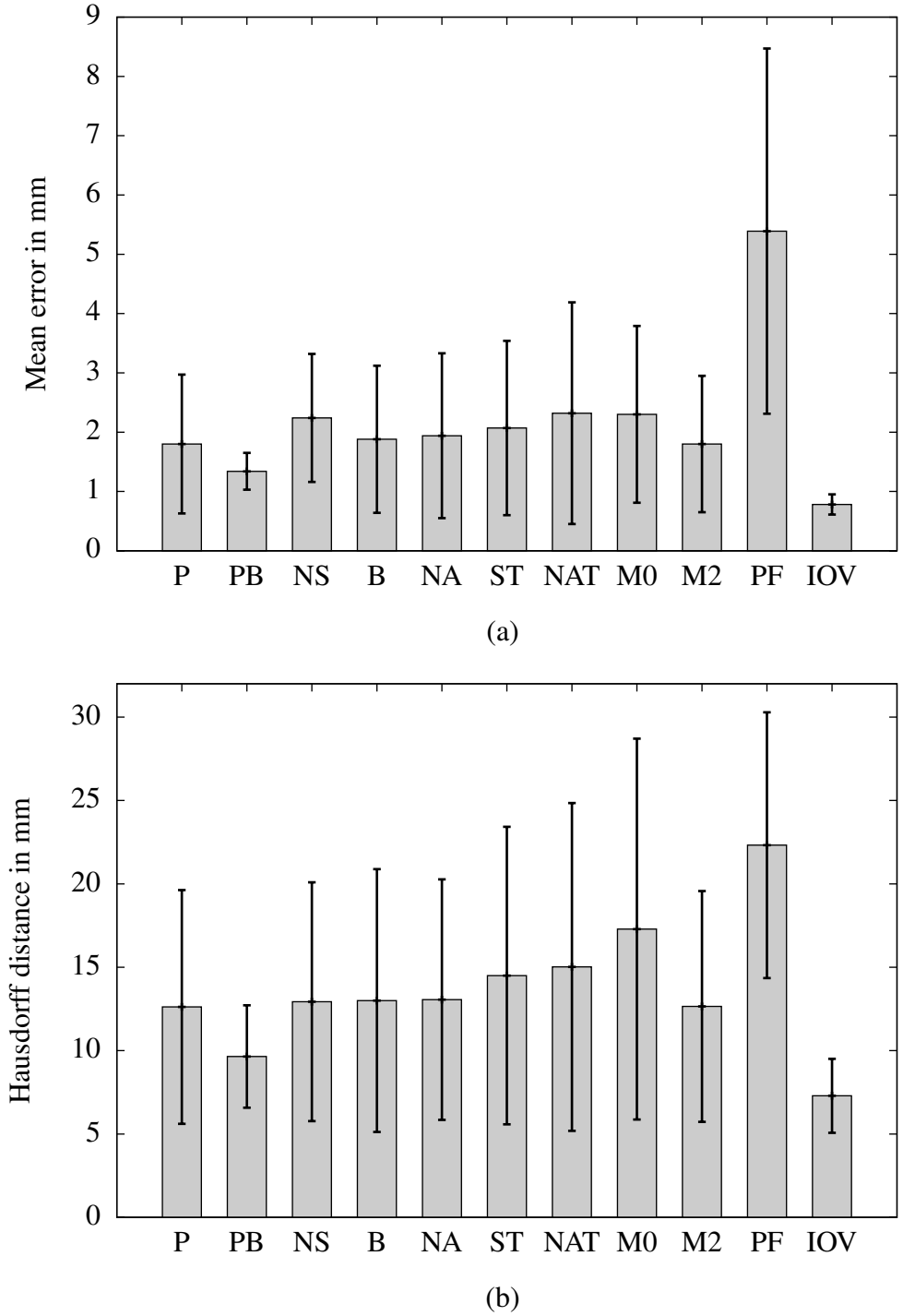
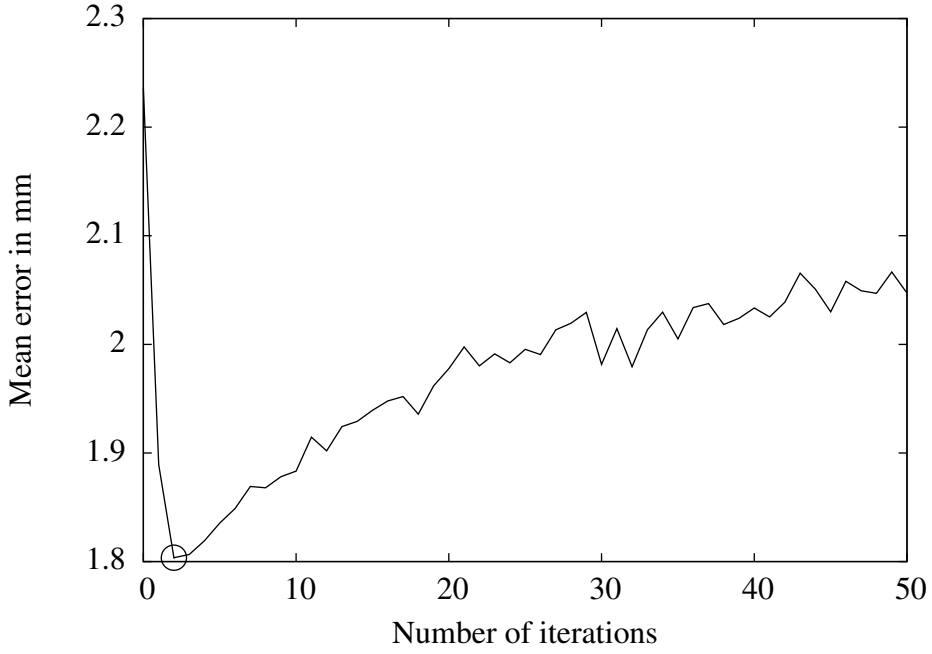
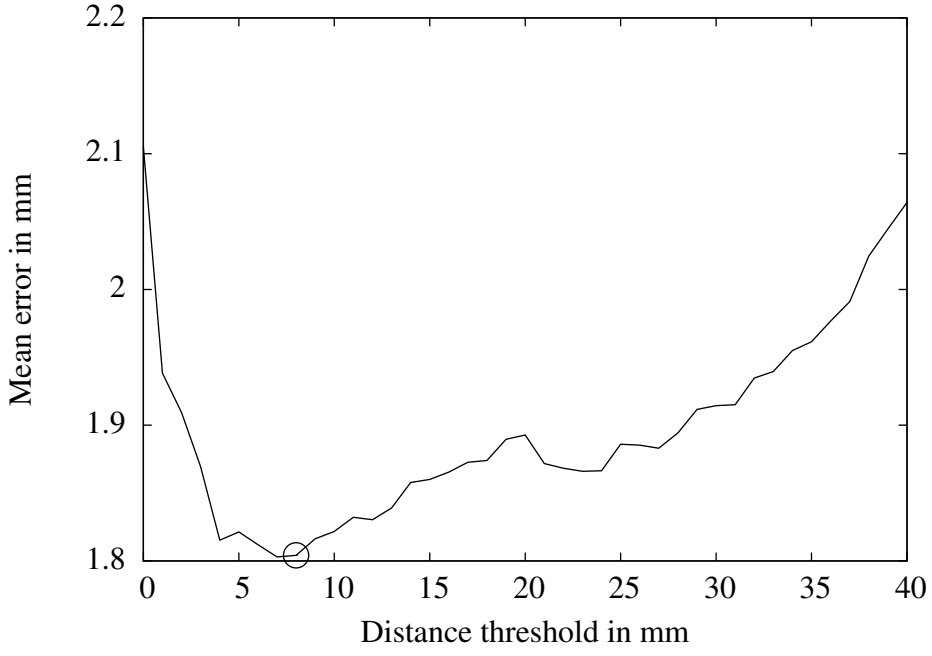


Figure 5.11: Mean segmentation error (a) and Hausdorff distance (b) of our segmentation method and different variants. The length of an error bar is two standard deviations. See text and Table 5.4 for an explanation of the abbreviations.

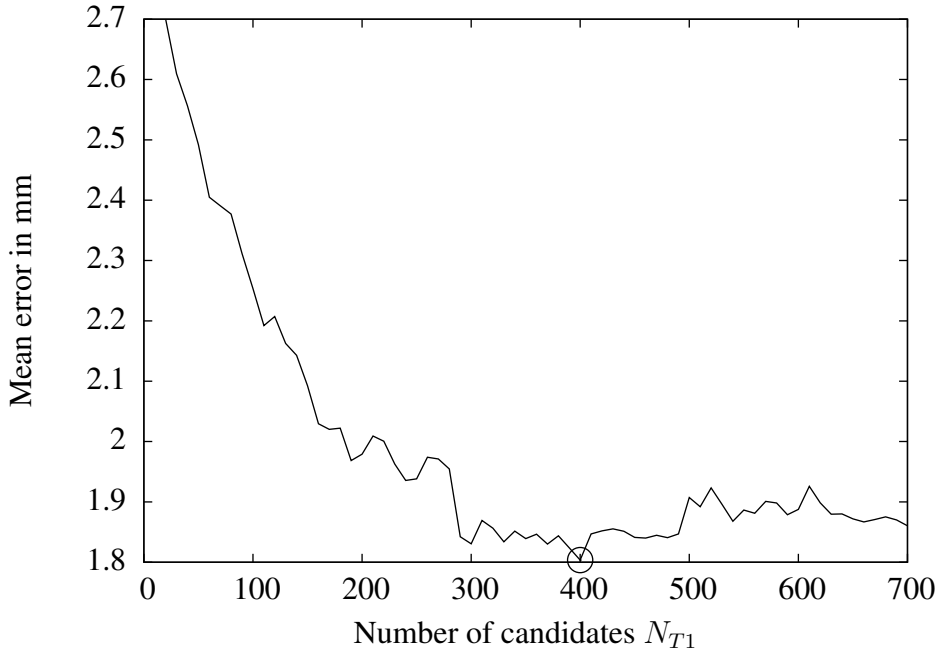


(a)

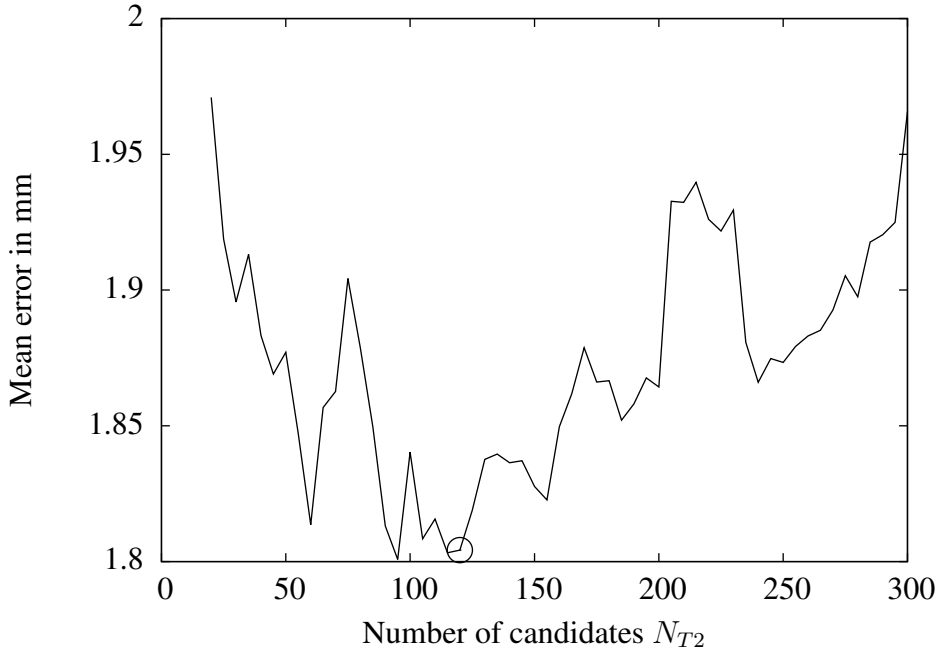


(b)

Figure 5.12: Mean segmentation error for different values of the number of surface refinement iterations (a) and the distance threshold used for clustering (b). The circles indicate the values that were selected in the other experiments.



(a)



(b)

Figure 5.13: Mean segmentation error for different values of the number N_{T1} of translation candidates of the first translation detector (a) and the number N_{T2} of translation candidates of the second translation detector (b). The circles indicate the values that were selected in the other experiments.

cases. Here, our method occasionally failed to properly find the esophagus boundary. If the 20% most difficult cases are excluded (PB), the mean error was 1.34 mm.

The surface refinement step has a significant impact on the accuracy: If it is omitted (NS), the error raises to 2.24 mm.

To evaluate the effect of the soft probability map $A(\boldsymbol{t})$, we measured the accuracy when only the binary air model $B(\boldsymbol{t})$ is used (B). The resulting error is 1.88 mm, meaning that $A(\boldsymbol{t})$ improves the accuracy by 4.3%. If also $B(\boldsymbol{t})$ is omitted (NA), the error is 1.94 mm, which means that modelling the distribution of air explicitly leads to an improvement of 7.2%. Using a second classifier for translation improves performance by 13%: If it is omitted (ST), the error raises to 2.07 mm. Omitting both the air model and the second translation detector (NAT) leads to an error of 2.32 mm.

In addition to a Markov order of one, we measured the error for orders of zero (M0) and two (M2). An order of zero means the detected ellipses in neighboring slices do not influence each other. The second order Markov model as described in section 5.2.3 also takes the curvature of the esophagus into account. A Markov order of zero yields an mean error of 2.30 mm, which shows that the Markov model clearly improves detection performance by resolving ambiguities. A Markov order of two does not further improve the performance. Therefore, we propose to use an order of one as it does not introduce unnecessary complexity.

With a mean error of 5.39 mm, the particle filtering (PF) approach described in section 5.2.3 performs poorly. The resulting segmentation was usually completely off the true esophagus. We observed that particle filtering is much more prone to tracking loss compared to the Markov chain based "detect and connect" approach. The reason is that in the "detect and connect" approach, each slice is searched exhaustively in the position detection step, while the particle filter does not search exhaustively but only evaluates the particles. Occasionally, the esophagus is hard to see on a number of slices. The PF often follows another structure, e.g. a vessel, and no particles remain on the true esophagus. Once the PF loses track, it usually does not recover. We observed that the "detect and connect" approach recovers much better after passing a difficult section.

In order to compare the performance of the detector to the performance of a human, we did an experiment on the inter observer variability (IOV): In ten datasets, the esophagus was manually segmented a second time by another person, and the second segmentations were treated like automatic ones. The mean error was 0.78 mm with a standard deviation of 0.17 mm.

Next, we evaluated how the presence of contrast agent in the esophagus affects the detector performance. For thorax-abdomen scans, patients usually drink 1.5 l of contrast agent over a 60 minutes period in preparation for the scan in order to contrast the digestive system, especially the intestine that is hard to see otherwise.

In 119 out of the 144 datasets, we did not see remains of orally given contrast agent (CA) inside the esophagus. In 19 datasets, the esophagus contains only small amounts of CA, and also only in some sections of the esophagus. Visually, the CA hardly makes a difference. In 6 datasets, the esophagus is filled with large amounts of contrast agent. The diameter of the esophagus is greatly increased. These 6 cases look very different from the remaining 138.

We do not have the medical findings of our images and therefore we neither do know from what a patient suffered and why s/he was scanned nor details about the examina-

Subgroup	num. datasets	mean err. in mm	Hausdorff dist. in mm
NC Not contrasted	119	1.62 ± 0.84	11.73 ± 6.44
HC Hardly contrasted	19	1.69 ± 0.68	12.00 ± 5.33
CD Contr. and dilated	6	5.75 ± 1.02	28.53 ± 1.88

Table 5.5: Detector performance depending on whether orally given contrast agent is visible in the esophagus.

tion. But in these 6 cases most likely 0.5 l of CA were administered when the patient was already lying on the table, directly before s/he was scanned. In combination with given Butylscopolamine, larger amounts of CA remain in the esophagus. This is typically done to contrast the upper digestive system and the esophagus itself. Some of the 6 patients may have suffered from achalasia, which means that the lower part of the esophagus does not properly open. It is also possible that parts of the esophagus were resected due to cancer and replaced with intestine tissue.

We refer to the 119 datasets as "not contrasted" (NC), to the 19 datasets as "hardly contrasted" (HC) and to the 6 datasets as "contrasted and dilated" (CD). The results can be found in Table 5.5. The performance of our proposed method on the "not contrasted" and "hardly contrasted" datasets is similar and better than the 1.80 mm average segmentation error. On the six "contrasted and dilated" datasets, the performance is much worse. The esophagus is greatly increased in diameter in these cases and mostly bigger than the aorta. Furthermore, the contrast agent leads to a unusual appearance. Here, the automatic segmentation did not cover the whole cross section of the esophagus.

Generally, machine learning methods often have problems with rare extreme cases. We think that cases like these can be handled in principle by our method as long as there are enough examples in the training data, but we did not empirically verify this.

We also evaluated how the mean error depends on different parameter settings. The results can be seen in Figure 5.12. When the number of surface refinement iterations is varied (a), at first the error steeply drops, reaches an optimum after 2-3 iterations, and rises again. We therefore kept the number of iterations fixed to two, which gives not only good results but is also computationally efficient. Figure 5.12 (b) shows results for different values of the distance threshold θ_d used for clustering. If this value is too low, the number of clusters K will be high, and for $\theta_d = 0$ equal to the number N of ellipse candidates. Then, clustering is unable to find modes in the distribution of the ellipse candidates. On the other hand, if θ_d is too high, most or even all candidates fall into the same cluster. In this case, there are no different hypothesis about the esophagus contour in a slice any more, and the Markov model becomes ineffective. Values between 6 mm and 8 mm performed best. In (c), the number of candidates N_{T1} generated by the first translation detector is varied. Selecting a too low value introduces the risk of missing the true esophagus, while a very high value means that many false alarms are propagated to further levels of the detection cascade. A value of $N_{T1} = 400$ is a reasonable choice. The number of candidates N_{T2} generated by the second translation detector (d) must be considerably lower than N_{T1}, otherwise the stage could be omitted. Low values are also computationally less expensive because less candidates have to be examined in the later stages of the cascade. But again a

Database	num. datasets	slice thickness in mm	covered body region	manual segmentation type
Thick slice	144	5	thorax or thorax-abdomen	full
Thin slice	10	0.5-0.8	thorax or thorax-abdomen	full
LIDC	27	3-5	thorax	slice
Superset of Fieselmann et al. [Fies 08a]	36	0.6-1.5	heart	slice
Fieselmann et al. [Fies 08a]	8	0.6-1.5	heart	full
Kurugol et al. [Kuru 10]	8	3.75	thorax	full
Rousson et al. [Rous 06]	20	unknown	heart	full

Table 5.6: Rows 1-4: Datasets used for evaluation in this chapter. Rows 5-8: Datasets used for evaluation in prior work.

too low value bears the risk of losing the true esophagus. Here, a value of $N_{T2} = 120$ was chosen.

5.3.2 Results on further datasets

We furthermore evaluated our method on other image databases that are listed in Table 5.6 together with databases used for evaluation in [Fies 08a, Kuru 10, Rous 06]. The evaluation results are shown in Table 5.7.

The first row of Table 5.7 shows the results of cross-validation of our method on the 144 datasets, referred to as "thick slice" data, as described above. Rows 2-4 show the performance of our method on three further test databases. In these three experiments, our model was trained on all 144 volume images from the "thick slice" database. We did not train on other databases because it is hard to obtain a sufficient number of ground truth segmentations.

Our 144 datasets all have a slice spacing of 5 mm. To obtain results on thin slice data, we evaluated our method on ten high-resolution datasets with a slice thickness in the range of 0.5-0.8 mm (second row of Table 5.7). An expert-reviewed ground truth segmentation

Method	fully automatic	test data	training data
Proposed method	yes	thick slice	thick slice
Proposed method	yes	thin slice	thick slice
Proposed method	yes	LIDC	thick slice
Proposed method	yes	superset of [Fies 08a]	thick slice
Fieselmann et al. [Fies 08a]	no	Fieselmann et al. [Fies 08a]	n/a
Kurugol et al. [Kuru 10]	no	Kurugol et al. [Kuru 10]	Kurugol et al. [Kuru 10]
Rousson et al. [Rous 06]	no	Rousson et al. [Rous 06]	Rousson et al. [Rous 06]

Method	cross-validation	mean err. in mm	Hausdorff dist. in mm	Dice coeff.
Proposed method	yes	1.80 ± 1.17	12.62 ± 7.01	0.74 ± 0.14
Proposed method	no	2.76 ± 2.76	14.91 ± 15.47	0.67 ± 0.21
Proposed method	no	1.36 ± 0.44	7.19 ± 2.78	0.73 ± 0.08
Proposed method	no	1.82 ± 1.27	9.64 ± 6.24	0.72 ± 0.11
Fieselmann et al. [Fies 08a]	n/a	unknown	unknown	0.60-0.84*
Kurugol et al. [Kuru 10]	yes	2.6 ± 2.1	unknown	unknown
Rousson et al. [Rous 06]	yes	unknown	unknown	0.80

Table 5.7: Performance on other datasets and comparison with other methods. *: Depending on the amount of user interaction.

was available for each of the ten datasets. With a mean segmentation error of 2.76 mm, the performance is considerably worse than the cross-validation error. One reason is an outlier case with a very high segmentation error caused by an air pocket of the lung that distracted the detector. The images are also noisier, and no thin slice data was included in the training set.

Next, our method was evaluated on publicly available data. Since we are not aware of a public chest CT database with images optimized for soft tissue, we selected a set of images from the database of the Lung Image Database Consortium (LIDC)[1]. We used all except one of the 28 volume images of the LIDC database that show the thorax and have a slice thickness between 3 mm and 5 mm. One volume image was excluded because of a rotated coordinate system, which is currently not handled by our method. In order to reduce the effort for the manual annotation, the data was not completely segmented. Instead, the contour of the esophagus was manually drawn in six cross-sectional slices and reviewed by an expert. Our method's mean error of 1.36 mm on this data is even better than the results of cross-validation. One reason is that there are no extreme cases among the datasets.

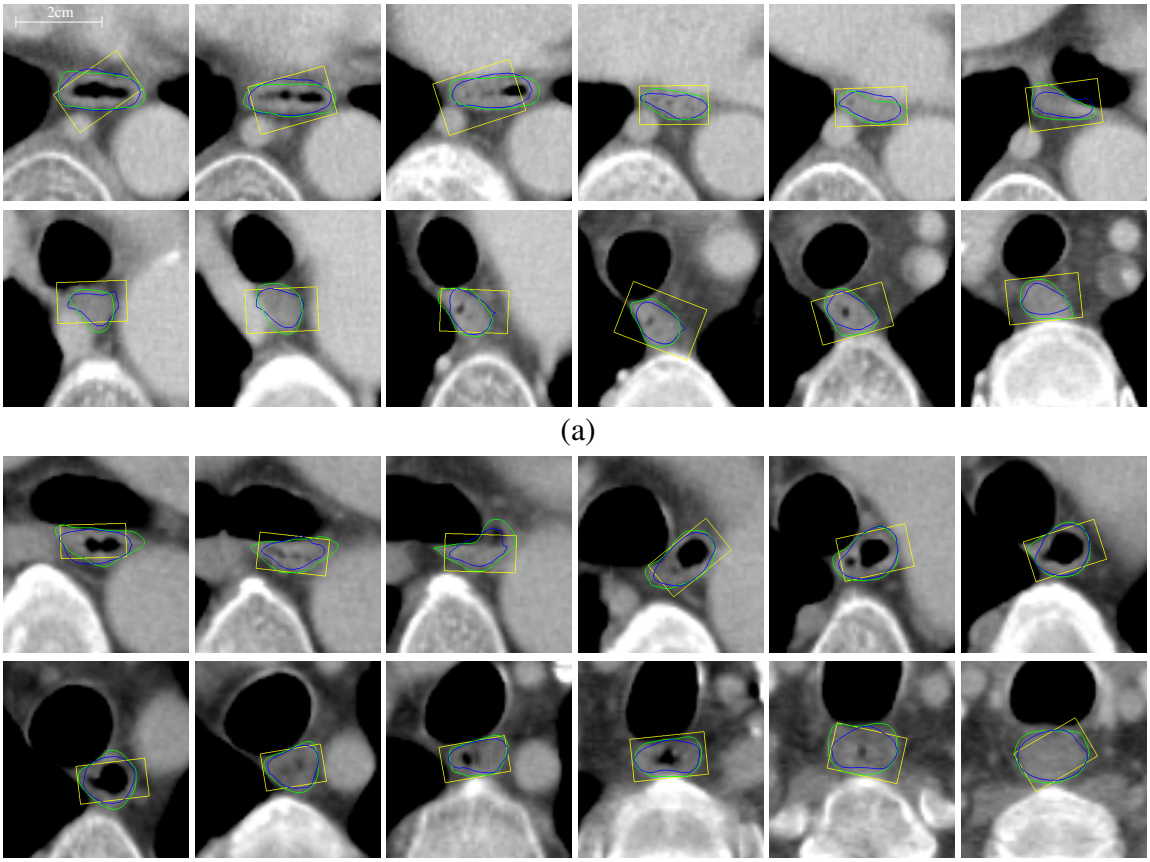
Finally, we were able to evaluate our method on a superset of the data that was used for evaluation in [Fies 08a]. We use a superset because it is not known on which of the images the method of [Fies 08a] was evaluated on. We also did not have the original ground truth data and manually annotated the esophagus contour in four axial slices in each volume image. Only four instead of six slices were annotated because the images cover a shorter segment of the esophagus. These annotations were reviewed by an expert as well. Even though the data is a superset and the ground truth may be slightly different, it still allows a relatively fair comparison. Our method achieved a mean segmentation error of 1.82 mm and a Dice coefficient of 0.72. In [Fies 08a], a Dice coefficient in the range of 0.60-0.84 is reported. The result depends on whether the user draws two, three or five contours manually into axial slices. While the result of [Fies 08a] is better if five contours are manually annotated, our automatic method outperforms [Fies 08a] with two manually drawn contours.

