Institut für Signalverarbeitung und Prozessrechentechnik
Universität zu Lübeck
Direktor: Prof. Dr.-Ing. Alfred Mertins

Active Contours with Spatially-Variant Definitions of Energy Terms Based on Local Region Descriptors

Inauguraldissertation
zur Erlangung der Doktorwürde (Dr. rer. nat.)
der Universität zu Lübeck
- Aus der Technisch-Naturwissenschaftlichen Fakultät -

vorgelegt von
Dipl.-Wirt.-Inf. Cristina Darolti

Lübeck, November 2008

Cristina Darolti
Institut für Signalverarbeitung
Universität zu Lübeck

Prüfungskomission

Erstberichterstatter:
Prof. (apl) Dr. rer. nat. Ulrich G. Hofmann

Zweitberichterstatter:
PD Dr.-Ing. Erhardt Barth

Prüfungsvorsitzender:
Prof. Dr. rer. nat. Thorsten Buzug

Active Contours with Spatially-Variant Definitions of Energy Terms Based on Local Region Descriptors

Determining the contents of an image is a fundamental task in computer vision. Consequently, the problem of separating an image into distinct, contiguous regions - the problem of image segmentation - has attracted the efforts of numerous researchers. A well-known framework in the field of image segmentation is the active contours framework. This thesis introduces region-based active contours incorporating two novel ideas.

First, the thesis proposes to drive contours by local region descriptors, which means that the statistics of features are computed from observations in localized windows. Moreover, the energy of active contours based on local region descriptors is constructed such that the number of its local minima is reduced, by assuming that features in local regions follow a normal distribution.

Second the thesis proposes a solution for situations in which local regions are badly modeled using Gaussian probability densities. For this purpose, the thesis introduces active contours with spatially variant definitions of energy terms. As a result, the energy of an active contour is based on a Gaussian density model in some local regions, while, in other local regions, the energy is based on densities estimated with a kernel density method. The kernel density method is used when multiple contours are in one another's vicinity when local regions are observed. This event is assumed to occur in the vicinity of an image boundary, where arbitrary densities of multiple regions must be well modeled. The properties of both proposed ideas are demonstrated in experiments with synthetic and natural images.

The novel active contours are applied to the segmentation of natural, textured images and to the marker-free tracking of hand motion with a multiple-camera computer-vision system. In the image segmentation application, we employ a novel, but time intensive, texture modeling framework. Texture is modeled by applying nonparametric density estimation methods to observed image patches, which are regarded as high-dimensional feature vectors. Combining local descriptors and spatially-variant definitions of energy terms, we obtain active contours for textured image segmentation that are two orders of magnitude faster than their global counterparts. The result of segmenting complex natural images with this method are very good.

Hand motion tracking is accomplished using local region descriptors. Due to very accurate segmentation results, the middle of the arm and of the wrist can be reliably computed, yielding good features to track. By tracking the motion of these features with a stereo camera system, the motion of hands can be measured. The prototypical hand measurement system is tested by measuring the surgical motion of a few surgeons and trainees.

Aktive Konturen mit ortsabhängigen Definitionen der Energieterme basierend auf lokalen Bildregionsdeskriptoren

Eine wichtige Aufgabe im Bereich der *Computer Vision* ist die automatisierte Bestimmung von Bildinhalten. Diese Aufgabe kann, wenn überhaupt, nur durch eine Abfolge von komplexen Prozeduren gelöst werden, deren primäre wiederum die Bildsegmentierung ist. Bildsegmentierung - also der Teilung eines Bildes in disjunkte, kontinuierliche Regionen - ist ein durchaus vielbeforschtes Gebiet, das unter anderen die Methode der *Aktiven Konturen* hervorgebracht hat. Ziel der vorliegenden Arbeit ist es, diese Methode derart zu verbessern, dass natürliche Bilder und Videoaufnahmen von Handbewegungen möglichst fehlerarm segmentiert werden konnten.

Als erster Ansatz werden dazu Aktive Konturen mit Hilfe lokaler Bildregionsdeskriptoren entwickelt, d.h. die auf die Kontur wirkenden Kräfte werden aus Merkmalsstatistiken generiert, welche nur über lokalen Fenstern einer Bildregion berechnet werden. Darüber hinaus wird bei der Definition der Energie dieser Aktiven Konturen auf die Anzahl der lokalen Energieminima in der Bildregion geachtet. Um diese Zahl zu minimieren, wird angenommen, dass die Merkmale in lokalen Bildregionen statistisch normal verteilt sind.

Der zweite Ansatz behandelt jene Fälle, in denen die Modellierung der deskriptiven Wahrscheinlichkeitsverteilungen der Merkmale in lokalen Bildregionen mit Hilfe Gaussscher Modelle eine schlechte Approximation liefert. Dieser Ansatz basiert darauf, dass die Energie der Aktiven Konturen abhängig von der Konturposition relativ zu anderen Konturen im Bild zu definieren ist, d.h. es müssen Aktive Konturen mit ortsabhängigen Definitionen der Energieterme entwickelt werden. Wenn eine Kontur in einem lokalen Fenster keine Nachbarkonturen hat, basiert wiederum deren Konturenergie auf einer Gaussschen Wahrscheinlichkeitsdichte. Sind mehrere Konturen in einem lokalen Fenster benachbart, dann basiert die Konturenergie auf einer Wahrscheinlichkeitsdichte, welche mit einem Dichteschätzer approximiert wird. Es gilt dabei die Annahme, dass sich mehrere benachbarte Konturen sich in der Umgebung einer Objektgrenze nähern. An dieser Stelle müssen die - ansonsten beliebig geformten - Wahrscheinlichkeitsdichten unterschiedlicher Objekte gut approximiert sein. Die Eigenschaften der Konturen, welche aus diesen Ideen folgen, werden in Experimenten mit synthetischen und natürlichen Bildern demonstriert.

Die entwickelten Aktiven Konturen werden angewendet um texturierte, natürliche Bilder zu segmentieren, Die Bildsegmentierungsapplikation benutzt eine neue Methode der Texturmodellierung. Kleine Bildausschnitte werden als Texturbeispiele betrachtet, welche in einem sehr hoch-dimensionalen Raum eingebettet werden. Deren Wahrscheinlichkeitsdichten werden mit nichtparametrischen Dichteschätzern approximiert. Durch die Zusammenführung lokaler Bildregionsdeskriptoren, ortsabhängiger Definitionen der Energieterme und ausschnittsbasierter Texturmodellierung werden Aktive Konturen für die Textursegmentierung konstruiert. Diese Aktiven Konturen werden um zwei Grössenordnungen schneller berechnet, als Aktive Konturen, welche die globale Statistik der Texturbeispiele approximiren. Die präsentierten Aktiven Konturen führen zu sehr guten Ergebnissen in der Segmentierung natürlicher Bilder.

Das markerfreie Tracking von Handbewegungen mit einem Multi-Kamera Computer Vision Aufbau wird im Weiteren durch die Anwendung lokaler Bildregionsdeskriptoren erreicht. Dank der sehr akkuraten Segmentierergebnisse, können Handmerkmale, wie die Armmitte und die Handgelenksmitte, zuverlässig berechnet und verfolgt werden. Die 3D-Bewegung dieser Merkmale kann schliesslich mit dem Stereokameraaufbau vermessen werden. Um den Aufbauprototyp zu testen, werden Handbewegungen einiger ausgebildeter und angehender Chirurgen vermessen.

Acknowledgements

Three sine-qua-non ingredients for this thesis were provided to me by:

Uli, who gave me the chance to do this work when I was in a delicate situation,

Erhardt, who considerably improved my writing,

Christoph, who provided scientific comments, discussions and much help in all the right moments.

My thanks go out to Alfred and Heiko for valuable reviews of my texts.

Prof. Buzug was very kind when accepting to serve on my thesis committee, for which I thank him.

My thanks also go out to my mother for the best schnitzels, meatballs and “vinete” in the quadrant, and to my grandmother, for being a monument of common sense. (Köszi anyá es mama!).

I should mention Miki, Paula (mulţam copii!) and Cello (obrigada!) who are very present in my life, and would probably be upset if they were to somehow find out they were not mentioned. :)

To Christoph

Contents

1. Introduction

> These are the kind of disciplines in the field of science where you have to learn to know when you know and you don't know and what it is you know and what it is you don't know. You got to be very careful not to confuse yourself.
>
> *Richard Feynmann*

This thesis proposes novel active contour methods and applies them to the segmentation of natural textured images and in a computer vision system for hand motion measurement.

Computer (or machine) vision is a research field reuniting a shoal of researches that pursue the ultimate goal of making machines see. Having found the time and the willingness to read this thesis, as you do so, your biological vision system is transparently and instantaneously solving some very complex engineering problems. These involve recording an image of the page, separating the black foreground containing message from the white background that is only its support, identifying word-spacings in the foreground, and identifying the letters, in order to know where to expect words and finally read them. In computer vision, these problems are known as image formation and representation, image segmentation, stereo vision and pattern recognition, and state-of-the-art solutions engineered for them are not comparable with biological solutions.

Image formation and representation is dictated by the physical device involved in the process. Machine hardware is clearly in advantage: it can record the world in much more detail than the human eye[1], concerning space, frequency and time. At extremes, cameras can record signals from millions of kilometers away, in the whole spectrum of electromagnetic waves, every few milliseconds. The recorded images may be played forward, backward or paused in time, at very high or low frame rates (allowing television viewers around the world to awe at the flight of a colibri bird on a Friday evening). A machine vision system can be equipped with many cameras that can be flexibly positioned, the whole system being limited only by the financial resources of its constructor. The advantages of machine vision systems end here.

Cameras cannot automatically adapt to the intensity of light as well as eyes do every time a person goes outdoor. Algorithms for segmentation, stereo computation and pattern recognition

[1] Biological eyes are very different in their properties; for example, some snakes have detectors for infrared wavelengths, while eagles can see three times farther than humans. For this reason, we limit our comparative discussion to human vision. After all, it is mainly the type of vision researchers try to imitate.

are rudimental in comparison. Segmenting black symbols from a white page is a relatively simple task [2]. Consider the far more complex task of picking this manuscript from a cluttered desk instead of any of the competing works. The human vision system identifies the manuscript within seconds, but state-of-the-art vision algorithms could not, even if one had a week's time to wait for the answer.

The difficulties start with the digital representation of an image. As Steven Pinker has put it (Pinker, 1999), "If you could see the world through a robot's eyes, it would look not like a movie picture decorated with crosshairs but something like this:"