For comparison, the results stated in [Kuru 10] and [Rous 06] are shown in rows six and seven of Table 5.7. Our method performed considerably better than the mean error of 2.6 mm reported in [Kuru 10] on three of four databases it was evaluated on. The Dice coefficient reported in [Rous 06] is better than ours. However, the comparability is limited because the methods of [Fies 08a, Kuru 10, Rous 06] all require user interaction, while our method does not. In [Fies 08a, Rous 06], the focus was on the section of the esophagus close to the left atrium, which is relatively short. Given two points on the centerline as used in [Rous 06], the trivial centerline estimate, which is the linear interpolation of the points, can already be close to the true centerline.

5.3.3 Examples and runtime requirements

Figure 5.14 shows examples of segmentation results of the proposed method in blue along with the corresponding ground truth in green. Axial cross sections of two volumes are shown in (a) and (b). The yellow boxes show the result of the path inference step. They are the tight bounding boxes of the ellipses which approximate the esophagus contour.

[1] `https://imaging.nci.nih.gov`



(a)

(b)

Figure 5.14: Examples of segmentation results on unseen data. Axial slices are shown for two datasets (a) and (b). Blue is the automatic segmentation, green is the ground truth, and the yellow boxes show the inferred path. The mean errors of the segmentations are 0.95 mm (a) and 0.88 mm (b). The bar in the top left slice indicates the scale.

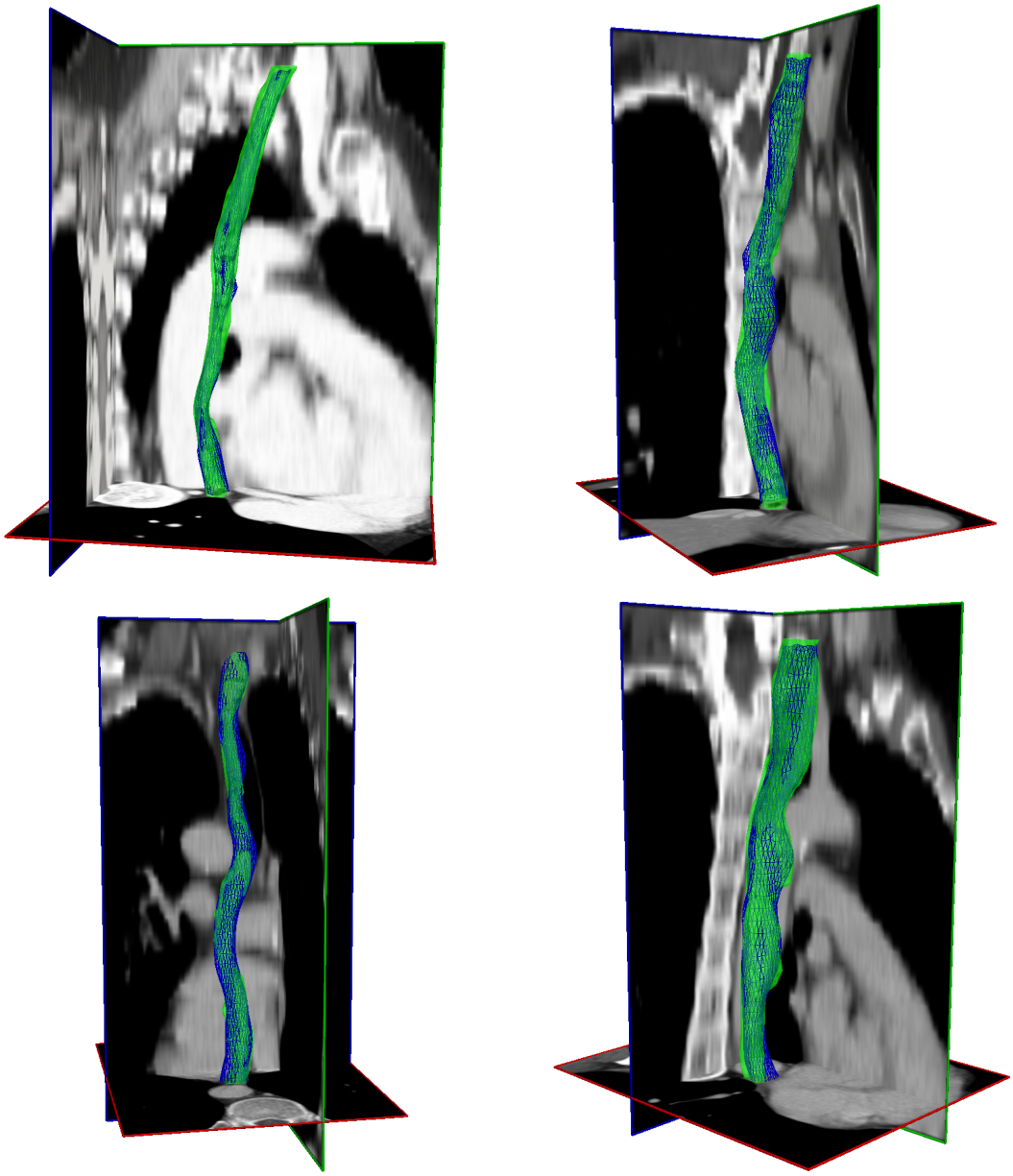


Figure 5.15: Examples of segmentation results on unseen data, shown in 3-D. Blue is the automatic segmentation and green is the ground truth. The mean errors of the segmentations are from left to right and top to bottom: 0.95 mm (same datasets as 5.14 (a)), 1.15 mm, 0.99 mm and 0.95 mm.

ROI detec.	prob. map generation	ellipse detec.	path inference	refinement	**total**
6.96	1.13	7.40	$0.40 \cdot 10^{-3}$	0.34	**15.83**

Table 5.8: Runtime in seconds for different steps of the method.

In Figure 5.15, four example segmentations are displayed in 3-D. These 3-D images also visualize the size of the ROI. All shown datasets were not included in the training data.

Table 5.8 shows the runtime requirements of the different steps of the proposed method. It was measured on a single CT scan of the entire torso on a standard PC with a 2.20 GHz dual core CPU. With 6.96 s, ROI detection is the second most time consuming step because the whole volume is searched exhaustively. The remaining steps only consider the ROI. Ellipse contour candidate detection including clustering takes 7.40 s and is the most time consuming step. Generating the probability map based on air and surface refinement is comparatively inexpensive, and the time needed for path inference is negligible. In total, segmenting the esophagus from a CT volume takes less than 16 seconds.

5.4 Conclusion

We have presented a fully automatic method for esophagus segmentation from CT scans. An ROI is detected by finding salient anatomical landmarks. A powerful detector that learned a discriminative model of the appearance and an explicit model of the distribution of air is combined with prior knowledge about the esophagus shape. It is used to infer the approximate contour of the esophagus by finding the maximum a posteriori estimate. Two alternative methods for shape knowledge representation and inference are compared: A "detect and connect" approach using a Markov chain model, and a particle filter. The influence of different model parameters on the performance was evaluated. In particular, the Markov chain model was evaluated with Markov orders of zero, one, and two. Finally, a surface is generated and further adapted to better fit the boundary, again using discriminative learning.

The accuracy was measured using cross validation on 144 datasets. We found that the Markov chain based "detect and connect" approach can well handle difficult regions and resolve ambiguities. It performed clearly better than the particle filter, which is much more prone to tracking loss. Explicitly modelling respiratory and esophageal air to support the appearance based detector improves the mean error by 7.2%. Our proposed method segments the esophagus from a CT scan without user interaction with a mean error of 1.80 mm in less than 16 s, which is only 1 mm above the inter observer variability.

Apart from the ROI detection and explicitly modelling air, the method is not specific to the esophagus and can easily be adapted to other tubular structures like the spinal chord or larger vessels.

In the following chapter, the method will be used to support lymph node detection in chest CT images by excluding the segmentation from search and putting more focus on the neighborhood, where lymph nodes are especially common.

Chapter 6

Lymph node detection and segmentation from chest CT

Parts of the work presented in this chapter were previously published in [Feul 10a] and [Feul 11b]. It is all original work of the thesis' author.

6.1 Motivation

This chapter focuses on detecting mediastinal lymph nodes in chest CT images. It builds upon the esophagus segmentation method of the previous chapter. Lymph nodes are often located close to the esophagus, and the outline of the esophagus therefore gives a valuable hint about where lymph nodes can be expected.

Lymph nodes play an important role in clinical practice, especially in the mediastinal area. They routinely need to be considered during oncological examination related to all kinds of cancer [Duwe 05, Lang 06], for instance lung cancer [McLo 92], where metastases settle in lymph nodes, but also lymphoma, which is a cancer of the lymphatic system itself. Furthermore, they are also relevant in case of inflammation in general.

Cancer causes affected lymph nodes to be enlarged. In order to assess the progress of the disease and to check whether treatment is effective, physicians are interested in statistics like the number of enlarged nodes or the total volume of the nodes, but also in the spatial distribution, and changes over time. Patients are commonly examined using CT.

Manually counting and measuring lymph nodes in the images is not only cumbersome but also error prone because annotations from different human observers and even from the same human observer vary significantly. In practice, lymph nodes are not annotated individually because it would take too much time, though the clinical value would be high[1]. An automatic detection is however challenging because lymph nodes have an attenuation coefficient similar to muscles and vessels and therefore low contrast to surrounding structures. Moreover, their shape and size varies a lot. Even a human needs days of training to consistently find lymph nodes in CT volume images. Examples of mediastinal lymph nodes are shown in Figure 6.1.

The topic has received increasing attention in the last five years. In [Kita 07], two blob detectors which are called 3-D Min-DD filter and extended 3-D Min-DD filter are used in a

[1] according to A. Cavallaro, a radiologist at the university clinics in Erlangen

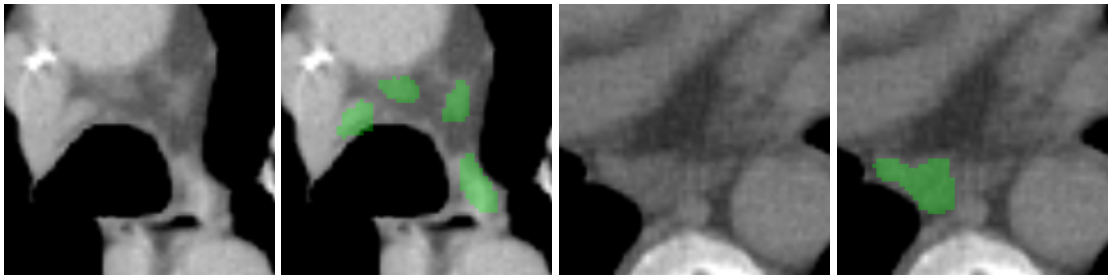


Figure 6.1: Two axial cross sections of CT volumes with expert-reviewed lymph node annotations (green).

cascade to detect lymph nodes in abdominal CT data. A Hessian based vessel detector, the CT Hounsfield units and morphological operations are used to reduce the number of false positives. In [Feue 09], a similar approach is used to detect lymph nodes in chest CT. Here, the first 3-D Min-DD filter is replaced with a Hessian based blob detector. The more expensive extended 3-D Min-DD filter is used at the second level of the cascade. For segmentation, a model-based approach using mass spring models was proposed in [Dorn 06]. It was also used for detection by placing models on a regular grid over the volume [Dorn 08]. A lymph node was assumed at positions where the model fitting converged with a good score. In [Feul 10a, Barb 10, Feul 11b], data driven approaches for lymph node detection were proposed. In all three cases, a discriminative model is trained to detect lymph nodes in CT from their appearance. In [Feul 10a, Feul 11b], the focus was on the mediastinal region (the region between the lungs), and the discriminative model was combined with prior anatomical knowledge that is modeled as a spatial prior probability. In [Barb 10], the focus was on the axillary region. After detection, lymph nodes are also segmented by fitting a sphere model to the image that is centered at the detection. The result of the segmentation is used to improve the detection: A good segmentation result indicates a good detection. A similar technique is used in [Feul 11b]: In a final step, each detection is verified by initializing a segmentation algorithm with the detection. But instead of fitting a sphere, graph cuts are adapted to the problem of lymph nodes segmentation. Features are extracted from the segmentation and used to train a classifier to learn whether a segmentation is a true or a false alarm.

This chapter is an extension of our prior work [Feul 10a] and [Feul 11b]. Figure 6.2 gives an overview of our system for lymph node detection and segmentation. Here, method A corresponds to [Feul 10a], and method B is based on [Feul 11b]. Both approaches make use of a cascade of binary classifiers. The first two stages of the cascade use 3-D Haar like features to generate a set of candidate lymph node centers. In method A, the final binary classifier of the cascade is trained to decide whether there is a lymph node with a certain size at a given position that comes from the second stage. Bounding boxes of lymph node position detections are estimated by evaluating different size hypothesis. In method B, the size detection stage is replaced by two other stages. In stage 3 of method B, the detected position candidates from stage 2 are verified by a third classifier that now uses gradient-aligned features as proposed in [Barb 10]. The result is again a set of position candidates. Finally, a segmentation algorithm is initialized with the lymph node position candidates. The graph cuts method is adapted to the problem of lymph nodes segmentation by incorporating knowledge about the shape and the appearance of the node into the segmentation

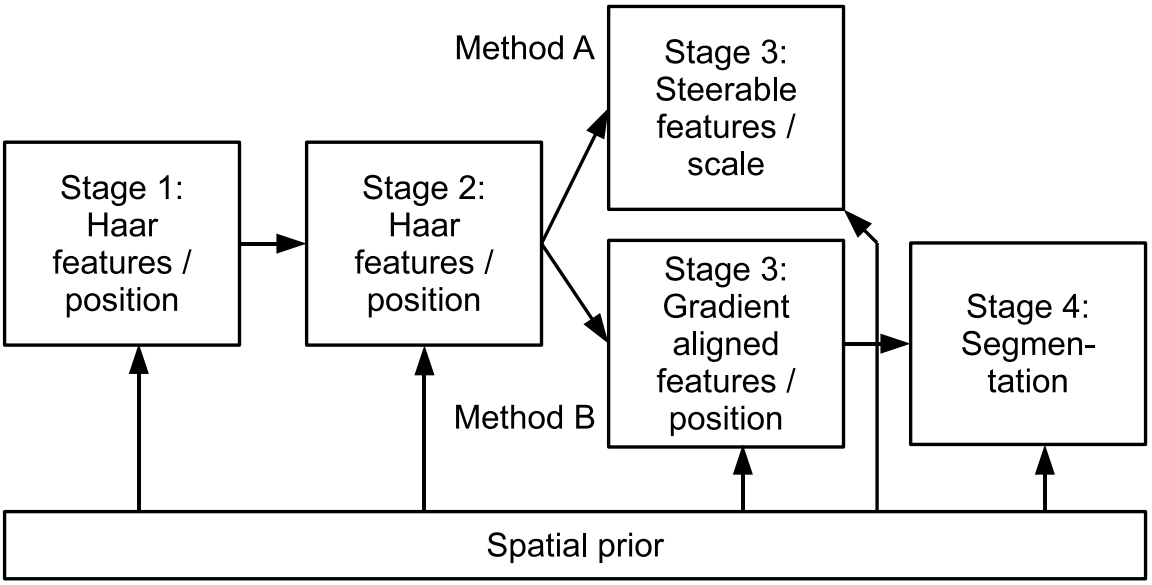


Figure 6.2: Overview of the detection and segmentation system. Lymph nodes are detected by a cascade of binary classifiers. The first two stages detect the center of a lymph node using 3-D Haar-like features. Then either the bounding box of a lymph node is detected directly (Method A), or alternatively, the detected positions are verified using gradient-aligned features in stage 3 (Method B). Method A ends at stage 3. In stage 4 of Method B, each position candidate from stage 3 is used to initialize a segmentation algorithm, features are extracted from the segmentation, and based on these features, a classifier decides whether the detection is a true lymph node or a false alarm. At all stages, prior anatomical knowledge is included in the form of a spatial prior probability, which is the same for all stages.

framework. A feature set is extracted from the segmentation, and a fourth classifier is trained to learn whether a segmentation is a true lymph node or not.

This chapter is extended in several ways with respect to [Feul 10a] and [Feul 11b]. Both approaches are explained in more detail and thoroughly compared to each other. Next, the method of [Feul 11b] is compared to a method that does not only use a single segmentation at the final verification stage 4, but instead combines hints from multiple alternative segmentations. Finally, this chapter contains experiments with different types of spatial priors.

The remainder of this chapter is structured as follows: Section 6.2 explains the different kinds of spatial priors that are used in this work to model anatomical knowledge. Section 6.3 describes how the region of interest containing the mediastinum is determined. Section 6.4 explains the first two stages of the detection cascade that method A and method B have in common. Section 6.5 explains the final bounding box detection step of method A that builds upon the lymph node center candidates generated in the first two stages. In section 6.6, method B is explained that segments lymph nodes instead of detecting bounding boxes and rejects or accepts them based on the segmentation result. Section 6.7 presents experiments and results, and section 6.8 concludes the chapter.

6.2 Spatial prior of lymphatic tissue

Mediastinal lymph nodes are very hard to detect only from their appearance. They have a similar attenuation coefficient like muscles and vessels, and both muscles and vessels cover a much larger volume of the body. Thus, an automatic detector has to cope with lots of clutter. Furthermore, the size of a lymph node can vary a lot. While healthy lymph nodes typically have a size in the range of a few millimeters up to one or two centimeters [Warw 58], lymph nodes that are pathologically enlarged, for instance, due to cancer or an infection, can have a size of five centimeters and more. Often, multiple enlarged lymph nodes are directly adjacent to each other and form clusters. Detecting lymph nodes in a cluster is especially challenging because the boundary between different nodes is often not clearly visible, or not visible at all. Then, the shape of the cluster can be almost arbitrary.

Because of these difficulties, it is vital to incorporate as much prior knowledge as possible into the detection. In particular, we know that

- lymph nodes do not appear anywhere. They always lie in fat tissue, so space inside any organ can be excluded.

- In the remaining area, lymph nodes are not distributed uniformly. Instead, it is much more likely to observe lymph nodes below the aortic arch and close to the trachea.

It turns out that exploiting this prior knowledge can help to greatly reduce the number of false detections and thus improve the overall detection performance.

In this work, this knowledge is modeled using a spatial prior probability $p(m = 1|\boldsymbol{t})$ of observing a lymph node at a given location $\boldsymbol{t} = (t_x, t_y, t_z)^T$. m denotes the binary class variable. Four different priors are proposed and compared against each other. These four variants partly build upon each other and are increasingly complex.

6.2.1 Automatic landmark detection and organ segmentation

While variant 1 is a trivial prior, the variants 2-4 depend on anatomical structures that first need to be detected in a CT volume image. We automatically find a set of 20 salient anatomical landmarks that lie mostly but not exclusively in the chest area and can be detected robustly. The detection method used here is described in [Seif 09]. Examples of landmarks are the bifurcation of the trachea, the bottom tip of the shoulder blade left and right, the topmost point of the aortic arch and the topmost point of the lung left and right.

Besides the landmarks, a number of different organs are segmented. The lungs and the trachea are detected using simple thresholding followed by a morphological opening operation. The four heart chambers are segmented by fitting a model described in [Zhen 07]. The esophagus is segmented as described in chapter 5. It is of special interest as it is often surrounded by lymph nodes, but at the same time can be confused with lymphatic tissue. All segmentation methods do not require user interaction.

6.2.2 Variant 1: Constant prior

In variant 1, the probability $p(m = 1|\boldsymbol{t})$ is simply modeled to be constant

$$p_1(m = 1|\boldsymbol{t}) \propto \text{const.}, \tag{6.1}$$

which means that no prior knowledge is used. This serves as a baseline for the remaining three variants.

6.2.3 Variant 2: Binary mask

In the second variant, the spatial prior is modeled to be proportional to a binary mask $B(\boldsymbol{t})$

$$p_2(m = 1|\boldsymbol{t}) \propto B(\boldsymbol{t}) = \begin{cases} 0 & \text{if } \boldsymbol{t} \text{ is inside an organ} \\ 1 & \text{otherwise} \end{cases} \tag{6.2}$$

that labels regions that cannot contain lymph nodes with 0 and other regions with 1. The lungs, the trachea, the esophagus and the heart are excluded, i. e. labeled with zero in the mask.

6.2.4 Variant 3: Global prior

The third variant consists of the binary mask $B(\boldsymbol{t})$ and a global probabilistic atlas

$$G(\boldsymbol{t}) \in [0, 1] \tag{6.3}$$

which is learned in the space of a reference patient. Non-rigid inter subject registration is used to map segmented lymph nodes from a set of test patients to the reference patient, where they are averaged. The segmentations are binary masks, and thus $G(\boldsymbol{t})$ is the spatial probability of lymphatic tissue. The learned probabilistic atlas is blurred with a Gaussian filter with a standard deviation of 12 mm which is necessary because of limited training data.

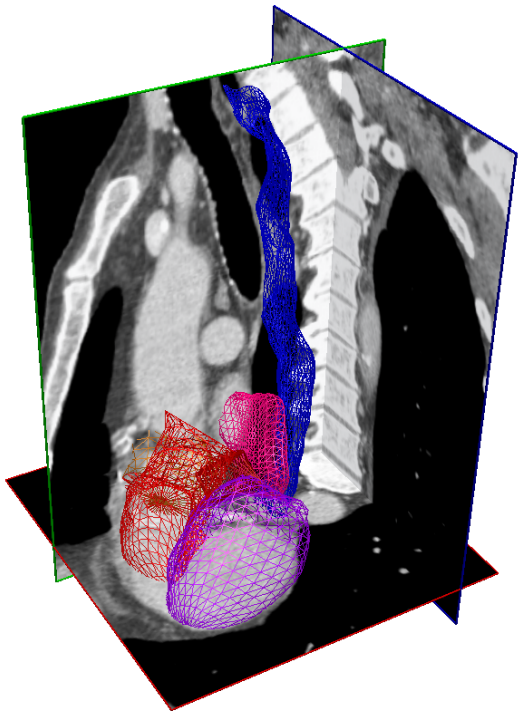


Figure 6.3: Heart and esophagus model fitted to a CT volume.

The registration is based on the set of 20 landmarks. If a landmark is not detected, e. g. because it is not visible in the image, it is omitted. A thin-plate spline (TPS) transformation [Book 89] is created from the detected landmarks and the reference landmarks and used for the warping. During training, the transformation maps from the reference space to the current image, and for the testing phase, it maps into the other direction.

The prior is then modeled to be

$$p_3(m = 1|\boldsymbol{t}) \propto B(\boldsymbol{t})G(\boldsymbol{t}) \tag{6.4}$$

the product of the binary mask and the probabilistic atlas.

6.2.5 Variant 4: Organ-specific local priors

A possible disadvantage of the global probabilistic atlas G is limited accuracy in case of inter-subject variations of anatomy or missing landmarks on points or surfaces of interest. In general, the problem of finding a transformation f that maps from one image to another is ill-posed if certain points or regions, like for instance a gap between two organs, only exist in one, or if the relative position of two structures, e.g. the esophagus and the trachea, is different in the two images. This is a problem all atlas based techniques have to cope with.