157 156 167 167 161 167 181 192 176 159 166 164 160 171 166 163 175 187 184 177 163
161 159 167 170 165 161 168 198 182 158 170 171 161 175 169 160 168 182 187 188 178
164 158 156 165 169 163 150 192 185 161 169 169 162 178 169 152 151 162 172 184 184
158 150 141 158 176 173 144 180 182 163 163 163 164 178 179 157 147 153 163 181 189
152 142 130 155 185 182 151 167 174 166 159 164 175 185 195 171 158 159 165 181 189
150 138 126 156 190 186 159 160 168 168 161 172 191 197 200 177 165 163 165 176 181
139 130 129 154 180 181 167 165 159 154 161 179 197 205 204 171 160 169 166 163 172
143 130 120 136 158 161 163 167 164 151 146 159 181 196 200 182 174 171 164 169 175
148 136 120 128 146 148 157 168 167 148 130 134 157 177 186 183 183 174 165 175 182
150 147 138 145 157 156 155 168 170 151 129 127 142 157 167 166 175 177 169 176 186
150 158 160 169 175 167 161 171 174 163 149 144 147 150 159 149 162 178 172 168 178
149 163 171 178 179 168 168 175 179 177 175 173 167 160 166 149 158 176 168 157 166
152 166 171 175 176 167 166 172 178 183 189 192 185 176 176 163 165 169 159 156 162
154 169 169 170 174 169 160 166 174 181 190 197 194 186 180 175 173 163 153 160 166
169 156 156 174 171 177 162 162 166 179 193 200 198 192 191 184 174 165 159 158 160
157 147 152 174 175 184 167 167 169 178 190 198 200 199 191 186 177 170 165 165 167
158 144 144 164 164 172 169 168 169 172 179 188 198 203 194 190 184 178 174 173 174
166 146 141 156 152 159 167 166 164 162 164 174 188 199 198 196 191 186 181 178 176

The numbers code for pixel intensities in a black and white image. The philosophy of any image processing algorithm is to make sense of the numbers, while accessing their spatial arrangement only locally. As an exercise, one can zoom into an image till pixels get very large, like in Fig. 1.1, and try to understand the structure of the image looking only through this magnifying window. The type of object shown in Fig. 1.1 is encountered extremely often by each person every day.

Image segmentation starts at the level of individual pixels. This is equivalent to finding the eye while seeing only little parts of the image (smaller than the image on the right). The task, which was trivial in the left image, suddenly becomes annoyingly difficult. To make the task even more difficult, even when the entire image is available, different human vision systems find different boundaries. In Fig.1.2 lighter boundaries in the contours-image represent boundaries that have been drawn by many test persons that observed the image on the left. The gray boundaries are

[2] There exist many systems that can scan a page and convert it into editable text

Figure 1.1.: One of the eyes of the person on the right, extremely magnified.

ones drawn by only a few, or one test person. The contours-image is not binary. The problems this creates relate to the fact that computer vision algorithms aim to imitate biological systems, which is rather difficult when different biological systems behave differently.

Figure 1.2.: An image and its segmentation by several different people (Bear, 2008). Note the discrepancies in the segmentation as depicted by the lines of different intensity.

Image segmentation refers to the separation of images into distinct, contiguous regions. It is an important part of computer vision, and it is an extensively studied problem. Accordingly, numerous approaches have been proposed. They all make use of two basic properties of images of real world objects: 1) spatial proximity and similarity between points on an object's surface, and 2) discontinuity between the surfaces of two objects. Nonetheless, there are various possible ways of organizing an overview of methods for image segmentation; two different examples can be found in (Forsyth and Ponce, 2003) and (Snyder and Qi, 2004). In the following, we present segmentation methods starting with simple ones and finishing with complex ones.

Thresholding is a simple and intuitive method that defines or computes the lowest and the highest intensity that occur in an object. Only pixels with intensities that do no fall between the two thresholds do not belong to the object. Thresholds can be selected automatically, based on knowledge about the system. Different types of knowledge or assumptions lead to different procedures for automatic selection, reviewed, for example, in (Jain et al., 1995). Considering

that the world can be imaged using only 256 different intensities, it is obvious that this method can only segment trivial images. Take the situation of a simple shaded scene (a black cube on a white board illuminated from one side). A single threshold will most probably not be appropriate throughout the image. The problem is remedied if we subdivide the image, and adapt the value of the threshold to each subimage. Adaptive thresholds are more robust when they are iteratively varied while verifying that the new segmentation is better than the previous. Even with these improvements, thresholding is limited to fairly simple images. The main reasons are that more objects have similar intensities in basic natural scenes, like the one in Fig.1.3 and that the spatial information is totally disregarded.

It is fairly easy to combine spatial information with thresholding in methods known as *clustering methods*. Starting with some image points, these methods cluster pixels that belong together using mainly three approaches.

Figure 1.3.: An image showing some of nature's basics (Flickr, 2008).

Region growing starts with a few pixels (selected automatically or manually) that will assimilate other pixels as long as some similarity criterion is larger than a threshold. For instance, to attempt segmentation in Fig.1.3, a set of seed points is placed on the dog and a set on the cat. The first set grows while the difference between light pixels inside the region and light pixels outside the region is smaller than a threshold. The second will behave similarly, but taking dark pixels.

Region merging starts by making each pixel a cluster. In an iterative procedure, the two clusters with the smallest inter-cluster distance are merged, until some satisfactory clustering has been reached. *Region splitting* starts with a single cluster that contains all pixels. Also iteratively, the algorithm splits the cluster that results in two clusters with the largest inter-cluster distance.

Region growing, merging and splitting depend on issues with complex answers. How should thresholds be determined? What is a good way to measure the distance between clusters? Many methods have been designed to address these issues. They include, for example, k-means clustering and statistical classification. Other methods include trees and graphs. For example, quadtrees are constructed by recursively splitting the image and arranging the pieces in white, black and gray nodes. When all pixels in a node are black or white, the node is not split further. Only gray nodes are processed in each iteration and are split into four equal regions, until all pixels are black or white, and thus a segmentation is obtained. Graphs are obtained by viewing clusters as nodes. The edges connecting the nodes are weighted according to similarity measures and connectivity (which clusters neighbor any given cluster) between clusters. The best segmentation is equivalent to finding a minimal length path or a maximum flow in the graph. Graph methods largely benefit from well developed graph algorithms. We turn from methods that exploit similarity and spatial proximity, to methods that exploit discontinuities between objects.

Discontinuities are usually perceived at locations where sudden changes in intensity, called edges, occur. Many methods for finding edges, called *edges detectors*, exist. Most edge detectors define edges via the image derivative. For example, the magnitude of the gradient at a pixel must be larger than some threshold for a pixel to qualify as edge-pixel. Looking at an image one may be fooled into thinking that most edges are similar to stair steps. However, they are similar, if at all, to ramps or roofs with very different slope magnitudes.

Ideal edges

In reality edges look like noisy signals. Edge detectors thus often find edges where we do not see them and miss some that are obvious to us. To illustrate this behavior, a Sobel edge detector has been applied to the image in Fig.1.4. The result can be visually improved if the spatial proximity between edge-pixels is additionally considered, as does, for example, the Canny method, but the main issue still remains.

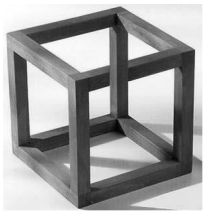
Real edge

original image Sobel edges Canny edges

Figure 1.4.: An original image (Flickr, 2008) and its edge images.

Another class of methods that involves image gradients and suffers from the same problem is *morphological watershed*. For a textbook overview of edge detectors and morphological watersheds see, for example, (Gonzalez and Woods, 2002a). The image is visualized as a basin in 3D, which has points of maximum depth, to which walls with slopes equal to those of image gradients converge. The basin is flooded with water from below. The crests where water fronts meet are discontinuities in the image. Watershed methods typically lead to over-segmentation, i.e. too many different regions are found in the image.

In fact, most images are too complex to be segmented based only on the information available in intensity or color data. Cognitive psychology provides evidence that the human visual system is preconditioned to see objects that it already knows. Experiments on visual perception include images of so-called *impossible objects* like the impossible cube, first published by the Swiss crystallographer Louis Albert Necker in 1832. We see that there is something wrong with the cube in this figure, but we still see a cube because it fits most of the general knowledge about the cube. The objects most eagerly recognized by the human visual system are faces. An intuitive proof is that we remember faces much easier than we remember names. Using magnet resonance imaging, researchers have shown that there is special sites and circuitry in the brain dedicated to recognizing faces,



Impossible cube

(Moeller et al., 2008; Tsao and Livingstone, 2008) are recent examples. The brain even seems to constantly search for faces. When one site is artificially stimulated, the others also show increased activity. This should be no surprise, since we are all familiar with the man in the moon, or the smiley in the clouds. The effect in Fig.1.5 is due, at least partly, to this constant search for faces.

Angels and daemons (Wade and Tavris, 2005) Double meaning scene (Gillbert, 1929)

Figure 1.5.: Images in which we see faces.

Computer vision algorithms that use knowledge of the shape and appearance of an object, or that make assumptions about them, are more robust than those based solely on image data. For example, the eyes of the dog in Fig.1.4 can be found by searching for round shapes in the edge image.

Many objects have much more complex shapes than lines, circles and corners. A correspondingly large amount of thinking is necessary in order to describe these shapes. The first tools that come to mind involve statistical learning. In the active shape models framework (Cootes et al., 1995), the shape of an object type is learned from examples, by learning an average shape and the amount by which it varies from examples. A segmentation algorithm uses the learned shape to search for an instance of the shape, for example, in an edge image or an image labeled by clustering. Learning algorithms can also be applied to appearance, not only to shape. Additionally, appearance must not refer to the whole object. For example, attempts have been made to learn what an edge looks like, in general (Martin et al., 2004). Learning from examples involves complex statistical modeling and methods and it cannot guarantee the success of segmentation.

A very important issue in shape learning is that 3D objects - that is the majority of objects in the world, have a different *projected* 2D shape, when viewed from different angles. It is not possible to learn all these shapes, since, in the majority of situations, the process would largely

exceed the computation power available at present. Many methods for model-based vision have been proposed (see, for example, (Forsyth and Ponce, 2003)), but this dissertation focuses on a different method for incorporating knowledge about the image into the segmentation process.

A shape model helps segment the object in more complex, cluttered images. When we know what to look for, the pile can be large, and we will still find the item. But what if we do not know what the scene is, that has been photographed? It would not be practicable to iterate through a list of objects known by the system for practical reasons regarding computation time. Also, it would be difficult to decide which and how many objects to put on the list. Even so, this does not mean that we do not know or cannot assume anything about the image. Our brains constantly make assumptions about the world we live in. One of them is that objects have smooth boundaries, that can at times take sharp corners and that the largest changes in visual properties occur around the boundaries. Take trees for instance. They come in all sorts of shapes, sizes and colors, with hulls that are frayed when observed exactly, but on the whole they create smooth shapes. None of them grow to have boundaries composed of randomly connected, zigzagged straight and circular edges, like in the figure that look like clouds from the middle up just to confuse us.

An impossible zigzagged tree with impossible texture

The smoothness assumption is incorporated into all methods of a particular class known as *active contours*, to which the methods of this dissertation belong. Active contours can be thought of as rubber-bands placed on the image. A rubber-band is stretched toward image boundaries by forces generated from the image. Active contours can use both edge detectors and methods that cluster image pixels based on the statistical analysis of the variation of pixel values in order to stretch the rubber-band (i.e. to evolve the contour). Each rubber-band has an energy associated with it, and its deformation stops only when this energy is minimal. The result is a set of regions with smooth boundaries and constant appearance. This is not to say that active contours can correctly segment all images. This is mainly because human perception of "smooth" boundaries and "constant" appearance is very flexible, and cannot be pressed into one mathematical model. Active contours model image segmentation as a mathematical problem and use calculus of variations to solve it. However, active contour methods have been successfully employed in a large number of applications, ranging from vehicle tracking in traffic to medical image segmentation.