In this work, we investigate using a mixture of spatially weighted probabilistic atlases as a solution to this problem. In addition to the landmark based global probabilistic atlas G, a set of local probabilistic atlases $L_i, i = 1 \ldots M$ is introduced. We already have

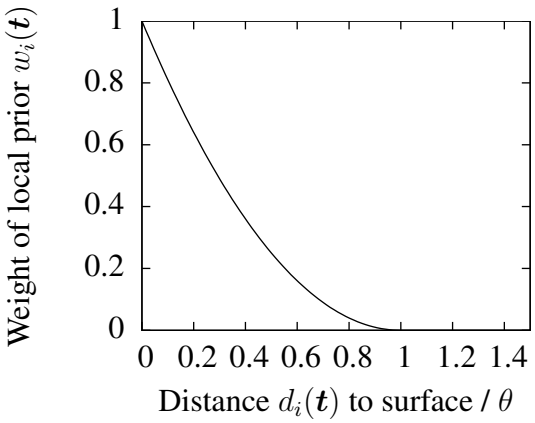


Figure 6.4: Local probabilistic atlas weighting function.

the heart and the esophagus available, which are represented as triangular meshes. The four chambers of the heart are treated like separate organs. The vertices of these meshes can be used as landmarks for a TPS based registration like in case of the global atlas G. If they were simply added to the existing set of landmarks, however, the resulting TPS transformation would easily become rugged and produce folds or become inaccurate if it was constrained to be smooth because of the problems described above. Here, each segmented organ is associated with a probabilistic atlas L_i. It is learned in the space of the organ's mean shape, which is generated by averaging the points of a set of training shapes that were aligned by a generalized Procrustes analysis. Each local atlas $L_i(\boldsymbol{t})$ is trained like the global, but now the vertices of the organ's triangle mesh serve as landmarks for the TPS transformation. In the testing phase, the final local atlas $L(\boldsymbol{t})$ for a test volume is the sum of the local atlases, weighted with $w_i(\boldsymbol{t})$ according to the minimum distance $d_i(\boldsymbol{t})$ of $\boldsymbol{t}$ to the surface of organ i:

$$L(\boldsymbol{t}) = \frac{1}{Z(\boldsymbol{t})} \sum_{i=1}^{M} L_i(\boldsymbol{t}) w_i(\boldsymbol{t}) \tag{6.5}$$

$$Z(\boldsymbol{t}) = \sum_{i=1}^{M} w_i(\boldsymbol{t}) \tag{6.6}$$

$$w_i(\boldsymbol{t}) = \begin{cases} \frac{(d_i(\boldsymbol{t})-\theta)^2}{\theta^2} & \text{if } d_i(\boldsymbol{t}) < \theta \\ 0 & \text{else.} \end{cases} \tag{6.7}$$

Here, θ denotes the maximal distance from the organ surface up to that the corresponding probabilistic atlas still has support. In the experiments, θ was set to 25 mm. The weight w_i decreases with increasing distance because the TPS transformation is only accurate close to the surface. The distance weight function (6.7) is shown in Figure 6.4.

Now the binary mask B, the global probabilistic atlas G and the local probabilistic atlas L are merged into the fourth variant $p_4(m = 1|\boldsymbol{t})$ of the prior probability of observing a lymph node at position $\boldsymbol{t}$. It is modeled as

$$p_4(m = 1|\boldsymbol{t}) \propto B(\boldsymbol{t}) \left(w_{\max}(\boldsymbol{t}) L(\boldsymbol{t}) + (1 - w_{\max}(\boldsymbol{t})) G(\boldsymbol{t}) \right), \tag{6.8}$$

where $w_{\max}$ is the maximum weight

$$w_{\max}(\boldsymbol{t}) = \max \left(w_1(\boldsymbol{t}), \ldots, w_M(\boldsymbol{t}) \right). \tag{6.9}$$

Thus, the term on the right hand side of (6.8) is zero inside an organ. Elsewhere, the global atlas $G(\boldsymbol{t})$ and the local atlas $L(\boldsymbol{t})$ are blended according to the minimum distance of $\boldsymbol{t}$ to the surface of any organ. Directly at an organ surface, $w_{\max}(\boldsymbol{t})$ equals one and $p_4(m = 1|\boldsymbol{t})$ only depends on the local atlas $L(\boldsymbol{t})$, while $p_4(m = 1|\boldsymbol{t})$ only depends on the global atlas $G(\boldsymbol{t})$ at a minimum distance of θ from any organ surface.

Figure 6.5 shows examples of the different prior types along with the original volume image they were computed from. Each image in a column shows the same slice of the volume, and the slices are parallel to the coordinate planes. The binary prior $p_2(m = 1|\boldsymbol{t})$ shown in Figure 6.5 (b) excludes already considerable portions of the volume. The "global" prior $p_3(m = 1|\boldsymbol{t})$ shown in Figure 6.5 (c) and the "combined" prior $p_4(m = 1|\boldsymbol{t})$ shown in (d) put special focus on relatively small regions of the volume. Differences between these two prior types are, however, small.

6.3 Region of interest detection

This work focuses on detecting mediastinal lymph nodes. Therefore, a region of interest (ROI) is automatically detected that covers the mediastinum. Similar as described in section 5.2.1, this ROI is anchored at the bifurcation of the trachea, because this landmark can be detected very robustly. It is contained in the set of landmarks that are detected as described in section 6.2.1. The ROI has a fixed minimum size of $18.4 \times 18.0 \times 19.5\text{cm}^3$ that is large enough to contain all ground truth annotations. If parts of the segmentations of the heart or the esophagus, which are used to compute the spatial prior, are outside this ROI, then it is enlarged to completely contain these segmentations. Figure 6.6 shows an example of an ROI.

6.4 Position candidate detection

In this section, the spatial prior described in section 6.2 is combined with a discriminative model of the lymph node appearance. We are finally interested in either a bounding box of a lymph node, or, even better, the precise outline. Detecting the bounding box or the outline in one step would, however, require the estimation of many parameters simultaneously. Instead of directly searching a high dimensional parameter space, we break the detection into smaller sub problems. At first, we only detect a number of candidates of possible lymph node center positions. In later steps, these candidates are verified or rejected, and used to initialize detectors for the lymph node size or the actual lymph node segmentation.

Stage one of the detection system (Figure 6.2) is a sliding window detector that uses a probabilistic boosting tree (PBT) classifier (see section 2.5) in combination with 3-D Haar-like features (section 3.2.2). The classifier is trained to learn the probability

$$p(m = 1|\boldsymbol{H}(\boldsymbol{t})) \tag{6.10}$$

of whether there is a lymph node model instance at a given position $\boldsymbol{t}$. Here, $\boldsymbol{H}$ denotes the Haar feature vector extracted at position $\boldsymbol{t}$. Haar features are used because they can be computed very efficiently so that it is even possible to search all positions in the ROI exhaustively.

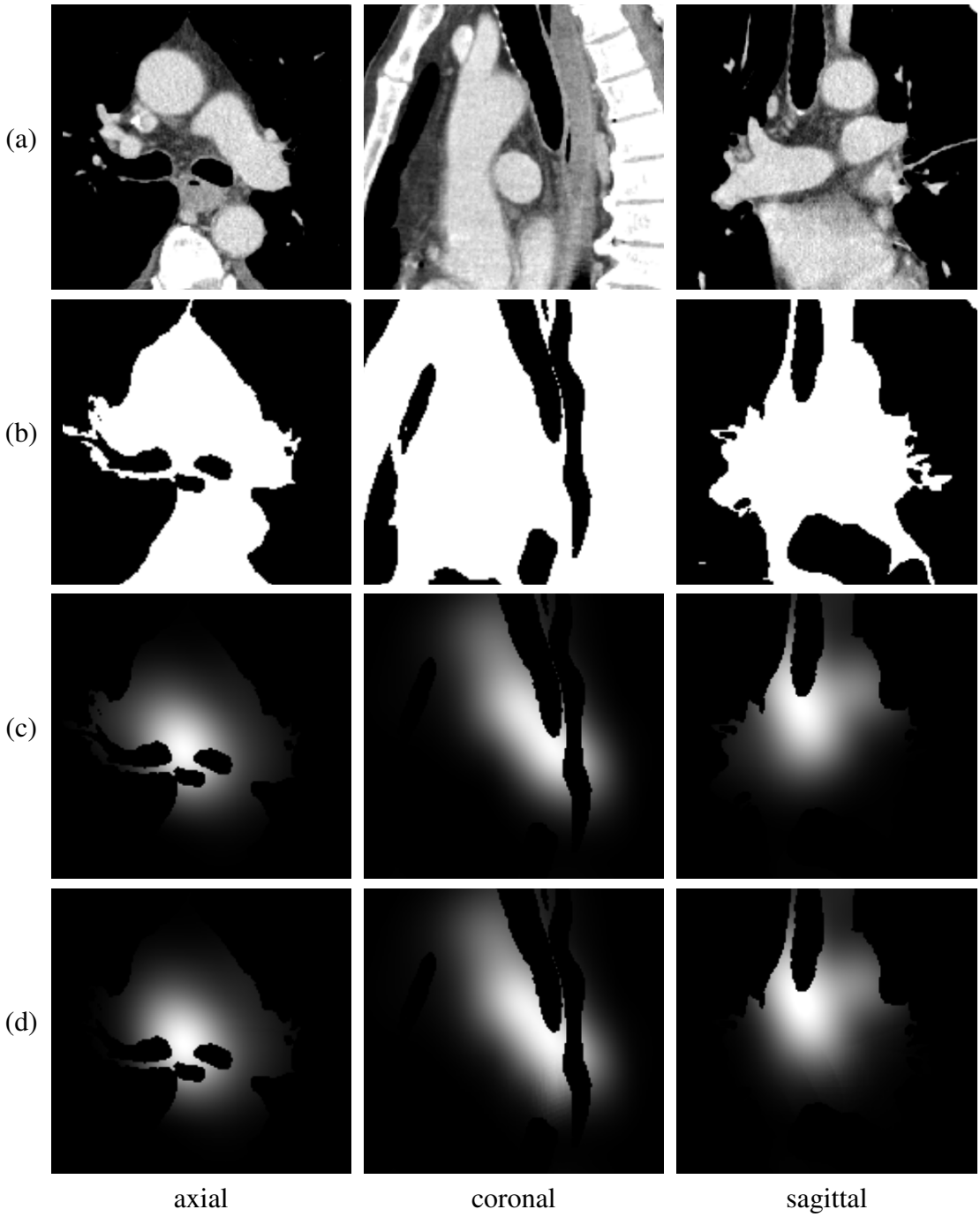


Figure 6.5: Examples of different spatial priors computed for a test volume. The three columns show axis-aligned orthogonal slices of the volume. (a): The input volume. (b): The binary prior $p_2(m = 1|\boldsymbol{t})$ (see eq. (6.2)) that excludes air and organs. (c): The "global" prior $p_3(m = 1|\boldsymbol{t})$ (see eq. (6.4)). (d): The "combined" prior $p_4(m = 1|\boldsymbol{t})$ (see (6.8)). It is generally similar to the "global" prior.

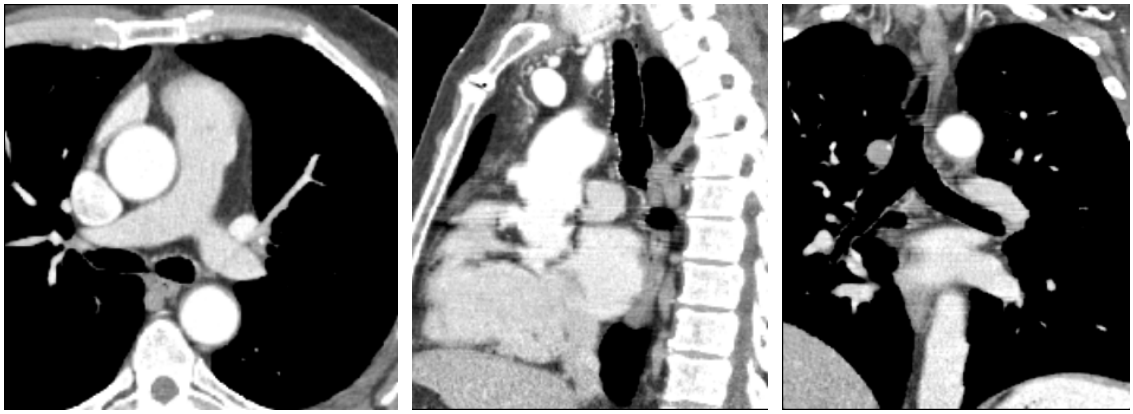


Figure 6.6: Axial, sagittal and coronal slice through the region of interest of a CT scan.

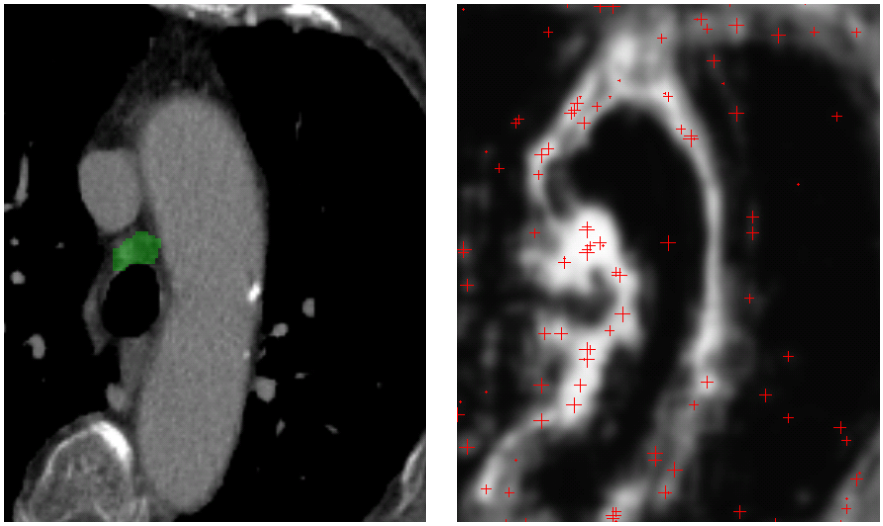


Figure 6.7: Local maxima of the probability map generated by the detector are used as position candidates. Note that this is a 2-D slice of a 3-D volume and points that look like local optima in 2-D are not necessarily local optima in 3-D.

Given the output of the classifier, a set of position candidates $C_{H1} = \{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_{|C_{H1}|}\}$ is generated. If a fixed threshold θ_{H1} is used and we select all $\boldsymbol{t}$ that satisfy $p(m = 1|\boldsymbol{H}(\boldsymbol{t})) > \theta_{H1}$, we run into the problem that lots of candidates are generated at lymph nodes which are clearly visible, but we do not get any candidates at lymph nodes which are hard to see. To overcome this, we use a technique proposed in [Chen 09]. First, a probability map is generated from the classifier output. This map is blurred using a Gaussian filter with a standard deviation of 1.5 mm, and local maxima in the probability map are selected as candidates. A standard deviation of 1.5 mm is a good compromise between a smooth probability map and not losing more details about the classifier output than necessary. An example of candidates extracted from the blurred probability map can be seen in Figure 6.7 along with the CT image data it was generated from.

Now, another PBT classifier is used to examine the position candidates in the set C_{H1} and to reject false positives, resulting in a set C_{H2} of candidates. Like the first one, the second classifier again uses 3-D Haar-like features. The difference is that the negative

examples for the training phase of this classifier are generated by scanning images using the first classifier and collecting false positives.

This set C_{H2} of lymph node center point candidates is now used in two alternative ways.

1. In the first way, the candidates are used to initialize either a detector that estimates the lymph node bounding box. This is called "method A" and further explained in section 6.5.

2. In the second way, lymph nodes are segmented, and the detected lymph node position is used as seed. The resulting segmentation is used to verify the detection result. This variant is called "method B" and explained in section 6.6.

6.5 Method A: Lymph node bounding box detection

6.5.1 Detecting the scale

So far, only the center point $\boldsymbol{t}$ of a lymph node was detected, but the size of the lymph nodes was not taken into account. The classifiers for translation are trained with data of different sizes. In this section, we are also interested in the size and want to find axes aligned bounding boxes of the lymph nodes. The boxes $\boldsymbol{b}$ are parameterized by their center $\boldsymbol{t}$ and size $\boldsymbol{s}$:

$$\boldsymbol{b} = (\boldsymbol{t}, \boldsymbol{s}) = (t_x, t_y, t_z, s_x, s_y, s_z). \tag{6.11}$$

We again follow the idea of marginal space learning proposed in [Zhen 07]: Instead of directly searching a high dimensional search space, which here consists of position $\boldsymbol{t}$ and size $\boldsymbol{s}$ parameters, candidates of position are generated as described in section 6.4 using a detector that is trained on a margin of the search space, which is spanned by $\boldsymbol{t}$ in this case. A second detector trained to learn the probability $p(m = 1|\boldsymbol{S}(\boldsymbol{t}, \boldsymbol{s}))$ of a lymph node at a given position with a given size then only needs to consider the position candidates C_{H2}, leading to an enormous speedup. Here, $\boldsymbol{S}(\boldsymbol{t}, \boldsymbol{s})$ denote steerable features (section 3.3) evaluated at position $\boldsymbol{t}$ and size $\boldsymbol{s}$. The sampling pattern is a regular grid of size $7 \times 7 \times 7$ that is scaled and translated according to $\boldsymbol{s}$ and $\boldsymbol{t}$.

6.5.2 Integrating the prior

Section 6.2 explained how a spatial prior $p(m = 1|\boldsymbol{t})$ of lymphatic tissue can be modeled, and sections 6.4 and 6.5.1 explained how lymph nodes are detected from their appearance using discriminative learning techniques. In this section, the spatial prior and the discriminative model are combined.

We start with integrating the prior into the position detection steps of section 6.4. This can be done in a similar way as the translation score of esophagus detection was combined with the probability map computed from the distribution of air in section 5.2.2. During position detection, we are interested in the probability

$$p(m = 1|\boldsymbol{H}, \boldsymbol{t}) \tag{6.12}$$

of whether there is a lymph node at a given position $\boldsymbol{t}$, with a given feature vector $\boldsymbol{H}$. With Bayes' rule, (6.12) can be reformulated as

$$p(m=1|\boldsymbol{H},\boldsymbol{t}) = \frac{p(\boldsymbol{H},\boldsymbol{t}|m=1)p(m=1)}{p(\boldsymbol{H},\boldsymbol{t})}. \tag{6.13}$$

For simplification, we assume that the feature vector $\boldsymbol{H}$ is statistically independent from the position $\boldsymbol{t}$. This is an approximation as $\boldsymbol{H}$ obviously depends on $\boldsymbol{t}$, but $\boldsymbol{t}$ determines $\boldsymbol{H}$ only for a certain image. The assumption is justified by the fact that the spatial prior clearly improves the performance as we will see, which means that $\boldsymbol{H}$ does not contain much information about $\boldsymbol{t}$.

Similar as in (5.12), (6.13) may now be transformed into

$$p(m=1|\boldsymbol{H},\boldsymbol{t}) = \frac{p(\boldsymbol{H}|m=1)p(\boldsymbol{t}|m=1)p(m=1)}{p(\boldsymbol{H})p(\boldsymbol{t})} \tag{6.14}$$

$$= \frac{p(m=1|\boldsymbol{H})p(m=1|\boldsymbol{t})}{p(m=1)}, \tag{6.15}$$

which is proportional to the product of (6.10) and the spatial prior $p(m=1|\boldsymbol{t})$ and is used as final translation detection score. A blurred map of this score is visualized in Figure 6.7 (right).

Similarly, if also scale is incorporated and we are interested in the probability of having a lymph node at a given position $\boldsymbol{t}$ with features $\boldsymbol{H}$ and $\boldsymbol{S}$, it can be expressed as

$$p(m=1|\boldsymbol{S},\boldsymbol{H},\boldsymbol{t}) = \frac{p(\boldsymbol{S},\boldsymbol{H},\boldsymbol{t}|m=1)p(m=1)}{p(\boldsymbol{S},\boldsymbol{H},\boldsymbol{t})} \tag{6.16}$$

$$= \frac{p(\boldsymbol{S}|m=1)p(\boldsymbol{H}|m=1)p(\boldsymbol{t}|m=1)p(m=1)}{p(\boldsymbol{S})p(\boldsymbol{H})p(\boldsymbol{t})} \tag{6.17}$$

$$= \frac{p(m=1|\boldsymbol{S})p(m=1|\boldsymbol{H})p(m=1|\boldsymbol{t})}{p^2(m=1)}. \tag{6.18}$$

The step from (6.16) to (6.17) is valid under the assumption that the steerable features $\boldsymbol{S}$ are statistically independent from the Haar-like features $\boldsymbol{H}$ and position $\boldsymbol{t}$. The probability (6.18) serves as final detection score. Based on it, a set

$$C_S = \{(\boldsymbol{t}_1, \boldsymbol{s}_1), \ldots, (\boldsymbol{t}_{400}, \boldsymbol{s}_{400})\} \tag{6.19}$$

that contains the 400 best translation and scale candidates is generated from the set C_{H2}.

6.6 Method B: Joint detection and segmentation

Directly detecting bounding boxes is a standard approach for object detection problems [Viol 01]. In case of lymph nodes, a segmentation contains valuable additional information such as the volume or the shape of the node. Furthermore, a segmentation can even be used to improve the quality of the detection result.

Standard features like Haar features or steerable features work well for a broad range of applications. But better performance can be achieved using features that are that are

designed for a particular problem. The idea is to not extract features at predefined locations as, for instance, on a regular grid, but on the estimated object boundary, inside, and in the neighborhood of the estimated object.

In this section, a set of problem-specific features is presented that is extracted from a candidate segmentation. In general, the quality of the segmentation result will be good if the segmentation was initialized with the center of a true lymph node, and otherwise it will be poor. Therefore, the segmentation result can give a valuable hint of whether the underlying detection is a true or a false positive. We use a variant of the graph cuts method for seeded image segmentation [Boyk 03] to segment a lymph node given its detection. Advantages of graph cuts are that the global optimum of the underlying cost function can be computed efficiently, and at the same time, it is flexible enough to allow adapting it to this particular problem, leading to a good performance.

In total, these segmentation based features are however computationally relatively expensive because the segmentation has to be carried out for every single candidate. Therefore, these features are used in the final step of the detection cascade. To avoid having to compute too many segmentations during test, an additional step is inserted into the detection cascade that reduces the number of position candidates C_{H2} generated by the second cascade level. It uses so-called gradient-aligned features. These are point features that are extracted at local maxima of the gradient magnitude. They are computationally less expensive and explained in section 6.6.1 in more detail.

6.6.1 Verifying detections using gradient aligned features

In order to reduce the number of lymph node position candidates in the set C_{H2} (see section 6.4) that is generated by the second Haar feature based detector, the candidates are verified by a detector that uses gradient-aligned features as proposed in [Barb 10].

Starting from the presumed center $\boldsymbol{t}$ of a lymph node, 14 rays are sent out in radial direction such that tree rays are parallel to the axes of the coordinate system, three are antiparallel, and the remaining eight hit the corners of an axes aligned cube placed at $\boldsymbol{t}$. Each ray is regularly sampled with a sample spacing of 1 mm. The image gradient magnitude is computed for each sample, and local maxima of the gradient magnitude above a threshold are detected along each ray. This is done for ten different threshold values, and not only on the original image, but also on two coarser versions with a voxel spacing that is two times and four times the original spacing, respectively. This resolution hierarchy is used to make the features more robust to noise.

At each of the first three local maxima of each ray, 24 simple points features are computed. These are the same point features that are also computed at each sample of the steerable feature sampling pattern as described in section 6.5.1. The same point features are computed halfway from the center $\boldsymbol{t}$ to each of the first three local optima, for each ray. Next, the distance from the center to each of the first three local optima is used as a feature. Finally, asymmetry is captured by measuring the differences of the distances of corresponding local optima to the center for all pairs of different rays.

Now an AdaBoost classifier is trained to learn the probability $p(m = 1|\boldsymbol{A}(\boldsymbol{t}))$ of whether there is a true lymph node ($m = 1$) given the gradient aligned feature vector $\boldsymbol{A}$ extracted at position $\boldsymbol{t}$. It is used to decide if a position candidate of set C_{H2} is a true

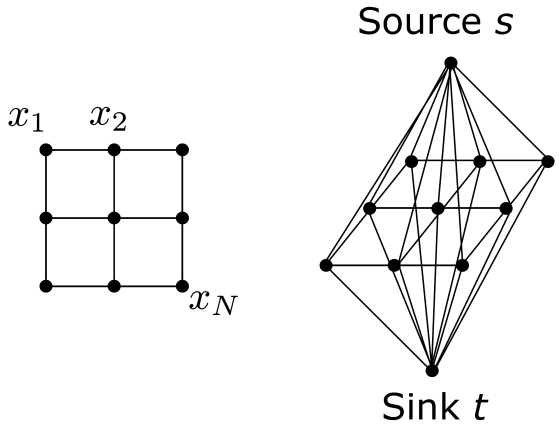


Figure 6.8: Illustration of the graph of a 2-D 3×3 image.

positive or a false positive. The set of candidates with the best classification score is kept and denoted with C_A.

6.6.2 Segmenting node-like structures using graph cuts

In this section, we adapt the graph cuts segmentation method to the problem of lymph nodes segmentation. Graph cuts can be used as an all-purpose segmentation method, but the performance can be considerably improved by using problem-specific seeds and weights.

We adapt graph cuts segmentation in two ways. First, we select the edge weights of the graph according to boundary and object probabilities that are obtained from intensity and joint intensity histograms extracted from manually segmented data.

Next, we propose a sphere shape prior that is well suited to segment blob-like nodal structures. For approximately spherical objects, this prior solves at the same time a major problem of graph cuts, which is the "small cut" behavior: When only few seeds are given, the cheapest cut is often the one directly around the seeds.

At this point, we already have the center $\boldsymbol{t} \in C_A$ of a detected lymph node candidate from our previous detection steps. We consider a sub-image cropped from the original volume image such that $\boldsymbol{t}$ is centered in the sub image. The size of the sub-image remains fixed at $4 \times 4 \times 4$ cm^3. This is relatively large and ensures that almost all lymph nodes fit into this window.

This sub-image is now converted into a graph representation. Each voxel is a node of the graph, and neighboring voxels are connected based on a neighborhood criterion. In this work, this is either a 6-neighborhood or a 26-neighborhood. A larger neighborhood leads to a more complex graph and is computationally more expensive, but often leads to a more accurate segmentation [Boyk 03]. There are furthermore two special nodes, the source and the sink, that are directly connected to multiple voxel nodes. This graph topology is also called s-t graph in the literature. Such a graph is illustrated in Figure 6.8 for a 2-D image and a 4-neighborhood.

Each edge is associated with a capacity c_{ij}. The capacity between voxels i and j is $c_{ij} = \beta_{ij}$ if they are neighbored. The so-called unary capacities of the edges from the source or to the sink are denoted by λ_i. The capacity c_{si} from the source s to voxel i is $c_{si} = \lambda_i$ if $\lambda_i > 0$, and $c_{si} = 0$ otherwise. Likewise, the capacity c_{it} from voxel i to the

sink t is $c_{it} = -\lambda_i$ if $\lambda_i < 0$, and $c_{it} = 0$ otherwise. Let N be the number of voxels in the sub-image. Each voxel is associated with a label $x_i \in \{0, 1\}$ that is either one for "foreground" or zero for "background". The labels $\boldsymbol{x} = (x_1 \dots x_N)$ of all voxels define a cut of the graph into two subgraphs with vertices

$$\mathcal{B} = \{s\} \cup \{i | x_i = 1\} \tag{6.20}$$

$$\mathcal{W} = \{t\} \cup \{i | x_i = 0\}. \tag{6.21}$$

The set $\mathcal{B}$ contains the source s and all voxels labeled with one, and $\mathcal{W}$ contains the sink and all voxels labeled with zero. Such a cut is associated with a cost

$$\mathcal{C}(\boldsymbol{x}) = \sum_{k \in \mathcal{B}} \sum_{l \in \mathcal{W}} c_{kl} \tag{6.22}$$

that equals the summed capacities of all cut edges. With the edge capacities defined above, this cost is

$$\mathcal{C}(\boldsymbol{x}) = \sum_{i=1}^{N} x_i \max(0, -\lambda_i) + \sum_{i=1}^{N} (1 - x_i) \max(0, \lambda) + \sum_{i=1}^{N} \sum_{j=1}^{N} \beta_{ij} x_i (1 - x_j). \tag{6.23}$$

The first two sums in (6.23) are the cost of cutting the edges from the source or to the sink, and the last sum is the cost of cutting the edges between voxels. The cost $\mathcal{C}(\boldsymbol{x})$ can be minimized efficiently using min-cut/max flow algorithms, such as the algorithm proposed in [Boyk 04], which was used in this work.

As shown in [Grei 89], in case of undirected edge weights $\beta_{ij} = \beta_{ji}$, minimizing the cost $\mathcal{C}(\boldsymbol{x})$ and maximizing

$$\underset{\boldsymbol{x}}{\operatorname{argmin}}\ \mathcal{C}(\boldsymbol{x}) = \underset{\boldsymbol{x}}{\operatorname{argmax}} \sum_i \lambda_i x_i + \frac{1}{2} \sum_{ij} \beta_{ij} \delta(x_i, x_j) \tag{6.24}$$

the right hand side of (6.24) with

$$\delta(a, b) = \begin{cases} 1 & : \quad a = b \\ 0 & : \quad \text{otherwise} \end{cases} \tag{6.25}$$

yields the same solution $\boldsymbol{x}$. In [Grei 89], the sum $\sum_i \lambda_i x_i$ was interpreted as a log likelihood function of the image intensities given the labels, and $\frac{1}{2} \sum_{ij} \beta_{ij} \delta(x_i, x_j)$ was interpreted as a prior of the labels. Then, optimizing (6.24) is a maximum a posteriori estimation of $\boldsymbol{x}$.

Here, the edges between neighboring voxels are in general directed, and the function that is maximized therefore slightly differs from (6.24). The cost (6.23) can be rewritten as

$$\mathcal{C}(\boldsymbol{x}) = \sum_{i=1}^{N} \max(0, \lambda_i) + \sum_{i=1}^{N} x_i \left[\max(0, -\lambda_i) - \max(0, \lambda_i)\right] + \sum_{i=1}^{N} \sum_{j=1}^{N} \beta_{ij} x_i (1 - x_j). \tag{6.26}$$

Because $\sum_{i=1}^{N} \max(0, \lambda_i)$ does not depend on $\boldsymbol{x}$, and since

$$\max(0, -\lambda_i) - \max(0, \lambda_i) = -\lambda_i, \tag{6.27}$$

minimizing (6.26) is equivalent to maximizing

$$\underset{\boldsymbol{x}}{\operatorname{argmin}}\, \mathcal{C}(\boldsymbol{x}) = \underset{\boldsymbol{x}}{\operatorname{argmax}} \sum_{i=1}^{N} x_i \lambda_i - \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_{ij} x_i (1 - x_j). \tag{6.28}$$

As in [Grei 89], the sum $\sum_{i=1}^{N} x_i \lambda_i$ is interpreted as a log likelihood of conditionally independent pixel intensities

$$\ln p(I_1 \dots I_N | \boldsymbol{x}) = \ln \prod_{i=1}^{N} p(I_i | x_i) \tag{6.29}$$

that can be reformulated to

$$\ln p(I_1 \dots I_N | \boldsymbol{x}) = \ln \prod_{i=1}^{N} p(I_i | x_i = 1)^{x_i} p(I_i | x_i = 0)^{1 - x_i} \tag{6.30}$$

$$= \sum_{i=1}^{N} x_i \ln \frac{p(I_i | x_i = 1)}{p(I_i | x_i = 0)} + \sum_{i=1}^{N} \ln p(I_i | x_i = 0). \tag{6.31}$$

If the unary capacities λ_i are set to $\ln p(I_i | x_i = 1) / p(I_i | x_i = 0)$, and if the prior $p(\boldsymbol{x})$ of the labels $\boldsymbol{x}$ is proportional to

$$p(\boldsymbol{x}) \propto \exp\left(- \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_{ij} x_i (1 - x_j) \right), \tag{6.32}$$

then (6.28) computes the MAP estimate of $\boldsymbol{x}$.

Here, the unary capacity λ_i is set to

$$\lambda_i = \ln \frac{p^u(x_i = 1 | I_i)}{1 - p^u(x_i = 1 | I_i)}, \tag{6.33}$$

which, except for u, equals $\ln p(I_i | x_i = 1) / p(I_i | x_i = 0)$ in case of equal priors $p(x_i = 0) = p(x_i = 1)$. As the segmentation is initialized with detected lymph node candidates, equal priors are a reasonnable assumption. The constant u is used to balance between the influence of the unary and the binary capacities. It is set to $u = 0.13$ in the experiments. The probability $p(x_i = 1 | I_i)$ is estimated non-parametrically using a histogram.

A high β_{ij} value reflects that voxels i and j are likely to have the same label. A high λ_i value means that, without knowing anything about its neighborhood, voxel i is more likely to be foreground.

Since the center $\boldsymbol{t}$ of the sub-image is assumed to be the center of the lymph node, it is used as positive seed and its $\lambda_{i_{\boldsymbol{t}}}$ value is set to ∞. The boundary voxels of the sub-image are marked as negative seeds and their unary capacities are set to $-\infty$.

If all other unary capacities λ_i were set to zero, and all binary capacities β_{ij} to some positive constant, then the cost of a cut would be proportional to its surface, and the smallest cut that separates the source from the sink would simply separate the positive seed from its direct neighbors (see Figure 6.9 left). This is also known as the small cut problem of graph cuts [Sino 07]. In this special setting, the problem can be solved by simply adding a factor

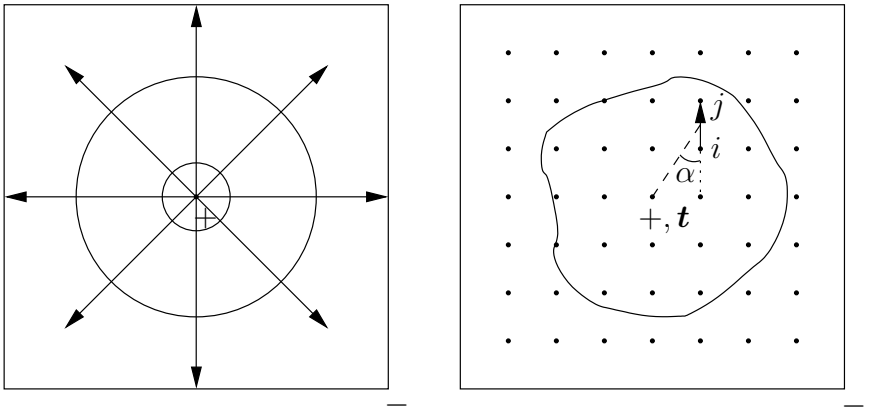


Figure 6.9: Left: The total flow through surfaces that separate the positive seed $(+)$ from the negative seeds $(-)$, for instance concentric circles (spheres) centered at the positive seed, is constant. If the unary edge weights λ_i are nonzero only at the seeds, and if β_{ij} is constant for all edges between neighboring voxels i, j, then the cost of a cut is proportional to its length (surface). Right: Illustration of $p(\text{out}_{ij})$. If the edge from voxel i to voxel j points away from the center $\boldsymbol{t}$ (i. e. $\cos\alpha \approx 1$), then it is likely to point in outward direction.

$\frac{1}{r_{ij}^2}$ to the capacities β_{ij}, were r_{ij} denotes the distance of the center point of the edge from voxel i to voxel j to the positive seed at $\boldsymbol{t}$. If the original capacities β'_{ij} are constant and $\beta_{ij} = \frac{1}{r_{ij}^2}\beta'_{ij}$, then the integrated capacity $K(r)$

$$K(r) = \sum_{(i,j)\in\mathcal{K}(r)} \beta_{ij} \tag{6.34}$$

over a sphere centered at $\boldsymbol{t}$ is nearly constant for different radii r (it is not exactly constant because of discrete voxels). In (6.34), $\mathcal{K}(r)$ denotes the set of edges intersected by the sphere centered at $\boldsymbol{t}$ with radius r. Now there is no bias any more toward a small cut. Because of the smaller surface, spherical cuts are preferred over non-spherical cuts, which is a desirable property for the purpose of segmenting a node-like structure. This method is not only simple, is also comes at no additional computational costs.

Other shape priors have been proposed for graph cuts segmentation. In [Slab 05], a prior for elliptic shapes was introduced. However, the segmentation must be solved iteratively. In [Funk 06], a method that favors cuts that are orthogonal to the line from the current point to the center was proposed. This is effectively a prior for blob-like structures but does not solve the small cut problem. A prior for star-shaped structures and also a balloon force that corresponds to a certain boundary length was introduced in [Veks 08]. This solves the small cut problem, but the balloon force is optimized iteratively. In [Das 09], the same balloon force as in [Veks 08] is used together with a prior for compact shapes, but there is no obvious extension of the shape prior to 3-D. [Wang 01] proposed to normalize the cost of a cut by its boundary length. This solves the bias toward a small cut. However, it also removes the bias toward a smooth surface and is therefore more prone to producing leaking segmentations. Furthermore, the minimum cut / maximum flow based global optimization technique of graph cuts is not applicable any more with this cost function and an iterative optimization scheme is required.

Motivated by (6.32), β_{ij} is based on the logarithm of the probability of observing the object boundary between the voxels i and j. The range of the attenuation coefficients of lymph nodes is restricted. This allows to estimate these probabilities according to intensity histograms. Furthermore, β_{ij} should be set according to the distance of the edge from voxel i to voxel j to the center $\boldsymbol{t}$ as pointed out above, and also according to the orientation of the edge, the intensities at i and j and common intensity jumps at the border of lymph nodes in order to use all available information. Here, it is set to

$$\beta_{ij} = -\frac{1}{r_{ij}^2} \cdot \frac{1}{d_{ij}} \ln \left[p(\text{out}_{ij}) p(x_i = 1, x_j = 0 | I_i, I_j) \right], \tag{6.35}$$

where

$$p(\text{out}_{ij}) = \frac{\cos \alpha_{ij} + 1}{2} \tag{6.36}$$

is the estimated probability that the edge from voxel i to voxel j is pointing in outward direction. In (6.36), α_{ij} is the angle between the edge from i to j and the line from the positive seed to the center of the edge. Thus, $\cos \alpha_{ij} = 1$ if the edge is pointing away from the central seed, and $\cos \alpha_{ij} = -1$ if it is pointing toward the center. See Figure 6.9 (right) for an illustration. The term $p(x_i = 1, x_j = 0 | I_i, I_j)$ denotes the probability of observing the object boundary between the adjacent voxels i and j given the intensity I_i of the voxel that is assumed to be inside and I_j of the voxel that it assumed to be outside the segmentation. The term d_{ij} in (6.35) denotes the Euclidean distance of the voxels i and j. It is relevant when a neighborhood system is used that does not only include the six direct neighbors.

Using directed capacities β_{ij} allows to incorporate additional knowledge about the object boundary. If the edge from voxel i to j is pointing in inward direction, then $p(\text{out}_{ij}) = 0$ and therefore $\beta_{ij} = \infty$. It models that the interior of the lymph node is expected to be closer to the center than the exterior of the node. If both $p(\text{out}_{ij}) = 1$ and $p(x_i = 1, x_j = 0 | I_i, I_j) = 1$, then $\beta_{ij} = 0$, meaning that cutting this edge comes at no costs.

In (6.35), $p(x_i = 1, x_j = 0 | I_i, I_j)$ can be expressed as

$$p(x_i = 1, x_j = 0 | I_i, I_j) = \frac{p(x_i = 1, x_j = 0, I_i, I_j)}{p(I_i, I_j)}. \tag{6.37}$$

Both $p(x_i = 1, x_j = 0, I_i, I_j)$ and $p(I_i, I_j)$ are estimated non-parametrically using joint intensity histograms. $p(I_i, I_j)$ is set to the number of neighboring voxels with intensities I_i and I_j divided by the number of neighboring voxels. This histogram is denseley populated due to the huge number of training samples. $p(x_i = 1, x_j = 0, I_i, I_j)$ is computed by first counting the number of neighboring voxels with the properties that voxel i is inside a lymph node and has an intensity of I_i, and voxel j is outside any lymph node and has an intensity of I_j, and then dividing this number by the number of neighboring voxels. However, $p(x_i = 1, x_j = 0, I_i, I_j)$ is sparse because of a limited number of training examples of points on the boundary of lymph nodes. Therefore, $p(x_i = 1, x_j = 0 | I_i, I_j)$ is smoothed with a Gaussian filter with $\sigma = 40$ HU, which is effectively a Parzen estimation. Figure 6.10 shows the estimated probability $p(x_i = 1, x_j = 0 | I_i, I_j)$. All histograms have 400 equally spaced bins in each dimension, where the lowest bin corresponds to -1024 HU and each bin is 4 HU wide.

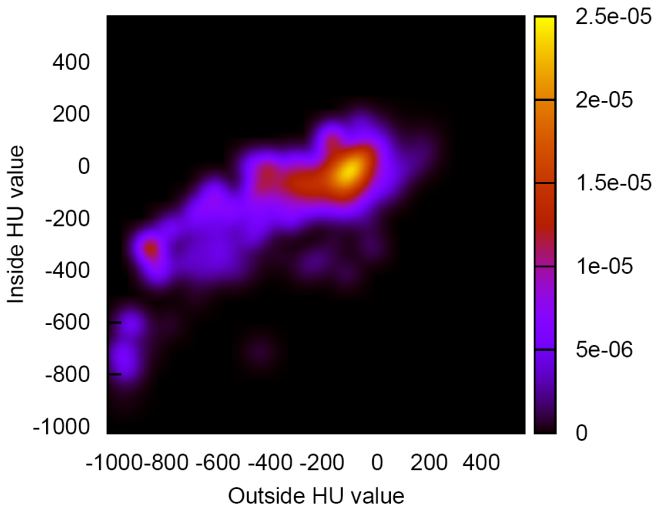


Figure 6.10: Estimate of the probability $p(x_i = 1, x_j = 0|I_i, I_j)$. It is asymmetric because the interior of a lymph node has usually a higher attenuation coefficient than its surroundings.

6.6.3 Segmentation based features

We now have the candidate segmentation that was initialized with the detected lymph node center $\boldsymbol{t}$. As final stage in the detection cascade, an AdaBoost classifier is trained with features extracted from the segmentation to learn whether $\boldsymbol{t}$ is a true lymph node or a false detection.

The first kind of features is histogram based: Given a binary segmentation mask image, a hierarchy of normalized histograms of the intensity values inside the segmentation is computed. The histogram at the first level has 256 bins. Each bin is one Hounsfield unit wide, and the first bin corresponds to -128 HU. Lymph nodes typically fall into this range of HU values. At the next level, the number of bins is halved, and the width of each bin is doubled, which is why the number of bins in the original level is a power of two. In total, seven levels are used. The frequency of each bin of each pyramid level is a scalar feature.

The second kind of features are again based on a hierarchy of histograms, but the histograms are now computed from the 3 mm wide neighborhood of the segmentation. The neighborhood is determined using morphological operations. Additionally, we use the second, third and fourth central moments of the histograms both inside and outside the segmentation.

Next, 100 points are randomly sampled from the surface of the segmentation. As proposed in [Barb 10], the gradient is computed at each point, and the points are sorted by their gradient magnitude. The sorting is necessary to enumerate the points. At each point, the normal to the surface is computed, and the normal is sampled at seven positions with a spacing of 1 mm between the samples. At each sample, steerable features are computed. All scalar features at all samples at all normals at all points are added to the feature pool.

Furthermore, features are used that capture the relative position of the lymph node center $\boldsymbol{t}$ within the tight axes aligned bounding box of the segmentation. A relative position $\boldsymbol{t}'$ of $\boldsymbol{t}$ inside this box is computed that is normalized to lie in $[-0.5, 0.5]$ for each dimension. A value of 0 indicates that $\boldsymbol{t}'$ is centered, and values of -0.5 and 0.5 indicate that $\boldsymbol{t}'$ lies

on the bounding box wall in this dimension. The minimum relative distance to any wall of the box, the difference of the maximum and the minimum distance to any wall, and the relative distance averaged over the three dimensions are used as features.

Finally, the volume, the surface, the sphericity, the maximum flow value and the maximum flow divided by the surface are used. In total, the feature pool contains 51436 features. The vector containing these features extracted at position $\boldsymbol{t}$ is denoted with $\boldsymbol{D}(\boldsymbol{t})$.

As in section 6.6.1, an AdaBoost classifier is used to select a small subset of the feature pool and is trained to learn the probability $p(m = 1|\boldsymbol{D}(\boldsymbol{t}))$ of whether $\boldsymbol{t}$ is a true lymph node and not a false positive given the segmentation based features. Note that, apart from the maximum flow features, the described feature set does not depend on the segmentation method and can therefore also be used in combination with other segmentation techniques, as described in the following section.

6.6.4 Alternative segmentation methods

In section 6.6.2, a graph cuts based segmentation method was presented that was specially designed for the problem of lymph nodes segmentation.

It is however interesting to see how the detection performance is affected if this graph cuts based segmentation method is replaced with a simpler segmentation method. We therefore also use both graph cuts with standard weights and a watershed based segmentation method as a baseline. Features extracted from these segmentations are used in the same way as described in section 6.6.3 to train a classifier to distinguish true positives and false positives.

Graph cuts with standard weights

It is popular in the literature [Boyk 06] to use graph cuts segmentation with unary weights

$$\lambda_i = 0 \tag{6.38}$$

set to zero for all voxels i except for the seeds. This means that no prior knowledge about the foreground or background intensities is available and is therefore not specific to a certain problem. The binary edge weight β_{ij} from voxel i to voxel j is commonly set to

$$\beta_{ij} = \exp\left(\frac{-(I_i - I_j)^2}{2\sigma_\beta^2}\right), \tag{6.39}$$

where σ_β is a constant that is typically set according to the noisiness of the input data. These weights are symmetric and simply mean that the object boundary is probably at image intensity jumps. The effect of a low value of σ_β is that the edge capacities β_{ij} quickly approach zero already at moderate intensity jumps. The cut is then more susceptible to noise, and also numeric problems can arise. If on the other hand σ_β is set to a high value, cuts at intensity jumps are more likely, and the surface of the cut becomes more important. As a result, the method is more prone to the small cut problem. Unless mentioned otherwise, we use a value of $\sigma_\beta = 16$ HU, which is a good compromise.

Hierarchical watershed segmentation

The watershed transformation [Beuc 92] is another popular low level method for image segmentation.

In order to enhance relevant edges, the input image is windowed with a soft tissue window (center: 16 HU, width: 400 HU). An edge image E is generated by computing the gradient magnitude of the windowed image I_W. To reduce the susceptibility to noise, the gradients are computed by convolving the image with the first derivatives of a 3-D Gaussian g with a standard deviation of 2 mm in each direction:

$$E(\boldsymbol{t}) = \|\nabla g(\boldsymbol{t}) * I_W(\boldsymbol{t})\|_2 \text{ with } \nabla g(\boldsymbol{t}) \in \mathbb{R}^3. \tag{6.40}$$

The convolution in (6.40) is carried out component-wise.

In this work, we use an hierarchical segmentation method [Beuc 94]. In the first step, it generates a very oversegmented mosaic image using the standard watershed transform of the edge image E. In order to reduce the number of watershed regions that are generated in this first step, E is thresholded: Voxels below a threshold θ_{WS} are set to θ_{WS}. Local minima of E serve as seeds in the watershed transform, and the effect of the thresholding is that seeds are merged if they are close to each other and have a low absolute value. The threshold θ_{WS} is set to $0.01 e_{\text{max}}$, where e_{max} is the maximum value that occurs in the edge image E.