Active contours have been intensively studied. Proposed methods differ in the way they generate forces from the image and the manner they formulate the smoothness term. The proposed methods are very promising, as the large number of applications shows, but they are also far from constituting a robust segmentation tool for all input images. There is thus still room for research to improve on these methods, and explore novel alternatives. For example, a very recent development includes active contours that evolve driven by forces generated only from local information, as will be discussed in Chapter 4.

In this dissertation, we present local region descriptors as a new concept for evolving contours based on statistics of local regions (i.e. ones that cover small parts of the image), and introduce energies constructed from local descriptors such that the number of local minima is reduced. In natural images, the variation of features in local regions is smaller than in global regions that cover entire image objects. Local regions are thus more likely to be well described by simpler statistical models than global regions. For the same reason, the probability densities of different local regions are less likely to overlap than those of global regions are.

We also present an entirely novel type of active contours that have spatially variant definitions of energy terms. For these contours the way forces are generated from the image depends on the position of the contour point on which they act. For all other contours, the definition of energy terms, and thus the definition of forces that act upon the contour, is spatially invariant. Spatially-variant energy terms have the advantage that the associated contour has a flexible behavior which adapts to the presence of other contours, and, indirectly, to the image data.

The novel techniques presented in this dissertation have been combined to create solutions for two computer vision problems. First, the spatially variant definition of energy terms was based on local information obtained from texture. The resulting active contours significantly improve the speed of a recently introduced texture segmentation framework. The segmentation framework is based on modeling image patches as Markov random fields to capture the properties of texture, and use them to segment images with active contours. Due to the fact that it uses local texture information, the presented texture segmentation method is two orders of magnitude faster than similar active contours based on global information.

Second, active contours segmentation based on local information enable the creation of a multicamera computer vision system for tracking and measuring hand motion. Hands are difficult items to track because they are very flexible causing the shape they project on a 2D image to change extremely. For accurately and robustly tracking the motion of hands, computer vision systems typically require the use of visual markers. Much research on hand tracking involves using a 3D hand model. The system presented in this dissertation uses neither. Because the segmentation method yields very accurate hand contours, exact hand features are computed to replace visual markers. Unlike markers, the computed features are visible from every camera perspective, the time is not lost with fixing the marker apparel on the hands. Economizing time proves to be very valuable in measuring the hand motion of surgeons during surgical procedures. Surgeons are well known for constantly being short of time.

The methods proposed here have a large range of applications. Besides medical image processing, that constantly requires better and faster algorithms, the proposed methods would be particularly suited as an object cutting tool for editing photographs.

1.1. Thesis Overview

The methods in this thesis build on the statistical analysis of region properties and active contours. Two chapters are thus dedicated to them. Chapter 2 reviews three main approaches to image modeling in computer vision, of which this thesis adopts the statistical approach. Fundamental concepts in probability theory and statistics needed for this approach are present in the remainder of the chapter. These include concise definitions and the mathematical notation for random variables and probability density, density estimation and Markov random fields. Nonparametric density estimation is discussed in more detail in order to give a better understanding of the trade-offs involved and because it is an important building block of this thesis. The chapter closes with a review of divergence measures for measuring the difference between two probability densities.

Chapter 3 describes the active contours framework as the combination of region and edge properties for describing objects, a method for generating motion forces from the selected properties, a method for implementing contour motion and a choice of initial contours. Extensive reviews of the active contours literature explain the possible choices for each component. This part of the dissertation is organized in four corresponding subsections.

The next two chapters present the core contributions of this dissertation to the theory of active contours. First, chapter 4 proposes to describe the properties of regions locally and thus generate motion forces from the local information. Similarly to statistical descriptors, the proposed construction is named local region descriptors. Local region descriptors are contrasted to global region descriptors. The chapter then discusses the fact that energies of active contours constructed with local region descriptors have many local minima, and it proposes solutions to this problem. Properties of active contours based on local region descriptors are pointed out in numerous segmentations of natural images.

Second, chapter 5 presents active contours with spatially-variant definitions of energy terms, and opposes these to other active contours, that typically have spatially invariant definition of energy terms. The general method is concretized in contours that have the data terms defined in one of two ways. The purpose of this particular contour is to further reduce the number of minima of active contours based on local information and thus make the proposed active contours more robust, as will be explained in the chapter.

Chapters 6 and 7 apply the proposed theoretical methods to two different types of problems. The first one in chapter 6 refers to the issue of segmenting textured images. Specifically, small image patches are modeled as Markov random fields in order to describe the properties of textures, which is a relatively recent development in texture segmentation. We review texture segmentation and show that this concept has been recently employed with active contours based on global information. The chapter demonstrates that using local information, these active contours can be modified such that the method's speed increases by two orders of magnitude. Using spatially

variant definitions of energy terms, the active contours are also robust.

Chapter 7 employs the methods of previous chapters in a multicamera computer vision for hand tracking. This chapter describes the problem of tracking hands using computer vision and presents a short overview of related work. It then introduces the system's setup and the basic principles of stereo vision needed for measuring the positions of points in 3D space. Accurate tracking of hand and arm contours provides the 2D points necessary for measuring 3D points. The practical use of the system is demonstrated in the tracking and measurement of hands during surgical procedures, which yields practical 3D measurements.

The work concludes with Chapter 8, which also discusses the ideas that remain unexplored in the course of this dissertation and touches a few directions for future research.