In the next step, segmentations with different "flooding levels" l_{WS} are generated by merging neighboring watershed regions. The higher l_{WS} is, the more regions are merged, and the less oversegmented is the resulting segmentation. l_{WS} is a relative value in the range of $[0, 1]$. A value of zero means that no regions are merged, and one means that all regions are merged into a single one. It corresponds to an absolute flooding level L_{WS} with

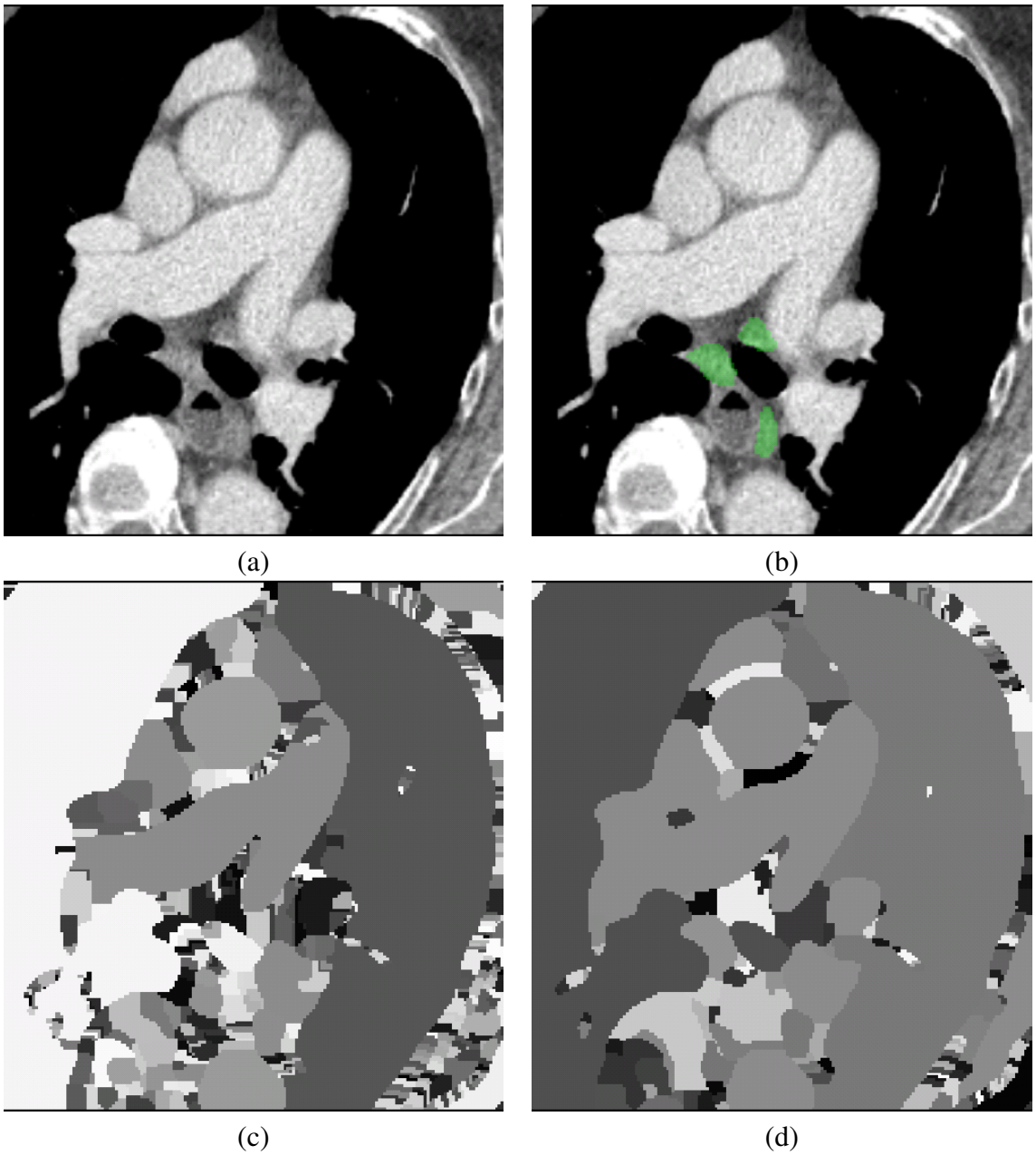
$$L_{\text{WS}} = l_{\text{WS}} e_{\text{max}}. \tag{6.41}$$

Region A floods region B of the edge image (i. e. A and B are merged) at level L_{WS} if the relative depth of region A is lower than L_{WS} and region B is across the lowest part of the boundary of region A. The relative depth of a region in the edge image is here the lowest value on the region boundary minus the lowest value in the region. In this work, we use two levels with $l_{\text{WS}} = 0.1$ and $l_{\text{WS}} = 0.2$. Example segmentations can be seen in Figure 6.11.

The segmentation based features $\boldsymbol{D}(\boldsymbol{t})$ are now extracted from the region that contains the center $\boldsymbol{t}$ of the detected lymph node that is to be verified. Note that, in contrast to the graph cuts method, neither the segmentation nor the segmentation based features as described in section 6.6.3 depend on the location of $\boldsymbol{t}$ within the region, apart from the three features that capture the relative position of $\boldsymbol{t}$ inside the bounding box of the segmentation. Therefore, the segmentation of the whole image region of interest is precomputed. Furthermore, the computed features of a region are cached. Thus, they do not need to be recomputed if another lymph node is verified whose center falls into the same region.

6.6.5 Combining clues from alternative segmentations

Instead of only using different segmentation methods alternatively, we also explored if the performance of the system can be improved by combining different segmentations. Even though our proposed segmentation method described in section 6.6.2 is already tuned to



(a) (b)

(c) (d)

Figure 6.11: Example of a slice of a 3-D hierarchical watershed segmentation. (a): An axial slice of a CT volume. (b): Manual ground truth segmentations of lymph nodes (green). (c-d): Slices of two 3-D watershed segmentations with flooding levels l_{WS} of 0.1 (c) and 0.2 (d). The image intensity of each region is randomly chosen.

the problem of lymph nodes segmentation, it is possible that segmentations generated with simpler methods still contain valuable information that helps to distinguish true lymph nodes from false alarms.

Here, different segmentation methods are initialized with the center of a lymph node detection. The features described in section 6.6.3 are extracted from all segmentations and added to a common feature pool. For instance, if the proposed graph cuts segmentation method is combined with two hierarchy levels of the hierarchical watershed segmentation, three alternative segmentations are generated, and the number of features will be three times larger compared to when only a single segmentation is used. It can also be viewed as treating the type of the segmentation method as a feature.

If N_S different segmentation methods are combined, the new feature vector

$$\boldsymbol{D}_{\text{combined}}(\boldsymbol{t}) = (\boldsymbol{D}_1(\boldsymbol{t}), \ldots, \boldsymbol{D}_{N_S}(\boldsymbol{t})) \tag{6.42}$$

is simply the concatenation of the vectors extracted form the single segmentations. The classifier is then trained on the joint feature pool to learn

$$p(m = 1|\boldsymbol{D}_{\text{combined}}(\boldsymbol{t})) \tag{6.43}$$

the probability of whether there is a true lymph node given the joint features of the segmentations initialized with the presumed lymph node center $\boldsymbol{t}$.

6.6.6 Integrating the prior

Section 6.5.2 already explained how the score of the detectors in branch A of the detection pipeline is combined with the spatial prior probability $p(m = 1|\boldsymbol{t})$ of observing lymphatic tissue at a certain location $\boldsymbol{t}$. The prior can be integrated into branch B of the pipeline in a similar way.

In contrast to method A where the scale was detected directly, now our object parameters are simply the center $\boldsymbol{t}$ of a lymph node, and the search space of all detectors in the detection pipeline is $\boldsymbol{t}$. As the scores of the different translation detectors are very dependent, and the scores of the first detectors are implicitly contained in the scores of the later ones because weak candidates are rejected at early and intermediate levels, we only use the score of the last detector to compute the final score. This final score is then the probability

$$p(m = 1|\boldsymbol{D}, \boldsymbol{t}) \tag{6.44}$$

of observing a lymph node given the segmentation based features $\boldsymbol{D}$ and the position $\boldsymbol{t}$. Similar as in section 6.5.2, we make the simplifying assumption that the segmentation based features $\boldsymbol{D}$ are statistically independent from the position $\boldsymbol{t}$. Under this assumption, (6.44) can be rewritten as in (6.14) and is

$$p(m = 1|\boldsymbol{D}, \boldsymbol{t}) = \frac{p(m = 1|\boldsymbol{D})p(m = 1|\boldsymbol{t})}{p(m = 1)} \tag{6.45}$$

proportional to the product of the spatial prior and the segmentation feature based detection score $p(m = 1|\boldsymbol{D})$.

6.7 Results

The proposed methods have been evaluated on 54 CT datasets showing the chest area. All scans were taken from patients suffering from lymphoma. The slice spacing was 1 mm, and the intra-slice resolution was typically in the range between 0.7 mm and 0.9 mm. The images were reconstructed using a soft-tissue kernel.

All datasets were resampled to an isotropic $1 \times 1 \times 1\text{mm}^3$ resolution. The mediastinal lymph nodes were manually segmented by a medical student and the author, and the segmentations were reviewed by a radiologist.

The detection performance was evaluated using threefold cross-validation. For each fold, the spatial prior, the classifiers and the graph cuts weights for the segmentation were trained on the training data and evaluated on the test data. The classifiers were only trained on lymph nodes that have a minimum size of 10 mm in at least two dimensions. Smaller lymph nodes are usually not pathologic [Lang 06] and were therefore neglected. The set of manual segmentations contained six huge cases with a size exceeding 5 cm. These were mostly not single nodes but a cluster of lymph nodes that were densely packed so that the boundaries became invisible in the acquired CT scan. Such cases were removed from the training set in order not to distract the detector with few extreme examples. Among the segmented lymph nodes, 289 were used for training. In order to achieve a better generalization and to avoid overfitting, the training data was mirrored by all three coordinate planes, resulting in $2^3 = 8$ times more training examples. For testing, only the original data was used.

In the testing phase, a lymph node is considered as detected if the center $\boldsymbol{t}$ of a detection is inside the tight axis-aligned bounding box of the lymph node. This criterion for a true positive detection is referred to as "in box". A lymph node is considered as a false negative (FN) if its size is at least 10 mm and it is not detected.

Occasionally, two or more detections are close together. In order to reduce the number of such double detections, the detected centers are spatially clustered and merged. Two detections are merged if their distance is below a distance threshold θ_d. The confidence value of the merged detection is set to the sum of the original ones. In method A, the threshold θ_d was set to 12 mm. Method B performed better with a lower threshold of 6 mm.

The way positive and negative training samples are generated considerably affects the detection performance. The problem with positive training examples is that the manual lymph node segmentations are often not convex. The main reason is that it is often not decidable where one lymph node ends and another one begins because there is no visible boundary. The straightforward approach would be to take the point of gravity or the center of the bounding box as positive example. However, this point is often close to the lymph node boundary or even outside the actual node. As a solution, a depth map is computed for each ground truth lymph node. The map contains the shortest distance to the surface for each voxel. Local maxima of the depth map that have a minimum distance of 2 mm from the surface are selected as positive training samples.

The negative training samples of the first stage are generated by randomly sampling the training images, but no candidates are generated inside ground truth lymph nodes and in regions where the spatial prior has a value of zero because these regions are not considered during test. This avoids confusing the detector with data it will never see in the testing

Classifier	features	tree levels	weak class.	candidates
Stage 1	Haar	2	20	not fixed
Stage 2	Haar	2	20	2000
Stage 3 A	Steerable	2	20	400
Stage 3 B	Gradient aligned	1	270	200
Stage 4 B	Segmentation based	1	270	100

Table 6.1: Features and parameter settings of the classifiers of the detection pipeline: The number of levels in the PBT classifier, the number of weak classifier per AdaBoost node, and the number of detection candidates generated at each stage. In stage three and four of method B, only a single AdaBoost classifier is used.

phase. In later stages, the negative training examples always come from the false positive detections of the previous stage. Thus, the classifiers get specialized on the difficult examples.

Table 6.1 lists the parameter settings and the feature types of the classifiers in the detection pipeline that were used in the experiments, unless otherwise mentioned.

Comparing method A and method B. The detection performance of method B is visualized in Figure 6.12 (a) as a free-response receiver operating characteristic (FROC) curve (green). Segmentation based features were extracted from a graph cuts segmentation with edge weights adapted to lymph nodes as described in section 6.6.2. Each voxel in the graph was directly connected to six neighbors (6 nb.). This was compared to method A, where the last two stages of the detection cascade, the gradient-aligned and segmentation based detectors, are replaced with a scale detector that used steerable features (red curve). When a fixed amount of false alarms is allowed, method B reaches a higher recall. For four false positives per volume, the detection rate of method B is 0.57 and by 33% better than the detection rate of 0.43 of method A. For seven false positives per volume, the detection rate of method B is 0.61 and by 20% better than method A.

Influence of the graph neighborhood size. Next, we examined how the size of the graph neighborhood in the segmentation step affects the detection performance (Figure 6.12 (b)). The two curves show the detection performance with a 6-neighborhood and a 26-neighborhood. The performance is very similar, indicating that the neighborhood size does not significantly affect the detection rate. However, we noticed that a larger neighborhood leads to smoother segmentations.

Influence of the number and the type of alternative segmentations. Figure 6.12 (c) shows how the segmentation method used in stage four of the cascade affects the detection performance. The performance of our proposed graph cuts segmentation method is shown in red. The blue curve shows the performance of a graph cuts segmentation with standard weights as described in section 6.6.4. Apart from evaluating alternative segmentation methods, we also did experiments with combined segmentations (see section 6.6.5). First, we use two watershed segmentations with different flooding level thresholds $l_{\mathrm{WS}} = 0.1$ and $l_{\mathrm{WS}} = 0.2$ (see section 6.6.4). Given a detected lymph node center candidate $\boldsymbol{t}$, the segmentation based features are extracted two times, once for each segmentation. The resulting feature vectors are called $\boldsymbol{D}_{\mathrm{WS}}^{(0.1)}(\boldsymbol{t})$ and $\boldsymbol{D}_{\mathrm{WS}}^{(0.2)}(\boldsymbol{t})$. In both cases, the features are extracted

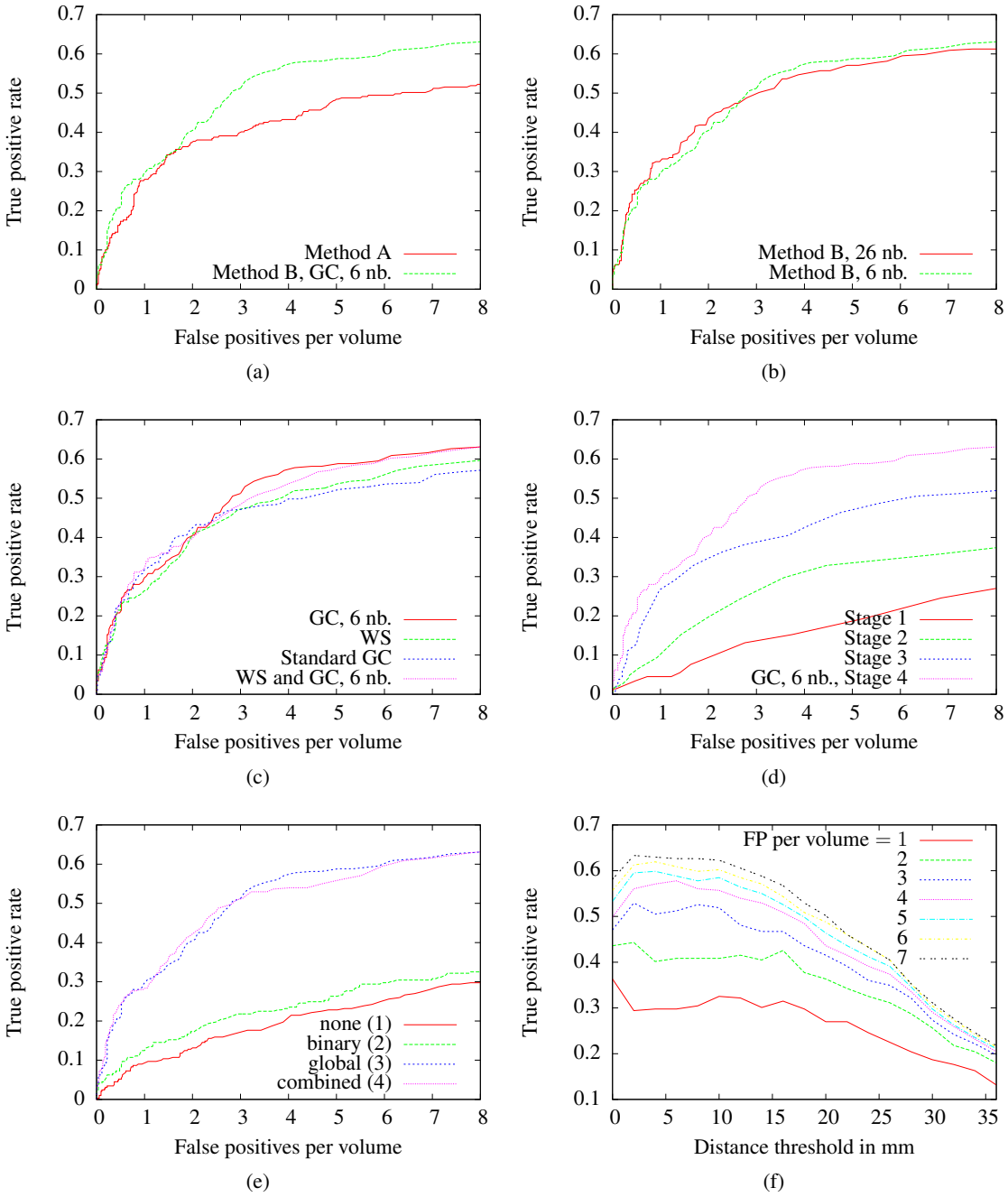


Figure 6.12: Detection performance of different methods and different parameter settings. (a): Comparison of method A with method B with the proposed graph cuts (GC) segmentation method and a 6-neighborhood. (b): Influence of the neighborhood size. (c): Influence of the segmentation method that is either the proposed one (red), watershed (green), graph cuts with standard weights (blue), or both watershed and the proposed graph cuts method (violet). (d): Performance at different stages of the cascade of method B. (e): Effect of using different prior types. (f): Influence of the threshold used for clustering. See text for further details.

from the region that contains $\boldsymbol{t}$. The green curve shows the performance of a detector that is trained on the combined feature pool

$$\boldsymbol{D}_{\mathrm{WS}}(\boldsymbol{t}) = \left(\boldsymbol{D}_{\mathrm{WS}}^{(0.1)}(\boldsymbol{t}), \boldsymbol{D}_{\mathrm{WS}}^{(0.2)}(\boldsymbol{t})\right)^T. \tag{6.46}$$

Next, we combined the features $\boldsymbol{D}_{\mathrm{WS}}(\boldsymbol{t})$ of the two watershed segmentations with features $\boldsymbol{D}_{\mathrm{GC}}(\boldsymbol{t})$ extracted from our proposed graph cuts segmentation into a feature vector

$$\boldsymbol{D}_{\mathrm{WS,GC}}(\boldsymbol{t}) = \left(\boldsymbol{D}_{\mathrm{WS}}(\boldsymbol{t}), \boldsymbol{D}_{\mathrm{GC}}(\boldsymbol{t})\right)^T. \tag{6.47}$$

The resulting detection performance is shown as pink curve. Results indicate that the detection performs better if the features are extracted from the proposed graph cuts segmentation instead of a standard graph cuts segmentation or a hierarchical watershed segmentation. But surprisingly, the detection performance does not further increase if both the graph cut and the hierarchical watershed segmentation are taken into account, indicating that the watershed segmentation does not contain a significant amount of additional information if the graph cuts segmentation is known.

Performance at different pipeline stages. Figure 6.12 (d) shows the detection performance at different levels of the cascade (see Figure 6.2). When four false alarms per volume image are allowed, the detection rates at stages one to four are 0.15, 0.31, 0.43 and 0.57. The performance improves considerably from stage to stage. In particular, the final segmentation based verification step clearly improves the detection performance.

Influence of the spatial prior. In order to examine how the type of the spatial prior affects the overall detection performance, the system was evaluated with the four different variants explained in section 6.2. The results are shown in Figure 6.12 (e). If no spatial prior is used (red curve), there is a large amount of false alarms among the detections, leading to a poor performance. Using a binary spatial prior (see eq. (6.2)) that excludes organs and regions filled with air from search slightly improves the detection performance (green curve). Using the "global" prior (see eq. (6.4)) that is a product of the binary prior and a probabilistic atlas greatly reduces the amount of false alarms (blue curve). When compared to using no spatial prior, the detection rate rises from 0.21 to 0.57 at four false alarms per volume, which corresponds to an increase of 171%. This demonstrates the importance of including prior anatomical knowledge in order to solve this challenging detection problem. When the more complicated "combined" prior is used (violet curve) that is a combination of multiple probabilistic atlases (see eq. (6.8)), the detection performance does not further increase. We therefore propose using the simpler "global" prior in order not to introduce unnecessary complexity.

Effect of clustering. In Figure 6.12 (f), the detection performance of our proposed method is shown for different values of the parameter θ_d that is the distance threshold up to which close detection candidates are merged. Each curve in Figure 6.12 (f) corresponds to a different number of allowed false alarms and therefore to a certain point on the FROC curve. It can be seen that when three or more false alarms are allowed, a moderate value of θ_d in the range of 2 mm to 10 mm leads to a better detection performance. The performance degrades for higher values of θ_d because then more and more detections are merged that belong to different lymph nodes.

Table 6.2 lists image databases that were used for evaluation in prior and this work, and Table 6.3 shows a comparison of the detection performances reported in prior work

Database	Body region	num. vol.	size/mm
Kitasaka et al. [Kita 07]	Abdomen	5	> 5.0
Feuerstein et al. [Feue 09]	Mediastinum	5	> 1.5
Dornheim [Dorn 08]	Neck	1	> 8.0
Barbu et al. [Barb 10]	Axillary	101	>10.0
This work	Mediastinum	54	>10.0
Intra-obs. var.	Mediastinum	10	>10.0

Table 6.2: Databases used for evaluation in prior and this work along with minimum size of a false negative lymph node.

Method	TP crit.	TP	FP	FN	TPR	FP per vol.
Kitasaka et al. [Kita 07]	overlap	126	290	95	57.0%	58
Feuerstein et al. [Feue 09]	overlap	87	567	19	82.1%	113
Dornheim [Dorn 08]	unknown	29	9	0	100%	9
Barbu et al. [Barb 10]	in box	298	101	64	82.3%	1.0
This method	in box	153	167	136	52.9%	3.1
This method	in box	176	332	113	60.9%	6.1
Intra-obs. var.	in box	23	8	19	54.8%	0.8

Table 6.3: Detection results compared to state of the art methods. The second column lists the criterion for a true positive detection. See text for details.

along with the performance of our method. The comparability, however, is limited because of different data, different criterions for a detection, different body regions and different minimum lymph node sizes used for evaluation. In [Kita 07] and [Feue 09], a lymph node is considered as detected if there is overlap between the segmentation and the detection. Here, this criterion is called "overlap". This error measure is however a suboptimal choice because a single huge detection covering the whole volume would result in a true positive rate (TPR) of 100% with zero FP, although the detection is obviously meaningless. Therefore, we measured the performance with the "in box" criterion mentioned earlier in this section (section 6.7). Both [Kita 07] and [Feue 09] report a very high number of false alarms. In [Dorn 08], very good results are reported, but the method was evaluated on a single dataset. In [Barb 10], good results are reported for the axillary region. Lymph nodes in the axillary regions are however easier to detect because they are mostly isolated in fat tissue and less surrounded by clutter as in the mediastinal region.

In order to compare the automatic detection results with the performance of a human, we did an experiment on the intra-human observer variability. Ten of the CT volumes were annotated a second time by the same person a few months later. The first segmentations served as ground truth, and the second ones were considered as detections. TPR and FP were measured in the same way as for the automatic detection. The TPR was 54.8% with 0.8 false positives per volume on average. While 0.8 FP is very low, a TPR of 54.8% shows that finding lymph nodes in CT is quite challenging also for humans. Figure 6.13 shows the first and the second segmentations for one of the ten datasets.

The computational requirements of the proposed methods are shown in Table 6.4. They were measured on a standard dual core PC with 2.67 GHz. Detecting the landmarks takes 7.0 s, and segmenting the heart and the esophagus takes 20.7 s. The proposed "global"

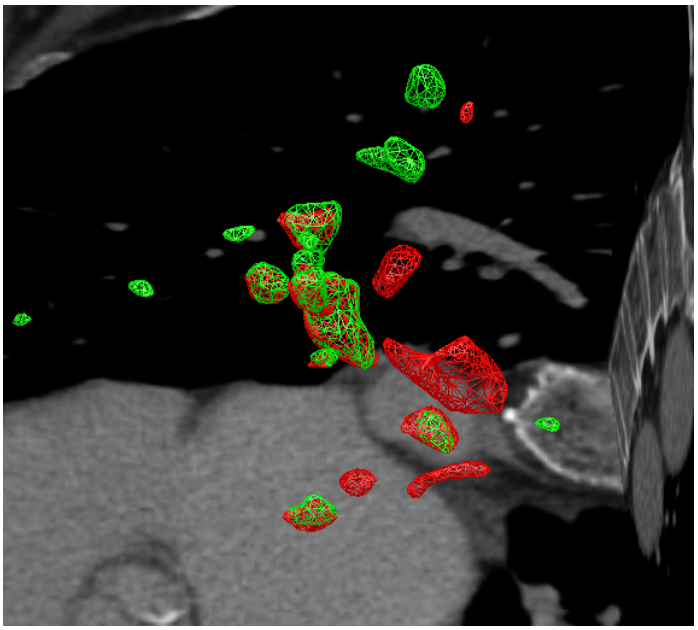


Figure 6.13: Two manual segmentations from the same person. Segmentations of a color were done in one session, and the second session took place several months after the first.

Landmark detection	7.0
organ segmentation	20.7
computing the prior („global“, eq. (6.4) / “combined“, eq. (6.8))	2.9 / 34.4
detection (method A)	19.1
detection and segmentation (method B, 6-neighborhood / 26-neighborhood)	26.0 / 75.2
total (method A / method B, 6-neighborhood / method B, 26-neighborhood)	**56.6 / 105.8 / 49.7**

Table 6.4: Computational requirements of the single steps of the presented methods in seconds

prior (type 3) takes 2.9 s to compute and is computationally far less expensive than computing the more complicated "combined" prior (type 4) that takes 34.4 s. Detecting and segmenting the lymph nodes takes 26.0 s if method B and the proposed graph cuts segmentation with a 6-neighborhood system is used. With a 26-neighborhood, this step takes considerably longer (75.2 s). Detecting the lymph nodes using method A takes 19.1 s. Here, no segmentation is performed. In total, detecting and segmenting the lymph nodes from a CT volume image using method B takes less than a minute (56.6 s) or 1 min 46 s depending on the neighborhood system. Detection using method A is faster (49.7 s) but less accurate, and the lymph nodes are not segmented.

Figure 6.14 shows example detections on unseen data. The second column shows detection results of method B along with the corresponding segmentations that were generated using the proposed graph cuts method and a 26-neighborhood. The small boxes indicate the center of a detected lymph node. Some segmentations do not have a visible detection because it lies in another slice. Bounding boxes detected by method A are shown in the third column for comparison. The manual ground truth segmentations are shown in the fourth column. In rows (a-e), method B properly detected and segmented the lymph nodes (second column). Row (f) shows examples of false positive detections. False positives lie especially on vessels, which can look similar to lymph nodes. Method A detects the clearly visible lymph nodes and some of the less clearly visible ones (fourth column). There are generally more false alarms compared to method B, even though the false alarms mostly have a lower detection score than the true positives.

Figure 6.15 shows 2-D slices of three example segmentations that were manually initialized with the same seed. In subfigure (a) and (b), the segmentation was done using graph cuts and standard weights (see eq. (6.38) and (6.39)). In (a), the parameter σ_β was set to 32 HU, and in (b), it was set to $\sigma_\beta = 16$ HU. A high value of σ_β allows voxel pairs across the object boundary that have a more similar attenuation coefficient. Thus, the surface of the cut becomes more important, and the segmentation is more likely to collapse. If, on the other hand, σ_β is set to a high value, the image is more likely to be cut at locations with a high gradient magnitude. However, the surface of the cut becomes less important, which can cause the segmentation to leak into neighboring structures. Subfigure (c) shows the segmentation result of our proposed segmentation method. The radial weighting prevents the segmentation from collapsing. But rugged segmentations are still penalized because of their larger surface. Thus, blob-like cuts are preferred, and the segmentation is less likely to leak into other structures. The neighborhood size is 26 in all three cases.

6.8 Conclusion

We have presented a method that automatically detects mediastinal lymph nodes in 3-D CT image data, which is a challenging problem due to low contrast to surrounding structures and clutter. We approach the problem from two sides: First, we heavily rely on prior anatomical knowledge, which is modeled as a spatial prior probability and learned from annotated data. A simpler "global" prior that makes use of a single probabilistic atlas is compared to a more complex "combined" prior that is a mixture of local and global atlases and designed to be less susceptible to anatomical variations. Next, this is combined with a discriminative model of the lymph node appearance. A detector is trained that consists of multiple classifiers that are used in a cascade. In the first stages of the cascade, a set of

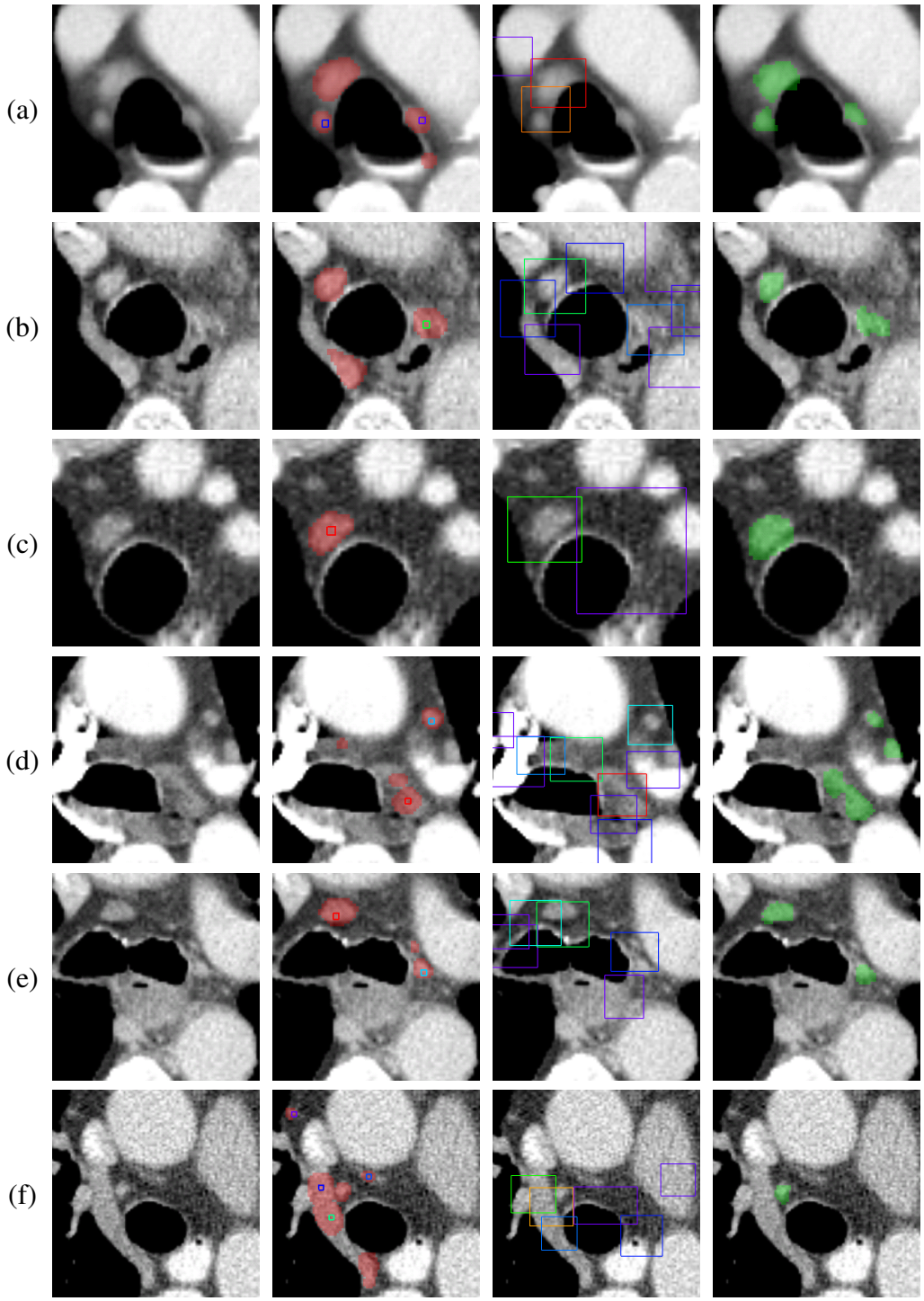


Figure 6.14: Detection and segmentation examples on unseen data shown in 2-D. First column: Plain CT slices. Second column: Detections (small colored boxes) and resulting segmentations (red) of method B. Third column: Bounding boxes detected by method A. In both columns, the detection score is color coded in HSV color space. Violet means lowest, red means highest score. Fourth column: Manual ground truth segmentations (green).

(a) (b) (c)

Figure 6.15: Manually initialized segmentations with different edge capacities. (a): Standard graph cuts weights with $\sigma_\beta = 32$ HU. (b): $\sigma_\beta = 16$ HU. (c): Proposed graph cuts segmentation method.

lymph node center candidates is generated. Two variants for the later stages are compared: In method A, a classifier is trained using regularly sampled point features ("steerable features") to detect the bounding box of a lymph node given its center point. In method B, the detected center point serves as seed for segmenting the lymph node. A feature set is proposed that is extracted from the segmentation. It is used to train a classifier to learn whether the detected lymph node is a true or a false positive. Thus, the segmentation helps to improve the detection performance by rejecting false alarms. We propose to segment the lymph nodes using a graph cuts based semiautomatic segmentation method for blob-like structures that requires a single seed point as input. This is compared to standard graph cuts and also to a watershed segmentation.

Evaluation on 54 datasets showed that the spatial prior greatly improves the detection performance. When a fixed number of false alarms is allowed, the detection rate is well more than doubled when a prior is used. It also turned out that the simpler "global" prior leads to a similar detection performance as the more complicated "combined" prior. We therefore propose using the simpler one that can also be computed about twelve times faster. The experiments further showed that the segmentation based verification step of method B considerably reduces the number of false alarms. Compared to method A, the detection rate of method B is 33% better if four false alarms per volume image are allowed. The best detection performance was achieved using the proposed graph cuts based segmentation method. Interestingly, the detection performance could not further be improved by generating additional watershed candidate segmentations for a detected lymph node center and joining the features from all segmentations. This indicates that the additional segmentations do not contain additional information about whether the detection is a true lymph node or not.

The proposed method B can detect and segment lymph nodes with a TPR of 52.0% at the cost of 3.1 FP per volume image and with a TPR of 60.9% at 6.1 FP per volume within 56.6 s. This TPR is similar to the intra-observer variability of a human that has a TPR of 54.8% with, however, only 0.8 FP per volume.

Chapter 7

Outlook

In this work, we solved various discriminative learning problems with boosting techniques. We either employed AdaBoost directly, or in the form of a probabilistic boosting tree that consists of multiple AdaBoost classifiers. Even though AdaBoost is a very powerful classification technique that is not very prone to overfitting, it is known that overfitting can occur if there are incorrectly labeled samples among the training data, or if the positive and negative samples are very similar in the space of the features used for training [Diet 00, Rtsc 01, Serv 03]. This is especially a problem for lymph node detection because of similar looking other structures. At the later stages of the detection cascade, the positive and the negative training samples are already very similar. Probabilistic boosting trees are more prone to overfitting than simple AdaBoost [Tu 05] and have furthermore the problem that differences in the detection scores often tell little about differences in the quality of the actual detections. It is therefore worth considering to replace some of the classifiers in this work with other classifiers, for instance random forests. Another option is using regularization techniques for AdaBoost as proposed in [Rtsc 01].

We currently rely on supervised learning techniques. Even though they work well, a major disadvantage is that they require a large amount of labeled training data. Dozens, better hundreds of training examples need to be labeled manually to achieve a good detection performance. In future work, it is worth exploring semi-supervised learning techniques [Chap 06, Zhu 09] to reduce the need for a large amount of manual annotations, even though this introduces the risk of generating wrong training examples that can confuse the detector.

Another promising research direction in future work is online learning techniques. Online learning means that a classifier is not trained once and then used for testing without further modifications, but can be modified. New training examples can be inserted at any time. While this is typically used to save training time when new training data becomes available, or to adapt a classifier to a special environment, for instance, a speaker in automatic speech recognition, it can also be used to adapt a detector to a certain person and a particular disease of that person. Currently, a major problem of supervised learning techniques for medical image analysis is the performance on pathologic data. Automatic object detection works well if different instances of the object of interest look similar. In case of pathologies, this is typically not the case. The reason is that there is a very wide spectrum of possible pathologies in the human body. For instance, a healthy esophagus is typically small in diameter, and occasionally appears filled with air and small amounts of contrast

agent. In case of achalasia, the esophagus is however greatly dilated. On CT images, it can appear completely filled with orally given contrast agent. It may also appear differently due to an intervention. A common therapy of esophageal cancer includes resecting parts of the esophagus. The resected tissue is then replaced with intestine tissue or parts of the stomach. The result may appear differently in CT images depending on which portion is resected and how it is replaced. However, parts of the *same* pathologic esophagus often look similar. Once a part of a pathologic esophagus has been detected with a sufficient confidence, the detector can be trained online with this part and look for similar sections in the same volume image. The same technique can also be applied for lesion detection. For instance, while there is a broad spectrum of different liver lesions, the lesions within a certain image often look similar. A detected lesion can be used to retrain the detector, which then would have a better performance on the remaining lesions in that image. Conceivable is also an interactive setting with a suitable user interface, where the user manually labels a few lesions in an image, and the detector is retrained to find similar ones.

In chapter 4, we proposed an N-dimensional SURF descriptor. In future work, this descriptor can be applied to other imaging problems. A possible application is finding point correspondences in volumetric data. Many image registration methods only work if the images to be registered are already approximately aligned. The descriptors can be used for an initial landmark based registration. They can also be used for finding point correspondences and for a landmark based registration of 4-D image data, for instance in 3-D+t volumetric videos. An interesting application of quantized 3-D descriptors is finding motion patterns in 2-D+t video data as done in [Scov 07] with 3-D SIFT descriptors. For instance, it might turn out that a certain hand movement or a turn of the head correspond to certain visual words. This could be used for motion recognition, but also for improving body part recognition in videos. Again, the same can be done with 3-D+t video data, like, for instance, 3-D cardiac ultrasound videos. It might turn out that pathologies that are correlated with certain motion patterns correspond to a particular set of visual words.

Concerning esophagus segmentation, the method presented in chapter 5 approximates the esophagus contour in cross-sectional slices first using ellipses. Then, the ellipse assumption is abandoned, and the first estimate of the esophagus surface is non-rigidly refined. In the refinement step, the only shape knowledge that is incorporated is that the esophagus surface is smooth. In future work, this smoothness assumption can be replaced with a statistical shape model.

In the path inference step of the esophagus segmentation method, shape knowledge is modeled using a Markov chain. However, it models the ellipse transitions between neighboring slices and therefore only the local shape. The lower part of the esophagus is usually curved in ventral direction, while the upper part is less curved. This information cannot be captured with a local model. Future work on esophagus segmentation could replace this local shape model with a global one to include this kind of knowledge, even though this can increase the computational requirements because Markov models can be solved very efficiently. To keep the computational requirements tractable, the global shape model could be combined with a voting scheme: Esophagus segments detected in an initial step can vote for one or multiple instances of global esophagus shapes that fit to the detected segment.

The esophagus segmentation method presented in this work is not very specific to the esophagus. Apart from the explicit model of esophageal air, it is a method for detecting and

segmenting a relatively straight tubular structure. It can also be applied to other problems in medical imaging, for instance, segmenting non-bifurcated sections of vessels or the spinal channel.

Concerning the presented lymph node detection method, an option for future work would be to better handle size differences by training a single detector on multiple image scales such that the positive training examples always appear to the detector in the same size. During test, a resolution hierarchy is generated form an input image, and the detector is run on each hierarchy level. A detection at the coarsest level indicates a detected big lymph node, and a detection at the finest level a small lymph node. This technique was, for instance, used in [Chen 09]. Its effect is this that the class of the positive samples becomes more compact in the sense that is covers a smaller volume in the feature space, which simplifies the learning problem.

A further interesting research direction concerns the segmentation step of the presented lymph node detection method. In a variant of the detection method, a number of alternative segmentations are generated for each detection candidate. The purpose of these multiple segmentations, however, is only to improve the detection performance. We observed that the graph cuts segmentation with special weights (see section 6.6.2) generally produces the best results that are therefore selected as final segmentations. But it is also on option to compute the final segmentation S from multiple candidate segmentations $S_1, \ldots, S_{N_S}$, for instance, by learning a function

$$f : S_1, \ldots, S_{N_S} \mapsto S \tag{7.1}$$

from the ground truth, which is taken as S, and the candidate segmentations.

The segmentation of a lymph node itself is valuable, but it is also of interest whether a lymph node is malignant or not. This, however, is a very difficult classification problem because the only relevant features available in CT images are the size, the shape and the attenuation coefficients, and there is overlap of the malignant and the non-malignant class in these features [Baze 02]. The classification problem can be simplified by also considering the change of these features over time, which would require to examine a sequence of images taken at different times and registering the detected lymph nodes.

Physicians are also interested in lymph nodes in other body regions than the mediastinum, for instance the neck, the abdomen, the groin and the axillary regions. In future work, the method presented in this work can be extended to these body regions. Applying the presented discriminative model to lymph nodes of other body regions is straightforward. Adapting the spatial prior is more effort because it requires automatic segmentation methods of anatomical structures that are close to lymph nodes or can easily be confused with lymph nodes. Only adapting the "global" probabilistic atlas (6.3), however, is easier and just requires a set of anatomical landmarks in the body region of interest that can be detected automatically. Once lymph nodes in all body regions can be detected, it is possible to automatically stage the progress of a lymphoma disease based on the affected regions, for instance according to the Ann-Arbor classification [List 89].

Parts of the presented lymph node detection method can be reused in other detection problems. Similar spatial priors can, for instance, be used to capture anatomical knowledge that supports detecting pathologies like lung nodules, bone lesions or, in general, blob-like structures in medical image data. But the idea is also applicable to non-medical computer vision problems like pedestrian or car detection if the viewpoint of the camera remains

fixed. When a camera observes a street scene, pedestrians are more likely to appear in the image region of the sidewalk than the region of the driveway. The segmentation based features that are used in the verification step of the lymph node detection pipeline can also be applied to similar detection problems such as detecting blob-shaped lesions.

The main goal of the Medico research project (section 1.2) that will presumably end in 2012 is to enable semantic search on medical image databases. The methods presented in this thesis can be used to automatically generate a number of labels from CT images, and these labels can afterwards be used for search. The method presented in chapter 4 can provide a label with the body region. High level features that are of interest to a physician and can be generated by the lymph node detection and segmentation method include the number of nodes and shape features such as the volume and the sphericity.

Chapter 8

Summary

Introduction

This work presents different methods for the automatic analysis of computed tomography (CT) image data. It is embedded into the German research project Theseus Medico that was initiated to create an intelligent search engine for medical images. The idea of this project is to generate a formal textual semantic description of each image in a database. These textual descriptions can afterwards be searched. A special focus of the project is the lymphoma disease, which is a cancer of the lymphatic system. The lymph nodes of affected patients are enlarged. In order to assess the progress of the disease and to verify that the treatment is effective, physicians are interested in statistics such as the number of enlarged nodes, their spatial distribution and shape features like the volume and sphericity.

In this work, a top-down approach is presented for automatically detecting and segmenting lymph nodes from CT data. The focus is on the body region of the mediastinum that is of particular interest to physicians because mediastinal lymph nodes need to be considered during oncological examinations related to all kinds of cancer.

Boosting techniques for discriminative learning

Discriminative learning techniques, and in particular boosting techniques, have become very popular for computer vision applications. An overview and theoretical background of classification techniques used in this thesis is given in chapter 2.

The theory of boosting is closely connected with *probably approximately correct learning* (PAC), which is a framework proposed in [Vali 84] for the mathematical analysis of machine learning. The framework defines a strong and a weak PAC learning algorithm. Informally, given enough independently drawn labeled random samples of a distribution with two classes, a strong PAC learning algorithm is, with an arbitrarily high probability, able to learn a decision rule that has an arbitrarily low generalization error, if the class distribution is PAC learnable. This in turn means, informally, that the probability distributions of the two classes are non-overlapping. A weak PAC learning algorithm has the same properties as a strong PAC learning algorithm, with the only difference that the generalization error only needs to be slightly better than chance. It has been shown in [Scha 90] that any weak PAC learning algorithm can be transformed into a strong PAC learning algorithm, and this transformation is called „boosting“.

A number of different boosting algorithms have been proposed, with the most popular one among them being *AdaBoost* [Freu 95], which stands for „adaptive boosting“. AdaBoost greedily searches for a weighted subset within a (possibly large) pool of weak classifiers. Each of the training samples is furthermore associated with a weight. In each iteration, the weak classifier that minimizes the *weighted* misclassification error is added to the subset. The weight of misclassified samples is afterwards increased, and the weight of correctly classified samples is decreased. As shown in [Frie 00], AdaBoost optimizes an exponential loss function using Newton updates, and the classification score has a probabilistic interpretation.

In [Viol 01], a cascade of AdaBoost classifiers was very successfully used for object detection. In [Tu 05], this idea was extended, and the cascade was generalized into a binary decision tree with an AdaBoost classifier at each node, which is called *probabilistic boosting tree* (PBT). The time complexity of the PBT training and testing algorithm is analyzed, to the best of our knowledge for the first time.

Features for 2-D and 3-D image analysis

The performance of a classifier is bounded by the quality of the features. In chapter 3, state of the art features are presented that are used in this thesis. *Haar-like* features were introduced in [Viol 01] and gained considerable popularity. They compute integrals of the image intensity over axis aligned rectangular regions, and multiple integral values are weighted and summed to generate a single scalar feature. Although being simple, these features are powerful because they can be computed very efficiently with the help of an integral image. In [Tu 06], these features were extended to 3-D.

A major disadvantage of Haar-like features is that it is not efficiently possible to compute rotated versions. While computing rotated versions of the integral image is a possible workaround in 2-D, this is prohibitively computationally expensive in 3-D. A set of features that can be easily rotated and also scaled in 3-D are the so-called *steerable features* proposed in [Zhen 07]. These are simple point features such as the image intensity, the gradient magnitude, and the components of the gradient. These features are evaluated on a sampling pattern that can simply be a regular grid, or, for instance, reflect the shape of an object being sought. Scaled and rotated versions can be computed by transforming the sampling pattern accordingly.

Another set of features popular in computer vision are *speeded-up robust features* (SURF) [Bay 06, Bay 08]. They are similar to the older SIFT features [Lowe 99] and have a similarly high descriptive power, but have the advantage that they can be computed faster. A SURF descriptor is a 64 dimensional vector that contains statistics of the image gradients in a local neighborhood.

Estimating the visible body region

CT scans containing the chest area often also show other body regions, for instance the abdomen or the neck. In chapter 4, a method is proposed that estimates the visible body region of a CT scan. This can be used for finding the region of the mediastinum and thus for pruning large portions of the image when searching for mediastinal lymph nodes.

In order to quantify the body region, a body coordinate (BC) axis is used that runs in longitudinal direction. Its origin and unit length are patient-specific and depend on anatomical landmarks. The body region of a test volume is estimated by registering it

only along the longitudinal axis to a set of reference CT volume images with known body coordinates. During these 1-D registrations, an axial image slice of the test volume is compared to an axial slice of a reference volume by extracting a descriptor from both slices and measuring the similarity of the descriptors. A slice descriptor consists of histograms of visual words. Visual words are code words of a quantized feature space and can be thought of as classes of image patches with similar appearance. A slice descriptor is formed by sampling a slice on a regular 2-D grid and extracting a Speeded Up Robust Features (SURF) descriptor at each sample point. The codebook, or visual vocabulary, is generated in a training step by clustering SURF descriptors. Each SURF descriptor extracted from a slice is classified into the closest visual word (or cluster center) and counted in a histogram. A slice is finally described by a spatial pyramid of such histograms. An extension of the SURF descriptors to an arbitrary number of dimensions (N-SURF) is introduced. In this work, 2-SURF and 3-SURF descriptors are used. Cross-validation on 84 datasets shows the robustness of the results. The body portion can be estimated with an average error of 15.5mm within 9s.

Apart from finding body regions in large CT volumes, this method can also be used for automatic labeling of images with their body region. This additional application fits well into the Theseus Medico project as it allows to search for images in a database that contain a particular body region.

Automatic segmentation of the esophagus

A structure that can easily be confused with lymphatic issue in CT data due to a similar distribution of attenuation coefficients is the esophagus. At the same time, the esophagus is often surrounded with lymph nodes because it is a natural gateway into the body. The problem of detecting mediastinal lymph nodes is therefore considerably simplified when the outline of the esophagus is known. However, its versatile shape and appearance and its low contrast in CT make the segmentation a challenging problem.

In chapter 5 of this work, a multi-step method for automatic esophagus segmentation is presented: First, a detector that is trained to learn a discriminative model of the appearance. It consists of a cascade of binary classifiers. For each axial slice, it generates a set of candidates of the esophagus contour. At this stage, the contour within an axial slice is modeled as an ellipse, which is already a good approximation with only five degrees of freedom. The discriminative model relies on local features to detect the esophagus, which is occasionally filled with air. This air can confuse the model because locally, air inside the esophagus looks similar to respiratory air. Globally, they can however be easily distinguished. Therefore, the detector is combined with an explicit model of the distribution of respiratory and esophageal air. In the next step, prior shape knowledge is incorporated using a Markov chain model. We follow a “detect and connect” approach to obtain the maximum a posteriori estimate of the approximate esophagus shape from the hypothesis about the approximate esophagus contour in axial image slices. Finally, the surface of this approximation is non-rigidly deformed along its normals to better fit the boundary of the organ.