2. Image Models and Statistics Theory

> I had a conversation some weeks ago about flying saucer with *laymen*. 'Cause I am scientific, I know all about flying saucers. I said, I don't think there are flying saucers. [..] That's the way it is scientific: it is scientific only to say what's more likely and less likely and not to be proving all the time possible-impossible. [..] I finally said to him: Listen [..], I think that it is much more likely that the reports of flying saucer are the results of known irrational characteristics of terrestrial intelligence, rather than the unknown rational efforts of extraterrestrial intelligence.
>
> *Richard Feynman at Cornel University, 1964.*

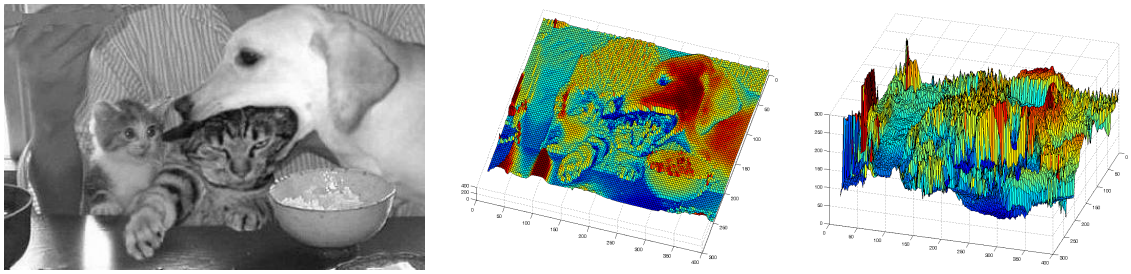
This thesis is concerned with the segmentation of images into regions that have a more or less uniform appearance. As was pointed out in the introduction, the segmentation task starts with a matrix of integers. In order to delineate regions in this matrix, the intuitive concept of uniform appearance must be shaped into a model.

2.1. Image Models

Researchers have taken different approaches for this problem. A short overview of three main approaches, i.e. functions on a grid domain, multiscale representation and statistical models, follows. The statistical model is the one on which this thesis is based, therefore relevant technical background from probability, statistics and information theory will also be shortly reviewed.

2.1.1. Images: Functions on a Grid Domain

The observed image is interpreted as an intensity function sampled on a grid, most often of square pixels. The observed intensity function is a noisy version of a piecewise smooth function $u(x,y)$, $I(x,y) = u(x,y) + n(x,y)$, with $u(x,y)$ the ideal smooth function and $n(x,y)$ the noise, as exemplary depicted in Fig.2.1. The piecewise smooth function is discontinuous only at object boundaries. To impose the smoothness constraint on the ideal function, one requires that some measure depending on the function's gradient be minimal. For instance, in (Mumford and Shah,



(a) (b) (c)

Figure 2.1.: Functions-on-a-grid-domain representation of an inexplicable image: (a) the original image $I(x,y)$ (Flickr, 2008), (b) and (c) two views of the surface described by the function $I(x,y) = u(x,y) + n(x,y)$. Note that in view (b) we can still recognize the image's structure, but by changing the perspective like in the view (c) we cannot, partly due to the fact that textured objects do not yield smooth functions.

1985) the term $|\nabla u|^2$ must be minimal. Minimizing the bounded total variation $|\nabla u|$ of the ideal function was proposed in (Rudin et al., 1992), in the sense that, on the same image, L_1-norm denoised approximations of the observed image look better than L_2-norm denoised approximations. The proposed model is known as the ROF-model, and it can be intuitively understood by thinking of the image as a 3D-surface which is propagated in the direction of its normals with speed proportional to its curvature.

Variational methods that employ this model use numerical iterative schemes to determine the piecewise constant function, and they can solve the tasks of denoising and segmentation simultaneously. However, the optimization problems that need to be solved are non-linear, and have a large number of local minima, which is a problem especially for the segmentation task. This issue can be alleviated by imposing strict constraints on the image. For example, a global minimum can be reached, independent of initialization, when the number of image objects is very small (Bresson et al., 2007).

Not all objects in natural images can be represented by smooth functions. An example are textured objects, as for example the cat in Fig. 2.1. These objects are very difficult to model as smooth functions of a grid domain. Attempts have been made to model texture as noise (Bresson et al., 2007) or as an oscillatory image part in a combination of cartoon-texture models (Bertalmio et al., 2003). Complex texture can be described much better with statistical models, since these are better equiped to deal with different variations of region features.

2.1.2. Images: Multiscale and Multiresolution Representation

The digital image is a matrix of numbers, as mentioned in the previous chapter. Fast image operators, e.g. image filters, analyze neighbored individual pixels, yielding information about local features at the finest scale of the image. Information about most image structures, like objects, is found when observing much larger patches of pixels at larger scales (Freeman and Adelson, 1991). The fundamental problem in multiscale analysis is to determine the scale which best reveals a certain structure, since, in general, structures of different sizes are best visible at different scales; for a textbook discussion see, for example, (Jaehne, 1997). For example, if a person's head and body are best identified at a larger scale, his eyes and hands are best identified at a lower scale. Multiscale methods thus analyze images at many different scales to determine the dominant scale in a local neighborhood. In this process, the images at larger scales are increasingly blurred versions of the image at a fine scale.

Two main directions of research that look at the image scale problem from different points of view cover most of multiscale analysis: wavelets and scale space methods. In scale space methods, for which (Lindeberg, 1998a,b) are often cited sources, images are blurred by applying the heat-diffusion equation to the image in a given period of time. This process is equivalent to filtering the image with Gaussian filters of increasing size, and the scale parameter is proportional to the standard deviation of the Gaussian kernel. An example of scale space for an image showing protesting cows is shown in Fig.2.2.

The scale parameter adds a dimension to the image representation that requires large amounts of memory, and many of the pixels at larger scales do not contain relevant information. A solution to both issues is the multiresolution representation, such that images at larger scales have less pixels than those at finer scales. The Gaussian pyramid is a standard multiresolution representation that shows only the coarsest structures in the image at the top of the pyramid. Each level is four times smaller than its predecessor (it is obtained by taking every second pixel in every second row of the smoothed predecessor). An example of a Gaussian pyramid is shown in Fig. 2.3. One can avoid loosing image details when constructing the Gaussian pyramid, with a Laplacian pyramid, see, for example, (Gonzalez and Woods, 2002b). The latter is obtained by subtracting two consecutive planes of the Gaussian pyramid.