The method is compared to an alternative approach that uses a new particle filter variant instead of a Markov chain to infer the approximate esophagus shape, to the performance of a human observer and also to state of the art methods, which are all semiautomatic. Cross-validation on 144 CT scans showed that the Markov chain based approach clearly

outperforms the particle filter. It segments the esophagus with a mean error of 1.80 mm in less than 16 s on a standard PC. This is only 1 mm above the inter observer variability and can compete with the results of previously published semiautomatic methods.

Lymph node detection and segmentation

Finally, chapter 6 of this work presents a method that fully automatically detects and segments lymph nodes in 3-D computed tomography images of the chest.

Lymph nodes can easily be confused with other structures, it is therefore vital to incorporate as much anatomical prior knowledge as possible in order to achieve a good detection performance. Here, a learned prior of the spatial distribution is used to model this knowledge. Different prior types with increasing complexity are proposed and compared to each other. The simplest one serves as a baseline. It corresponds to the assumption that the spatial distribution of lymphatic tissue is constant, meaning that no prior is used. The second one is a a binary mask that marks organs and regions filled with air. These regions can be excluded from search because lymph nodes always lie in fat tissue. Next, a non-binary prior of lymphatic tissue is proposed that uses a probabilistic atlas learned from manually annotated data in the space of a reference patient. Finally, a variant is explored that additionally uses multiple local probabilistic atlases to better handle anatomical variations in the chest.

This is combined with a powerful discriminative model that detects lymph nodes from their appearance. It first generates a number of candidates of possible lymph node center positions. Two model variants are compared to each other. The first one directly detects the bounding box of a lymph node given its detected center point. In the second variant, a segmentation method is initialized with a detected candidate.

The graph cuts method is adapted to the problem of lymph nodes segmentation. A setting is proposed that requires only a single positive seed and at the same time solves the small cut problem of graph cuts. Furthermore, a feature set is proposed that is extracted from the segmentation. A classifier is trained on this feature set and used to reject false alarms.

Cross-validation on 54 CT datasets showed that for a fixed number of four false alarms per volume image, the detection rate is well more than doubled when using the spatial prior and is increased by 33% when using the segmentation based verification variant instead of the simple bounding box detector. In total, our proposed method detects mediastinal lymph nodes with a true positive rate of 52.0% at the cost of only 3.1 false alarms per volume image and a true positive rate of 60.9% with 6.1 false alarms per volume image.

Outlook

This work uses boosting techniques to solve various learning problems. Especially the probabilistic boosting tree has known issues and it is therefore worth considering to replace some of the classifiers with different ones, for instance with random forests. Future work could also explore using semi-supervised and online learning techniques to reduce the required amount of labeled training data, and to adapt a classifier to a particular patient or pathology.

It would be interesting to see if the N-SURF descriptors proposed in this work are of use for analyzing 4-D data, such as 3-D+t videos, for instance for finding point correspondences or certain motion patterns.

The final refinement step of the esophagus segmentation method presented in this work iteratively deforms and smoothes the previously detected surface. In future work, the smoothing could be replaced with a shape model. As it is, the method is furthermore fairly generic, and can also be applied to different straight tubular structures.

The performance of the lymph node detector could be further improved by training a single detector on multiple scales such that the lymph nodes seen during training all have a similar size, as this makes the learning problem simpler. The segmentation performance, in turn, can potentially be improved by generating candidate segmentations with different methods and learning a function that combines the candidate segmentations to produce the final one. Apart from segmentations, clinicians are also interested in whether a lymph node is malignant or not. This problem is simplified by also considering earlier scans of a particular patient. Finally, it would be interesting to adapt the presented method to other body regions, or to other blob-like lesion types.

Appendix A

Proof of N-D integral image theorem

This work proposes an extension of integral images to an arbitrary number of dimensions. Below, theorem 1 is repeated and afterwards proven.

Let I denote an N-dimensional image. An axis aligned N-D (hyper-)cuboid within the image region is described by its upper bounds $\boldsymbol{p}$ and lower bounds $\boldsymbol{q}$ with $p_i \geq q_i, i = 1 \ldots N$. Both p_i and q_i are voxel indices of the i-th dimension. Let the sum of voxels inside the (hyper-)cube be denoted by

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{i_1=q_1+1}^{p_1} \sum_{i_2=q_1+1}^{p_2} \cdots \sum_{i_N=q_N+1}^{p_N} I(\boldsymbol{i}). \tag{A.1}$$

Let further II denote the integral image of I

$$II(\boldsymbol{p}) = \begin{cases} \sum_{i_1=1}^{p_1} \sum_{i_2=1}^{p_2} \cdots \sum_{i_N=1}^{p_N} I(\boldsymbol{i}) & \text{if } i_j > 0 \, \forall j \in \{1 \ldots N\} \\ 0 & \text{else,} \end{cases} \tag{A.2}$$

let $T(N, d)$

$$T(N, d) = \left\{ \boldsymbol{t} \in \{0, 1\}^N \,\middle|\, \sum_{i=1}^{N} t_i = d \right\} \tag{A.3}$$

denote the set of permutations of a N-dimensional vector that contains d ones and $N - d$ zeros, and let $C_N(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q})$ be

$$C_N(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q}) = II \begin{pmatrix} (1 - t_1)p_1 + t_1 q_1 \\ \vdots \\ (1 - t_N)p_N + t_N q_N \end{pmatrix}, \tag{A.4}$$

where II denotes the integral image of image I, then the sum $C(\boldsymbol{p}, \boldsymbol{q})$ of the images values inside a hyper-box with upper bounds $\boldsymbol{p}$ and lower bounds $\boldsymbol{q}$ is

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{d=0}^{N} (-1)^d \sum_{\boldsymbol{t} \in T(N,d)} C_N(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q}). \tag{A.5}$$

Proof. We prove the theorem by complete induction over the dimension N.

Basis ($N = 1$): In this case, $\boldsymbol{p} = (p_1)$ and $\boldsymbol{q} = (q_1)$ are 1-D vectors and (A.5) becomes

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{d=0}^{1} (-1)^d \sum_{\boldsymbol{t} \in T(1,d)} C_1(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q}) \tag{A.6}$$

$$= \sum_{\boldsymbol{t} \in \{(0)\}} C_1(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q}) - \sum_{\boldsymbol{t} \in \{(1)\}} C_1(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q}) \tag{A.7}$$

$$= II\left((1-0)p_1 + 0 \cdot q_1\right) - II\left((1-1)p_1 + 1 \cdot q_1\right) \tag{A.8}$$

$$= II\left(p_1\right) - II\left(q_1\right) \tag{A.9}$$

$$= \sum_{i=1}^{p_1} I(i) - \sum_{i=1}^{q_1} I(i) \tag{A.10}$$

$$= \sum_{i=q_1+1}^{p_1} I(i), \tag{A.11}$$

which proves the theorem for $N = 1$.

Induction step: Let

$$T(N-1, d) \times \{x\} = \left\{ \boldsymbol{t} \in \{0,1\}^N \,\middle|\, \sum_{i=1}^{N-1} t_i = d, t_N = x \right\} \tag{A.12}$$

denote the set of N-D vectors that have $x \in \{0, 1\}$ as last component, and whose $N-1$ first elements are d zeros and $N-1-d$ ones. We start now by splitting the sum over the permutations $T(N, d)$ in two parts. First, we sum over all elements in $T(N, d)$ that have a zero as last component, and then over the elements that have a one as last component:

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{d=0}^{N} (-1)^d \left[\sum_{\boldsymbol{t} \in T(N-1,d) \times \{0\}} C_N(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q}) + \sum_{\boldsymbol{t} \in T(N-1,d-1) \times \{1\}} C_N(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q}) \right]. \tag{A.13}$$

After expanding $C_N(\boldsymbol{t}, \boldsymbol{p}, \boldsymbol{q})$, we get

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{d=0}^{N} (-1)^d \left[\sum_{\boldsymbol{t} \in T(N-1,d) \times \{0\}} II \begin{pmatrix} (1-t_1)p_1 + t_1 q_1 \\ \vdots \\ (1-t_{N-1})p_{N-1} + t_{N-1} q_{N-1} \\ p_N \end{pmatrix} \right. \tag{A.14}$$

$$\left. + \sum_{\boldsymbol{t} \in T(N-1,d-1) \times \{1\}} II \begin{pmatrix} (1-t_1)p_1 + t_1 q_1 \\ \vdots \\ (1-t_{N-1})p_{N-1} + t_{N-1} q_{N-1} \\ q_N \end{pmatrix} \right]. \tag{A.15}$$

Now we define $\boldsymbol{p}'$ and $\boldsymbol{q}'$ as

$$\boldsymbol{p}' = (p_1 \dots p_{N-1})^T \tag{A.16}$$

$$\boldsymbol{q}' = (q_1 \dots q_{N-1})^T \tag{A.17}$$

the $N-1$-D vectors that contain the first $N-1$ entries of $\boldsymbol{p}$ and $\boldsymbol{q}$, respectively, and

$$II^i_{N-1}(\boldsymbol{p}') = \begin{cases} \sum_{k_1=1}^{p_1} \cdots \sum_{k_{N-1}=1}^{p_{N-1}} I(k_1, \ldots, k_{N-1}, i) & \text{if } k_j > 0 \, \forall j \in \{1 \ldots N-1\} \\ 0 & \text{else} \end{cases} \tag{A.18}$$

as the $N-1$-D integral image of the i-th slice of image I, where "slice" refers to the outermost dimension N, and let further $\boldsymbol{a} \circ \boldsymbol{b}$ be the Hadamard (element-wise) product of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$. Then, we can rewrite (A.14) as

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{d=0}^{N} (-1)^d \left[\sum_{\boldsymbol{t} \in T(N-1,d)} \sum_{i=1}^{p_N} II^i_{N-1}\left((\mathbf{1} - \boldsymbol{t}) \circ \boldsymbol{p}' + \boldsymbol{t} \circ \boldsymbol{q}'\right) \right. \tag{A.19}$$

$$\left. + \sum_{\boldsymbol{t} \in T(N-1,d-1)} \sum_{i=1}^{q_N} II^i_{N-1}\left((\mathbf{1} - \boldsymbol{t}) \circ \boldsymbol{p}' + \boldsymbol{t} \circ \boldsymbol{q}'\right) \right]. \tag{A.20}$$

After expansion, we get

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{d=0}^{N} (-1)^d \sum_{\boldsymbol{t} \in T(N-1,d)} \sum_{i=1}^{p_N} II^i_{N-1}\left((\mathbf{1} - \boldsymbol{t}) \circ \boldsymbol{p}' + \boldsymbol{t} \circ \boldsymbol{q}'\right) \tag{A.21}$$

$$+ \sum_{d=0}^{N} (-1)^d \sum_{\boldsymbol{t} \in T(N-1,d-1)} \sum_{i=1}^{q_N} II^i_{N-1}\left((\mathbf{1} - \boldsymbol{t}) \circ \boldsymbol{p}' + \boldsymbol{t} \circ \boldsymbol{q}'\right). \tag{A.22}$$

By using that $T(N-1, N) = T(N-1, -1) = \emptyset$ are both the empty set, this can be rewritten as

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{d=0}^{N-1} (-1)^d \sum_{\boldsymbol{t} \in T(N-1,d)} \sum_{i=1}^{p_N} II^i_{N-1}\left((\mathbf{1} - \boldsymbol{t}) \circ \boldsymbol{p}' + \boldsymbol{t} \circ \boldsymbol{q}'\right) \tag{A.23}$$

$$+ \sum_{d=1}^{N} (-1)^d \sum_{\boldsymbol{t} \in T(N-1,d-1)} \sum_{i=1}^{q_N} II^i_{N-1}\left((\mathbf{1} - \boldsymbol{t}) \circ \boldsymbol{p}' + \boldsymbol{t} \circ \boldsymbol{q}'\right). \tag{A.24}$$

After adapting the limits of the outer sum in (A.24) and changing the order of summation, we get

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{i=1}^{p_N} \sum_{d=0}^{N-1} (-1)^d \sum_{\boldsymbol{t} \in T(N-1,d)} II^i_{N-1}\left((\mathbf{1} - \boldsymbol{t}) \circ \boldsymbol{p}' + \boldsymbol{t} \circ \boldsymbol{q}'\right) \tag{A.25}$$

$$- \sum_{i=1}^{q_N} \sum_{d=0}^{N-1} (-1)^d \sum_{\boldsymbol{t} \in T(N-1,d)} II^i_{N-1}\left((\mathbf{1} - \boldsymbol{t}) \circ \boldsymbol{p}' + \boldsymbol{t} \circ \boldsymbol{q}'\right). \tag{A.26}$$

Now we use that (A.1) holds for $N-1$-D integral images. If we define $C^j_{N-1}(\boldsymbol{p}', \boldsymbol{q}')$ as the sum over the voxels in the j-th slice of image I inside the (hyper-)box with bounds $\boldsymbol{p}'$ and $\boldsymbol{q}'$, it is

$$C^j_{N-1}(\boldsymbol{p}', \boldsymbol{q}') = \sum_{i_1=q_1+1}^{p_1} \dots \sum_{i_{N-1}=q_{N-1}+1}^{p_{N-1}} I(i_1 \dots i_{N-1}, j) \tag{A.27}$$

$$= \sum_{d=0}^{N-1} (-1)^d \sum_{\boldsymbol{t} \in T(N-1,d)} II^j_{N-1} \left((\mathbf{1} - \boldsymbol{t}) \circ \boldsymbol{p}' + \boldsymbol{t} \circ \boldsymbol{q}' \right). \tag{A.28}$$

By inserting (A.28) into (A.26), we obtain

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{i=1}^{p_N} C^i_{N-1}(\boldsymbol{p}', \boldsymbol{q}') - \sum_{i=1}^{q_N} C^i_{N-1}(\boldsymbol{p}', \boldsymbol{q}'). \tag{A.29}$$

Since the sums over the slices $1 \dots q_N$ cancel out, this is equal to

$$C(\boldsymbol{p}, \boldsymbol{q}) = \sum_{i=q_N+1}^{p_N} C^i_{N-1}(\boldsymbol{p}', \boldsymbol{q}') \tag{A.30}$$

$$= \sum_{i_N=q_N+1}^{p_N} \sum_{i_1=q_1+1}^{p_1} \dots \sum_{i_{N-1}=q_{N-1}+1}^{p_{N-1}} I(i_1 \dots i_N), \tag{A.31}$$

which proves the theorem. □

Appendix B

Notation

B.1 Boosting techniques for discriminative learning

y	class variable
Y	set of class variables
K	number of classes
$\boldsymbol{x}$	feature vector
X	feature space
$C(k, j)$	cost of deciding for the class y_j if the true class is y_k
$\mathbb{E}(C)$	expected costs
y_{opt}	class that minimizes the expected costs
T	number of weak classifiers to train
N	number of labeled training samples
w_n	weight of sample n
h_t	tth weak classifier
$\mathcal{H}$	set of all possible weak classifiers
ϵ_t	weighted classification error of weak classifier h_t
$[a \neq b]$	equals 1 if $a \neq b$ and 0 otherwise
α_t	weight of weak classifier h_t
Z	normalization term
H	strong classifier
$J(H)$	expected exponential costs of strong classifier H
H_t	strong classifier after the tth AdaBoost iteration
$w_n^{(t)}$	weight of sample n after the tth AdaBoost iteration
L	number of layers in the PBT
Δp	minimum absolute difference the score of a sample must have from $\frac{1}{2}$ in order not to be passed to both children during PBT training and testing
$\mathcal{N}$	a PBT node
$F_{\mathcal{N}}$	classification score of PBT node $\mathcal{N}$
S	set of labeled training samples
$\epsilon_{\max}$	maximum weighted classification error
$S_{\text{left}}, S_{\text{right}}$	training set of left/right child node

a	sample growth factor during PBT training
l	current PBT tree layer

B.2 Features for 2-D and 3-D image analysis

p_1	voxel index along the horizontal axis
p_2	voxel index along the vertical axis
I	image
II	integral image of I
x, y, z	3-D world coordinates
$\boldsymbol{p}$	point in voxel coordinates
$\boldsymbol{r}$	point in world coordinates
c_i	Haar filter response that approximates the directional derivative of the smoothed image along the ith coordinate axis
s	scale of the SURF descriptor that equals the spacing of the sample pattern and half the size of the Haar filters
$\boldsymbol{v}$	4-D feature vector of a bin of the SURF sampling pattern

B.3 Estimating the visible body region of 3-D CT scans

N_D	dimension of the slice descriptor of [Dick 08]
x, y, z	3-D world coordinates
N	number of image dimensions
I	image
II	integral image
$\boldsymbol{p}, \boldsymbol{q}$	points in voxel coordinates
$T(N, d)$	set of permutations of an N-dimensional vector that contains d ones and $N - d$ zeros
$\boldsymbol{t}$	N-dimensional vector containing zeros and ones
$C(\boldsymbol{p}, \boldsymbol{q})$	sum of voxels inside an axis aligned cuboid with opposite corners $\boldsymbol{p}$ and $\boldsymbol{q}$
b	number of bins per dimension the SURF sampling pattern is partitioned into
σ	standard deviation of the Gaussian centered at the interest point that is used to weight the Haar filter responses
$\boldsymbol{v}$	$2N$-D feature vector of a bin of the SURF sampling pattern
r	radius of sphere-shaped sampling pattern used to determine the canonical orientation
G	number of points in this sphere-shaped sampling pattern
$\boldsymbol{R}$	rotation matrix
$\boldsymbol{c}^{(i)}$	ith gradient vector used to determine the canonical orientation
$\boldsymbol{u}^{(i)}$	ith eigenvector of the PCA of the gradient vectors
$\overline{\boldsymbol{c}}$	mean gradient vector
$\boldsymbol{u}_a^{(i)}$	ith eigenvector of the PCA with canonical direction

$\boldsymbol{u}_b^{(i)}$	normalized $\boldsymbol{u}_a^{(i)}$
$\boldsymbol{c}_p^{(i)}$	vector $\boldsymbol{c}^{(i)}$ that is projected into the plane orthogonal to $\overline{\boldsymbol{c}}$ and transformed into a $N-1$-D basis
X, Y	two sets of n dimensional features
n	dimension of the feature space
l	current pyramid level
L	maximum pyramid level
D_l	number of bins in the histogram at level l
H_X^l, H_Y^l	histograms of the features in X and Y at level l
$\mathcal{I}(H_a, H_b)$	intersection of histograms H_a and H_b
$\mathcal{I}^l$	short for $\mathcal{I}(H_X^l, H_Y^l)$
κ^L	pyramid match kernel
M	number of feature classes („visual words“)
X_i, Y_i	sets of 2-D vectors of the image coordinates of the features of class i in X and Y, respectively
$\mathcal{K}^L$	spatial pyramid match kernel
n_s	minimum number of consecutive non-air voxels that are needed to trigger the patient detector
j, k	two image slices taken from the 3-D volume images I_j and I_k
z_j, z_k	vertical level where slices j and k where extracted
$d(j, k)$	dissimilarity measure for slices j and k
H_j	concatenated histograms of visual words of slice j
J, K	two lists of subsequent axial slices
$f(z)$	objective function for rigid registration
C	set of candidate z offsets
c_i	ith z candidate
w_i	weight of ith z candidate
g_s	1-D convolution kernel at scale s that both smoothes and computes the second derivative
$f_d(z_1, z_2)$	objective function for non-rigid registration using dynamic time warping
Δ_z	height of the test stack in world coordinates
$e_{\text{top}}, e_{\text{bottom}}$	registration error at the top and bottom of the test volume
e	averaged registration error

B.4 Automatic segmentation of the esophagus in 3-D CT scans

$\boldsymbol{e}$	ellipse parameter vector
$\boldsymbol{t}$	ellipse center
θ	ellipse rotation angle
$\boldsymbol{s} = (a, b)^T$	semi major and semi minor axis of the ellipse
N	number of ellipse candidates
m	binary class variable
$\boldsymbol{H}(\boldsymbol{t})$	3-D Haar-like feature vector extracted at position $\boldsymbol{t}$

C_{T1}	first set of translation candidates
N_{T1}	number of candidates in C_{T1}
C_{T2}	second set of translation candidates
N_{T2}	number of candidates in C_{T2}
$\boldsymbol{S}(\boldsymbol{t}, \theta)$	vector of steerable features extracted at position $\boldsymbol{t}$ and angle θ
$\boldsymbol{S}(\boldsymbol{t}, \theta, \boldsymbol{s})$	vector of steerable features extracted at position $\boldsymbol{t}$, angle θ and scale $\boldsymbol{s}$
N_{TR}	number of rotation and translation candidates
C	set of ellipse candidates
$B(\boldsymbol{t})$	value of binary mask of respiratory air at position $\boldsymbol{t}$
$E(\boldsymbol{t})$	value of binary mask of esophageal air at position $\boldsymbol{t}$
$\boldsymbol{p}$	point of gravity of a 2-D region of esophageal air
$g(r)$	1-D filter function, similar to a Gaussian with a standard deviation of σ_a but with limited support
w	support of the filter g
$A(\boldsymbol{t})$	map of the esophagus probability at position $\boldsymbol{t}$ estimated from esophageal air
$C(\boldsymbol{t})$	model of the overall esophagus probability at position $\boldsymbol{t}$ given the global distribution of air
θ_d	distance threshold in the clustering step
$\boldsymbol{c}^{(i)}$	ith center after candidate clustering
$\gamma^{(i)}$	score of $\boldsymbol{c}^{(i)}$
$\boldsymbol{v}_t$	observed axial image slice t
Φ_t	clique potential of observation clique t in the Markov model
Ψ_t	clique potential of transition clique $t \rightarrow t+1$ in the Markov model
$\boldsymbol{\Sigma}_p$	covariance matrix of the normal distribution of the translation transition
$\boldsymbol{m}_p$	mean of the normal distribution of the translation transition
$\boldsymbol{\Sigma}_s$	covariance matrix of the normal distribution of the scale transition
$\boldsymbol{m}_s$	mean of the normal distribution of the scale transition
$\sigma_r(l)$	standard deviation of the normal distribution of the rotation transition at circularity level l
$m_r(l)$	mean of the normal distribution of the rotation transition at circularity level l
$\hat{\boldsymbol{c}}_{1:T}^{(\mathrm{MAP})}$	maximum a posteriori estimate of the esophagus path, parameterized as a sequence of T ellipses
$\boldsymbol{\Sigma}_{p2}$	covariance matrix of the normal distribution of the second order translation transition
$\boldsymbol{m}_{p2}$	mean of the normal distribution of the second order translation transition
T	number of slices
$\mathcal{S}_t$	set of particles at slice t
$\boldsymbol{e}_t^{(i)}$	ellipse parameters (state vector) of particle i at slice t
$P_t^{(i)}$	weight of particle i at slice t

I	number of particles
δ	Dirac delta function

B.5 Lymph node detection and segmentation from chest CT

$\boldsymbol{t}$	lymph node center
$B(\boldsymbol{t})$	binary mask of organs evaluated at position $\boldsymbol{t}$
$G(\boldsymbol{t})$	global probabilistic atlas of the spatial lymph node distribution evaluated at position $\boldsymbol{t}$
$L_i(\boldsymbol{t})$	ith local probabilistic atlas evaluated at position $\boldsymbol{t}$
M	number of local probabilistic atlases
$L(\boldsymbol{t})$	combination of all M local probabilistic atlases at position $\boldsymbol{t}$
$w_i(\boldsymbol{t})$	weight of local probabilistic atlas i at position $\boldsymbol{t}$
$Z(\boldsymbol{t})$	normalization term at position $\boldsymbol{t}$
$d_i(\boldsymbol{t})$	distance of $\boldsymbol{t}$ to the surface of organ i
θ	maximal distance from the organ surface up to that the corresponding probabilistic atlas still has support
$w_{\max}(\boldsymbol{t})$	weight of most influential local probabilistic atlas at position $\boldsymbol{t}$
$\boldsymbol{H}(\boldsymbol{t})$	3-D Haar-like feature vector extracted at position $\boldsymbol{t}$
C_{H1}, C_{H2}	sets of lymph node center candidates generated by first and second Haar feature based detector, respectively
$\boldsymbol{b}$	axis aligned bounding box parameters
$\boldsymbol{S}(\boldsymbol{t}, \boldsymbol{s})$	vector of steerable features extracted at position $\boldsymbol{t}$ and scale $\boldsymbol{s}$
C_S	set of lymph node bounding box candidates
$\boldsymbol{A}(\boldsymbol{t})$	gradient aligned feature vector extracted at position $\boldsymbol{t}$
C_A	set of lymph node center candidates generated by the gradient aligned feature based detector
s	the source node
t	the sink node
λ_i	unary weight of voxel i
β_{ij}	binary weight (capacity) of the edge from voxel i to voxel j
c_{ij}	capacity from node i to node j in the graph
x_i	binary label of voxel i that is either "foreground" or "background"
$\mathcal{B}$	set containing the source s and all voxels labeled as "foreground"
$\mathcal{W}$	set containing the sink t and all voxels labeled as "background"
$\mathcal{C}$	cost of a cut that separates the source s from the sink t
$\delta(a, b)$	Kronecker delta
N	number of image voxels
r_{ij}	distance of the edge from voxel i to voxel j to the positive seed
I_i	image intensity of voxel i
u	normalization constant
$\mathcal{K}(r)$	set of edges intersected by a sphere with radius r

$K(r)$	integrated capacities of edges intersecting a sphere with radius r
d_{ij}	Euclidean distance of voxels i and j
α_{ij}	angle between the edge from voxel i to voxel j and the line from the positive seed to the center of the edge
$p(\text{out}_{ij})$	probability that the edge $i \rightarrow j$ is pointing in outward direction
$\boldsymbol{D}(\boldsymbol{t})$	segmentation based features extracted at position $\boldsymbol{t}$
σ_β	parameter of standard graph cuts weights that is set according to the noisiness of the data
g	zero-mean 3-D Gaussian
I_W	windowed image
E	edge image
θ_{WS}	threshold of the edge image E
e_{max}	maximum value of the edge image E
l_{WS}	relative watershed flooding level
L_{WS}	absolute watershed flooding level
N_S	number of segmentations
$\boldsymbol{D}_i(\boldsymbol{t})$	segmentation based features extracted from segmentation i at position $\boldsymbol{t}$
$\boldsymbol{D}_{\text{combined}}(\boldsymbol{t})$	features extracted from different segmentations at position $\boldsymbol{t}$ concatenated into a single vector
$\boldsymbol{D}_{\text{WS}}^{(l_{\text{WS}})}(\boldsymbol{t})$	segmentation based features extracted at position $\boldsymbol{t}$ from a watershed transform with a relative flooding level of l_{WS}
$\boldsymbol{D}_{\text{WS,GC}}(\boldsymbol{t})$	concatenation of features extracted at position $\boldsymbol{t}$ from a watershed and a graph cuts segmentation

List of Figures

List of Tables

Bibliography

[Barb 10] A. Barbu, M. Suehling, J. Xu, D. Liu, S. K. Zhou, and D. Comaniciu. "Automatic Detection and Segmentation of Axillary Lymph Nodes". In: *MICCAI (1), Lecture Notes in Computer Science, LNCS 6361*, pp. 28–36, Beijing, China, September 2010.