Wavelet transforms (Mallat, 1999; Gonzalez and Woods, 2002b; Tziritas and Labit, 1994) represent an alternative for multiresolution decomposition. The wavelet pyramid contains both the coarse structure and the details in one binary tree. The planes in the tree are obtained by decomposing the image in limited frequency subbands defined by a function basis. The latter is in turn generated from a single function that is scaled and shifted. These functions are well localized both in the space and frequency domain. Wavelets are a very flexible and fast tool (see e.g. (Jain, 1989) for a review of some transforms used in image processing).

Figure 2.2.: Scale-space for an image showing protesting cows (Cows, 2008). The scale-space is obtained by filtering the original image with Gaussian filters of increasing variance.

2.1.3. Images: Observations of Random Variables

This model handles image data as statistical quantities, e.g. (Forsyth and Ponce, 2003). Variations in the image are usually caused by measurement noise in the imaging device, and by fluctuations in the visual appearance and the geometry of the object. Statistical modeling gives the possibility to determine the law of variation from the data for image denoising and segmentation purposes. Pixels are treated as individual measurement points, to which classical statistics concepts can be applied. As an example, we show the (not photoshopped) image in Fig.2.4 [1]. The pixels of the image are lined into a vector of values for which a histogram may be constructed to gain information about the way pixel intensities in the image vary. Two main statistical modeling approaches exist.

Pixels are very often considered to be independent identically distributed random variables, because this hypothesis considerably simplifies calculations for determining the variables' (joint) probability density. This model does not include any information on the spatial relation between pixels. Spatial information is, however, very important in understanding the structure of an im-

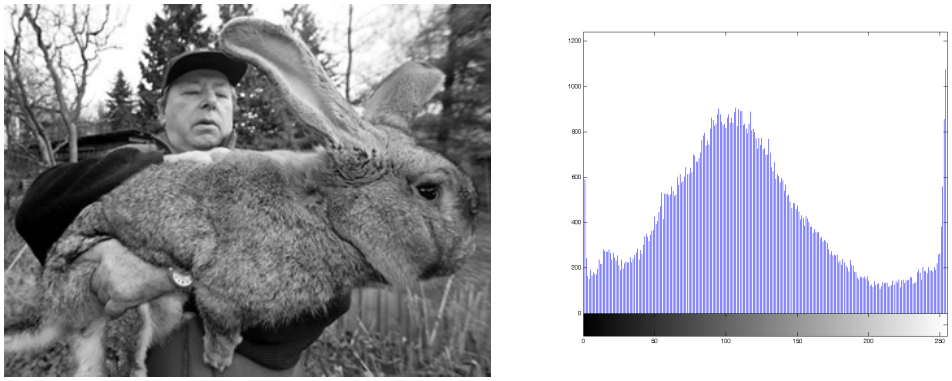
[1] The rabbit is a "Deutschen Riesen Grau" bred in Eberswalde by Karl Szmolinsky, who stopped exporting them to Nord Coreea because they were eaten without being allowed to breed first.

Figure 2.3.: Scale-space for an image showing protesting cows. The scale-space is obtained by filtering the original image with Gaussian filters of increasing variance.

age, since pixels depend on one another over large image areas. Considering all the interactions results in intractable computation problems. To simplify the problems, it is supposed that interactions between pixels are of short range. Pixels depend only on those other pixels found in their immediate neighborhood, creating a Markov Random Field (MRF), e.g. (Geman and Geman, 1984). In a MRF setting, one estimates the probability density of a pixel conditioned on its neighborhood, or the joint probability of pixels in the same neighborhood.

For the segmentation task, it is considered that each image object is generated from a different random process, characterized by the probability density estimated with one of the two modeling approaches. The variation laws for the different objects must be different in order to obtain a successful segmentation. An image of sand dunes and sky is a good example of an image showing objects with different probability densities.

Determining the probability density function (pdf) of each of the random variables involved



(a) (b)

Figure 2.4.: Statistical modeling of images: (a) (not photoshopped) image (Rabbit, 2008), and (b) its histogram showing information about the variation of pixel intensities in the image.

is part of the statistical modeling. Generally, it is accomplished in a maximum likelihood or Bayesian framework, where the probability of pdf parameters given the data is expressed in terms of the probability of the data given the parameters. The modeler must specify the type of density function before any estimation is possible.

The values observed at any pixel may be measured or computed image features. For example, the pixel intensity is measured in gray images, while pixel color is measured in color images. From this raw data, features that enhance the image or that describe properties like texture can be computed. A pixel may be characterized by any number of features. The random variable associated with a pixel's intensity is replaced by a random vector. For instance, the pixel's color is described by a random vector with three components. Although theoretically unlinked, practically the dimension of the random vector restricts the choice on the density function model to be used. It is common to select non-parametric pdf models for more than three feature components (Brox et al., 2003a; Awate et al., 2006; Silverman, 1986), while for three or less features both parametric and non-parametric estimation is employed. For the segmentation of complex natural imaging, statistical modeling is a difficult task, because it is extremely challenging to press natural variations into mathematical functions that have minimally overlapping supports. To justify this statement, we fist review relevant methods of probability theory and statistics.

2.2. Probability, Statistics and Density Estimation

The ideas in this dissertation are constructed on the basis of statistical image modeling. This section reviews relevant concepts of probability theory and density estimation, and establishes the corresponding mathematical notation.