[Bay 06] H. Bay, T. Tuytelaars, and L. Van Gool. "SURF: Speeded-Up Robust Features". In: *European Conference on Computer Vision (ECCV), Lecture Notes in Computer Science, LNCS 3951*, pp. 404–417, Graz, Austria, May 2006.

[Bay 08] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. "Speeded-Up Robust Features (SURF)". *Computer Vision and Image Understanding*, Vol. 110, No. 3, pp. 346 – 359, June 2008.

[Baze 02] A. Bazemore and D. Smucker. "Lymphadenopathy and Malignancy". *American Family Physician*, Vol. 66, No. 11, pp. 2103–2111, December 2002.

[Beuc 92] S. Beucher and F. Meyer. *Mathematical Morphology in Image Processing*, Chap. 12, The Morphological Approach to Segmentation: The Watershed Transformation, pp. 433–482. CRC Press; First Edition, New York, NY, USA, September 1992.

[Beuc 94] S. Beucher. "Watershed, hierarchical segmentation and waterfall algorithm". In: J. Serra and P. Soille, Eds., *Proc. Mathematical Morphology and its Applications to Image Processing*, pp. 69–76, Kluwer Academic Publishers, Dordrecht, The Netherlands, September 1994.

[Bhat 05] A. Bhattacharya, V. Ljosa, J.-Y. Pan, M. R. Verardo, H. Yang, C. Faloutsos, and A. K. Singh. "ViVo: Visual Vocabulary Construction for Mining Biomedical Images". In: *Fifth IEEE International Conference on Data Mining (ICDM)*, pp. 50–57, Houston, Texas, USA, November 2005.

[Bish 07] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, New York, NY, USA, 1st ed. 2006. corr. 2nd printing Ed., October 2007.

[Book 89] F. Bookstein. "Principal Warps: Thin-Plate Splines and the Decomposition of Deformations". *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on*, Vol. 11, No. 6, pp. 567–585, June 1989.

[Boyk 03] Y. Boykov and V. Kolmogorov. "Computing geodesics and minimal surfaces via graph cuts". In: *Computer Vision (ICCV), IEEE International Conference on*, pp. 26 –33, Nice, France, October 2003.

[Boyk 04] Y. Boykov and V. Kolmogorov. "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision". *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on*, Vol. 26, No. 9, pp. 1124 –1137, September 2004.

[Boyk 06] Y. Boykov and G. Funka-Lea. "Graph Cuts and Efficient N-D Image Segmentation". *International Journal of Computer Vision*, Vol. 70, No. 2, pp. 109–131, November 2006.

[Chap 06] O. Chapelle, B. Schölkopf, and A. Zien, Eds. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, USA, 2006.

[Chen 09] T. Chen, W. Zhang, S. Good, K. S. Zhou, and D. Comaniciu. "Automatic Ovarian Follicle Quantification from 3D Ultrasound Data Using Global/local Context with Database Guided Segmentation". In: *Computer Vision (ICCV), IEEE International Conference on*, pp. 795 – 802, Kyoto, Japan, September 2009.

[Crow 84] F. C. Crow. "Summed-area tables for texture mapping". In: *11th annual conference on Computer graphics and interactive techniques, SIGGRAPH*, pp. 207–212, ACM, New York, NY, USA, July 1984.

[Das 09] P. Das, O. Veksler, V. Zavadsky, and Y. Boykov. "Semiautomatic segmentation with compact shape prior". *Image Vision Comput.*, Vol. 27, No. 1-2, pp. 206–219, January 2009.

[Dick 08] V. Dicken, B. Lindow, L. Bornemann, J. Drexl, A. Nikoubashman, and H.-O. Peitgen. "Realtime image recognition of body parts scanned in computed tomography datasets". In: *22nd International Congress on Computer Assisted Radiology and Surgery (CARS)*, Kuessaberg, Germany, June 2008. (poster no. 200).

[Diet 00] T. Dietterich. "Ensemble Methods in Machine Learning". In: *Multiple Classifier Systems (MCS), . Lecture Notes in Computer Science, LNCS 1857*, pp. 1–15, Cagliari, Italy, June 2000.

[Dorn 06] J. Dornheim, H. Seim, B. Preim, I. Hertel, and G. Strauß. "Segmentation of Neck Lymph Nodes in CT Datasets with Stable 3D Mass-Spring Models". In: *MICCAI (2), Lecture Notes in Computer Science, LNCS 4191*, pp. 904–911, Copenhagen, Denmark, October 2006.

[Dorn 08] L. Dornheim and J. Dornheim. "Automatische Detektion von Lymphknoten in CT-Datensätzen des Halses". In: *Bildverarbeitung für die Medizin, Informatik aktuell, ISBN 978-3-540-78639-9. Springer*, pp. 308–312, Berlin, Germany, April 2008.

[Duwe 05] B. V. Duwe, D. H. Sterman, and A. I. Musani. "Tumors of the Mediastinum". *Chest*, Vol. 128, No. 4, pp. 2893–2909, October 2005.

[Duyg 02] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. "Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary". In: *European Conference on Computer Vision (ECCV), Lecture Notes in Computer Science, LNCS 2351*, pp. 97–112, Copenhagen, Denmark, June 2002.

[Fei 05] L. Fei-Fei and P. Perona. "A Bayesian hierarchical model for learning natural scene categories". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pp. 524–531, San Diego, CA, USA, June 2005.

[Feue 09] M. Feuerstein, D. Deguchi, T. Kitasaka, S. Iwano, K. Imaizumi, Y. Hasegawa, Y. Suenaga, and K. Mori. "Automatic Mediastinal Lymph Node Detection in Chest CT". In: *SPIE Medical Imaging*, Orlando, Florida, USA, February 2009.

[Feul 09a] J. Feulner, S. K. Zhou, A. Cavallaro, S. Seifert, J. Hornegger, and D. Comaniciu. "Fast Automatic Segmentation of the Esophagus from 3D CT Data Using a Probabilistic Model". In: *MICCAI (1), Lecture Notes in Computer Science, LNCS 5761*, pp. 255–262, London, UK, September 2009.

[Feul 09b] J. Feulner, S. K. Zhou, S. Seifert, A. Cavallaro, J. Hornegger, and D. Comaniciu. "Estimating the Body Portion of CT Volumes by Matching Histograms of Visual Words". In: *SPIE Medical Imaging*, Orlando, Florida, USA, February 2009.

[Feul 10a] J. Feulner, S. K. Zhou, M. Huber, J. Hornegger, D. Comaniciu, and A. Cavallaro. "Lymph Node Detection in 3-D Chest CT using a Spatial Prior Probability". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pp. 2926 – 2932, San Francisco, CA, USA, June 2010.

[Feul 10b] J. Feulner, S. K. Zhou, M. Huber, A. Cavallaro, J. Hornegger, and D. Comaniciu. "Model-Based Esophagus Segmentation from CT Scans Using a Spatial Probability Map". In: *MICCAI (1), Lecture Notes in Computer Science, LNCS 6361*, pp. 95–102, Beijing, China, September 2010.

[Feul 11a] J. Feulner, S. Zhou, M. Hammon, S. Seifert, M. Huber, D. Comaniciu, J. Hornegger, and A. Cavallaro. "A Probabilistic Model for Automatic Segmentation of the Esophagus in 3-D CT Scans". *Medical Imaging, IEEE Transactions on*, Vol. 30, No. 6, pp. 1252 – 1264, June 2011.

[Feul 11b] J. Feulner, S. K. Zhou, M. Hammon, J. Hornegger, and D. Comaniciu. "Segmentation based Features for Lymph Node Detection from 3-D Chest CT". In: *Machine Learning in Medical Imaging (MLMI), MICCAI workshops, Lecture Notes in Computer Science, LNCS 7009*, pp. 91–99, Toronto, Canada, September 2011.

[Feul 11c] J. Feulner, S. Zhou, E. Angelopoulou, S. Seifert, A. Cavallaro, J. Hornegger, and D. Comaniciu. "Comparing axial CT slices in quantized N-dimensional SURF descriptor space to estimate the visible body region". *Computerized Medical Imaging and Graphics*, Vol. 35, No. 3, pp. 227 – 236, April 2011.

[Fies 08a] A. Fieselmann, S. Lautenschläger, F. Deinzer, M. John, and B. Poppe. "Esophagus Segmentation by Spatially-Constrained Shape Interpolation". In: *Bildverarbeitung für die Medizin, Informatik aktuell, ISBN 978-3-540-78639-9. Springer*, pp. 247–+, Berlin, Germany, April 2008.

[Fies 08b] A. Fieselmann, S. Lautenschläger, F. Deinzer, and B. Poppe. "Automatic Detection of Air Holes Inside the Esophagus in CT Images". In: *Bildverarbeitung für die Medizin, Informatik aktuell, ISBN 978-3-540-78639-9. Springer*, pp. 397–401, Berlin, Germany, April 2008.

[Fitz 99] M. Fitzgibbon, A. W.and Pilu and R. B. Fisher. "Direct least-squares fitting of ellipses". *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on*, Vol. 21, No. 5, pp. 476–480, May 1999.

[Flor 05] C. Florin, N. Paragios, and J. Williams. "Particle Filters, a Quasi-Monte-Carlo-Solution for Segmentation of Coronaries". In: *MICCAI, Lecture Notes in Computer Science, LNCS 3749*, pp. 246–253, Palm Springs, CA, USA, October 2005.

[Freu 90] Y. Freund. "Boosting a weak learning algorithm by majority". In: *COLT '90: Proceedings of the third annual workshop on Computational learning theory*, pp. 202–216, Rochester, NY, USA, August 1990.

[Freu 95] Y. Freund and R. E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting". In: *EuroCOLT '95: Proceedings of the Second European Conference on Computational Learning Theory, Lecture Notes in Computer Science, LNCS 904*, pp. 23–37, Barcelona, Spain, March 1995.

[Freu 96] Y. Freund and R. E. Schapire. "Experiments with a New Boosting Algorithm". In: *International Conference on Machine Learning (ICML)*, pp. 148–156, Omnipress, Bellevue, WA, USA, June 1996.

[Freu 99] Y. Freund and R. Schapire. "A short introduction to boosting". *J. Japan. Soc. for Artif. Intel.*, Vol. 14, No. 5, pp. 771–780, September 1999. (In Japanese, translation by Naoki Abe).

[Frie 00] J. Friedman, T. Hastie, and R. Tibshirani. "Additive logistic regression: a statistical view of boosting". *Annals of Statistics*, Vol. 28, No. 2, pp. 337–407, April 2000.

[Funk 06] G. Funka-lea, Y. Boykov, C. Florin, M. p. Jolly, R. Moreau-gobard, R. Ramaraj, and D. Rinck. "Automatic heart isolation for CT coronary visualization using graph-cuts". In: *IEEE International Symposium on Biomedical Imaging (ISBI): From Nano to Macro*, pp. 614–617, Arlington, VA, USA, April 2006.

[Grau 05] K. Grauman and T. Darrell. "The pyramid match kernel: discriminative classification with sets of image features". In: *Computer Vision (ICCV), IEEE International Conference on*, pp. 1458–1465, Beijing, China, October 2005.

[Grei 89] D. M. Greig, B. T. Porteous, and A. H. Seheult. "Exact Maximum A Posteriori Estimation for Binary Images". *Journal of the Royal Statistical Society. Series B (Methodological)*, Vol. 51, No. 2, pp. 271–279, 1989.

[Guel 02] M. O. Gueld, M. Kohnen, D. Keysers, H. Schubert, B. B. Wein, J. Bredno, and T. M. Lehmann. "Quality of DICOM header information for image categorization". In: *SPIE Medical Imaging*, San Diego, CA, USA, February 2002.

[Huan 06] T.-C. Huang, G. Zhang, T. Guerrero, G. Starkschall, K.-P. Lin, and K. Forster. "Semi-automated CT segmentation using optic flow and Fourier interpolation techniques". *Computer Methods and Programs in Biomedicine*, Vol. 84, No. 2-3, pp. 124–134, December 2006.

[Iona 10] R. I. Ionasec, I. Voigt, B. Georgescu, Y. Wang, H. Houle, H. Fernando-Vega, N. Navab, and D. Comaniciu. "Patient-Specific Modeling and Quantification of the Aortic and Mitral Valves From 4-D Cardiac CT and TEE". *Medical Imaging, IEEE Transactions on*, Vol. 29, No. 9, pp. 1636–1651, September 2010.

[Isar 98a] M. Isard and A. Blake. "CONDENSATION: Conditional Density Propagation for Visual Tracking". *International Journal of Computer Vision*, Vol. 29, No. 1, pp. 5–28, August 1998.

[Isar 98b] M. Isard and A. Blake. "A Smoothing Filter for CONDENSATION". In: *European Conference on Computer Vision (ECCV), Lecture Notes in Computer Science, LNCS 1406*, pp. 767–781, Freiburg, Germany, June 1998.

[Kear 94] M. J. Kearns and U. V. Vazirani. *An introduction to computational learning theory*. MIT Press, Cambridge, MA, USA, 1994.

[Kind 80] R. Kindermann and J. L. Snell. *Markov Random Fields and Their Applications*. American Mathematical Society, Providence, RI, USA, 1980.

[Kita 07] T. Kitasaka, Y. Tsujimura, Y. Nakamura, K. Mori, Y. Suenaga, M. Ito, and S. Nawano. "Automated Extraction of Lymph Nodes from 3-D Abdominal CT Images Using 3-D Minimum Directional Difference Filter". In: *MICCAI (2), Lecture Notes in Computer Science, LNCS 4792*, pp. 336–343, Brisbane, Australia, October 2007.

[Krau 04] N. Krause and Y. Singer. "Leveraging the margin more carefully". In: *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, p. 63, ACM Press, Banff, Alberta, Canada, July 2004.

[Ksch 01] F. Kschischang, B. Frey, and H.-A. Loeliger. "Factor graphs and the sum-product algorithm". *Information Theory, IEEE Transactions on*, Vol. 47, No. 2, pp. 498–519, January 2001.

[Kthe 03] U. Köthe. "Edge and Junction Detection with an Improved Structure Tensor". In: *DAGM-Symposium, Lecture Notes in Computer Science, LNCS 2781*, pp. 25–32, Magdeburg, Germany, September 2003.

[Kuru 10] S. Kurugol, N. Ozay, J. G. Dy, G. C. Sharp, and D. H. Brooks. "Locally Deformable Shape Model to Improve 3D Level Set Based Esophagus Segmentation". In: *Pattern Recognition (ICPR), IEEE International Conference on*, pp. 3955–3958, Istanbul, Turkey, August 2010.

[Lang 06] A. J. de Langen, P. Raijmakers, I. Riphagen, M. A. Paul, and O. S. Hoekstra. "The size of mediastinal lymph nodes and its relation with metastatic involvement: a meta-analysis". *European Journal of Cardio-Thoracic Surgery*, Vol. 29, No. 1, pp. 26–29, January 2006.

[Laze 06] S. Lazebnik, C. Schmid, and J. Ponce. "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pp. 2169–2178, New York, NY, USA, June 2006.

[List 89] T. Lister, D. Crowther, S. Sutcliffe, E. Glatstein, G. Canellos, R. Young, S. Rosenberg, C. Coltman, and M. Tubiana. "Report of a committee convened to discuss the evaluation and staging of patients with Hodgkin's disease: Cotswolds meeting [published erratum appears in J Clin Oncol 1990 Sep;8(9):1602]". *Journal of Clinical Oncology*, Vol. 7, No. 11, pp. 1630–1636, November 1989.

[Lowe 99] D. Lowe. "Object recognition from local scale-invariant features". In: *Computer Vision (ICCV), IEEE International Conference on*, pp. 1150–1157, Kerkyra, Corfu, Greece, September 1999.

[McLo 92] T. C. McLoud, P. M. Bourgouin, R. W. Greenberg, J. P. Kosiuk, P. A. Templeton, J. A. Shepard, E. H. Moore, J. C. Wain, D. J. Mathisen, and H. C. Grillo. "Bronchogenic carcinoma: analysis of staging in the mediastinum with CT by correlative lymph node mapping and sampling". *Radiology*, Vol. 182, No. 2, pp. 319–323, February 1992.

[Mlle 07] M. Möller and M. Sintek. "A Generic Framework for Semantic Medical Image Retrieval". In: *Proceedings of the 1st International Workshop on Knowledge Acquisition from Multimedia Content KAMC'07, CEUR Workshop Proceedings 253*, Genoa, Italy, December 2007.

[Papp 04] C. Pappone, H. Oral, V. Santinelli, G. Vicedomini, C. C. Lang, F. Manguso, L. Torracca, S. Benussi, O. Alfieri, R. Hong, W. Lau, K. Hirata, N. Shikuma,

B. Hall, and F. Morady. "Atrio-Esophageal Fistula as a Complication of Percutaneous Transcatheter Ablation of Atrial Fibrillation". *Circulation*, Vol. 109, No. 22, pp. 2724–2726, June 2004.

[Rous 06] M. Rousson, Y. Bai, C. Xu, and F. Sauer. "Probabilistic minimal path for automated esophagus segmentation". In: *SPIE Medical imaging*, pp. 1361–1369, San Diego, CA, USA, February 2006.

[Rtsc 01] G. Rätsch, T. Onoda, and K.-R. Müller. "Soft Margins for AdaBoost". *Machine Learning*, Vol. 42, No. 3, pp. 287–320, March 2001.

[Scha 07] M. Schaap, I. Smal, C. Metz, T. van Walsum, and W. Niessen. "Bayesian tracking of elongated structures in 3D images". In: *International Conference on Information Processing in Medical Imaging (IPMI), Lecture Notes in Computer Science, LNCS 4584*, pp. 74–85, Kerkrade, The Netherlands, July 2007.

[Scha 90] R. E. Schapire. "The Strength of Weak Learnability". *Machine Learning*, Vol. 5, No. 2, pp. 197–227, June 1990.

[Scov 07] P. Scovanner, S. Ali, and M. Shah. "A 3-dimensional sift descriptor and its application to action recognition". In: *Proceedings of the 15th international conference on Multimedia*, pp. 357–360, ACM, New York, NY, USA, September 2007.

[Seif 09] S. Seifert, A. Barbu, S. K. Zhou, D. Liu, J. Feulner, M. Huber, M. Suehling, A. Cavallaro, and D. Comaniciu. "Hierarchical parsing and semantic navigation of full body CT data". In: *SPIE Medical Imaging*, p. 725902, Orlando, Florida, USA, February 2009.

[Serv 03] R. A. Servedio. "Smooth boosting and learning with malicious noise". *Journal of Machine Learning Research*, Vol. 4, pp. 633–648, December 2003.

[Sino 07] A. Sinop and L. Grady. "A Seeded Image Segmentation Framework Unifying Graph Cuts And Random Walker Which Yields A New Algorithm". In: *Computer Vision (ICCV), IEEE International Conference on*, pp. 1 –8, Rio de Janeiro, Brazil, October 2007.

[Slab 05] G. Slabaugh and G. Unal. "Graph cuts segmentation using an elliptical shape prior". In: *Image Processing (ICIP), IEEE International Conference on*, pp. II – 1222–5, Genoa, Italy, September 2005.

[Soma 06] *Somatom Sensation 10/16 Application Guide*. Siemens medical, 2006.

[Swai 91] M. J. Swain and D. H. Ballard. "Color indexing". *International Journal of Computer Vision*, Vol. 7, No. 1, pp. 11–32, November 1991.

[Tu 05] Z. Tu. "Probabilistic boosting-tree: learning discriminative models for classification, recognition, and clustering". In: *Computer Vision (ICCV), IEEE International Conference on*, pp. 1589–1596, Beijing, China, October 2005.

[Tu 06] Z. Tu, X. Zhou, L. Bogoni, A. Barbu, and D. Comaniciu. "Probabilistic 3D Polyp Detection in CT Images: The Role of Sample Alignment". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pp. 1544 – 1551, New York, NY, USA, June 2006.

[Vali 84] L. Valiant. "A Theory of the Learnable". *Communications of the ACM*, Vol. 27, No. 11, pp. 1134–1142, November 1984.

[Veks 08] O. Veksler. "Star Shape Prior for Graph-Cut Image Segmentation". In: *European Conference on Computer Vision (ECCV), Lecture Notes in Computer Science, LNCS 5304*, pp. 454–467, Marseille, France, October 2008.

[Viol 01] P. Viola and M. Jones. "Rapid Object Detection using a Boosted Cascade of Simple Features". In: *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pp. 511 – 518, Kauai, HI, USA, December 2001.

[Vita 09] D. Vitanovski, R. Ionasec, B. Georgescu, M. Huber, A. Taylor, J. Hornegger, and D. Comaniciu. "Personalized pulmonary trunk modeling for intervention planning and valve assessment estimated from CT data". In: *MICCAI (1), Lecture Notes in Computer Science, LNCS 5761*, pp. 17–25, London, UK, September 2009.

[Wang 01] S. Wang and J. Siskind. "Image segmentation with minimum mean cut". In: *Computer Vision (ICCV), IEEE International Conference on*, pp. 517 –524, Vancouver, Canada, July 2001.

[Warw 58] R. Warwick and P. L. Williams. *Gray's anatomy (Thirty-fifth ed.)*. Longman, London, UK, 1858.

[Wels 09] M. Wels, Y. Zheng, G. Carneiro, M. Huber, J. Hornegger, and D. Comaniciu. "Fast and Robust 3-D MRI Brain Structure Segmentation". In: *MICCAI (2), Lecture Notes in Computer Science, LNCS 5762*, pp. 575–583, London, UK, September 2009.

[Zhen 07] Y. Zheng, A. Barbu, B. Georgescu, M. Scheuering, and D. Comaniciu. "Fast Automatic Heart Chamber Segmentation from 3D CT Data Using Marginal Space Learning and Steerable Features". In: *Computer Vision (ICCV), IEEE International Conference on*, pp. 1–8, Rio de Janeiro, Brazil, October 2007.

[Zhu 09] X. Zhu and A. B. Goldberg. *Introduction to Semi-Supervised Learning*. Vol. 3 of *Synthesis Lectures on Artificial Intelligence and Machine Learning*, Morgan & Claypool Publishers, San Rafael, CA, USA, 2009.

[Zill 09] S. Zillner and M. Huber. "Intelligent Medical Image Search". *it-Information Technology*, Vol. 51, No. 5, pp. 285–293, September 2009.

Index