2.2.1. Probability and random variables

Many phenomena in nature, from light to people, are uncertain. This means that one cannot know exactly the outcome of a specific event, but can only guess it using ones experience. Looking out the window in the evening, a person will see mainly the sunset, but will also see a faint reflexion of themselves. In physics, this phenomenon is known as the partial reflexion of light off glass, and raises the question: why is it that glass reflects only four percent of the incident light? Why does glass not reflect all of the photons that hit it or none at all? The answer is that the phenomenon is probabilistic, only some of the photons get reflected from the glass, and there is no telling which of all the photons sent toward a glass plate will be the reflected ones. Probability, random processes and random variables can formalize the fact that, when a photon is sent toward the glass, we do not know whether it will be reflected or not, but, we do know that, in general, one of twenty-five photons sent toward the glass will be reflected. Probability and random variables provide the mathematical tools to quantify intuitive concepts like: "about", "almost", "seldom", "often".

The notions introduced shortly in this sections can be found, in detail, in any good book on probability. For example, we refer the reader to the textbook of Papoulis (1990).

For an experiment with uncertain outcome, probability gives a measure of the chances of each possible outcome. The set of all possible outcomes is denoted as the *sample space*, Ω, of that experiment. An *event* A is the subset of samples for which a given something occurs. The probability of an event is the chance that the event will occur. It is a function, denoted $P(A)$, defined on Ω, which assigns a positive value to each event in the sample space, $P(A) \geq 0$, such that the probability of the sample space adds to one, $P(\Omega) = 1$, and the probability that either one of two events occurs satisfies
$P(A \cup B) = P(A) + P(B)$, if and only if $A \cap B = \emptyset$, otherwise, and
$P(A \cup B) = P(A) + P(B) - P(A \cap B)$).

In many situations, the occurrence of one event may lead to the occurrence of another. Light can be reflected from glass only if it is emitted by a source. However, only a part of the photons emitted by the source will reach the glass. The probability that a photon emitted by the source will be reflected from the glass is a *conditional probability*. The probability of an event A conditioned by the occurrence of event B is denoted $P(A|B)$ and it is defined as

$$P(A|B) = \frac{P(A \cap B)}{P(B)}.$$

The conditional probability $P(B|A)$ can be written in a manner similar to the equation above. If $P(A \cap B)$ is expressed from both forms, *Bayes' law*, the law used for Bayesian inference, is obtained

$$P(A|B) = P(B|A)\frac{P(A)}{P(B)}. \tag{2.1}$$

If the occurrence of an event A has no influence over the occurrence of an event B, the events are said to be independent, and the following holds $P(A|B) = P(A)$, and $P(A \cap B) = P(A)P(B)$.

In general, the notation $P(A)$ refers to the probability of the specific event A. To refer to the events of an experiment in general, the concept of random variable is introduced. A random variable is a symbol X to which the sample space Γ and the probability function P are associated.

A *discrete random variable* has a discrete sample space and the probability of a specific event x_i is denoted $p(x_i) = P(X = x_i)$. In our photon example, the sample space of the light reflexion experiment has two events, and $p(photon\_reflected) = 0.04$.

A *continuous random variable* has a continuous sample space. The number of possible outcomes is thus infinite, which means that the probability of almost all possible events is zero. A finite number of possible outcomes is obtained if the continuous sample space is divided into finite intervals, and the uncertainty in the random variable is measured with the *probability density function* (pdf):

$$p(x_i) = \lim_{\delta \to 0} \frac{P(x_i < X < x_i + \delta)}{\delta}. \tag{2.2}$$

A possible event is that the random variable X takes a value in a given interval, and the event's probability is given by

$$P(x_i < X < x_{i+1}) = \int_{x_i}^{x_{i+1}} p(x)dx. \tag{2.3}$$

Since the probability of the sample space still must be one, the probability density of a continuous random variable integrates to one: $\int_{-\infty}^{\infty} p(x)dx = 1$. A special type of event is that the random variable is smaller than a value x. The probability $P(X < x)$ regarded as a function of x constitutes the *cumulative distribution function* (cdf) of the random variable X. The pdf is the derivative of the cdf, fact that can be deduced from their definitions.

Random variables are characterized by an ample number of statistics. We mention here the two most used ones, *expected value* and *variance*. The expected value is defined as

$$E_X[X] = \sum_{x_i \in \Omega} x_i p(x_i) \tag{2.4}$$

for a discrete random variable, or

$$E_X[X] = \int_{\Omega} x p(x)dx. \tag{2.5}$$

for a continuous random variable. The law of large numbers ensures that the limit of the sample mean $E_x[X] \equiv \frac{1}{n} x_i$ is the expected value.

In addition to the expected value (or mean), it is instructive to know how close observations from X will be to the mean, on average. For this purposes, one computes the expected variation, called

variance

$$Var(x) = E_X[(X - E_X[X])^2] = E_X[X^2] - E_X[X]^2. \quad (2.6)$$

The mean distance to the expectation is the square root of the variance, and it is known under the name *standard deviation*.

2.2.2. Statistical inference

In practice, the probability density of the studied phenomenon is unknown, only measurements being available. The unknown aspects of the phenomenon can be inferred from statistics on the measurement data. The general principle of statistical inference is to select a model for the random variable, and then test how well the model explains the data using various different methods (Papoulis, 1990). We mention here three of the most widely used methods in computer vision. All three models rely on Bayesian inference, which allows to express the desired probability of the model given the data in terms of the known probability of the data given the model.

Maximum likelihood (ML) selects the random variable model that maximizes the likelihood of the observed data $x_1, x_2, ...x_n$, denoted as the data sample $\mathbf{x}$ (Papoulis, 1990). The likelihood of the sample $\mathbf{x}$ is defined as a conditioned probability $L(x) = P(x|X) = P(x_1, x_2, .., x_n|X)$. The model is selected to maximize the likelihood of the sample in an iterative step procedure: 1) guess a random variable model that might explain the phenomenon; 2) evaluate the model's quality by computing likelihood of the data sample, 3) iterate steps 1) and 2), and retain the model that makes the data most probable after having evaluated more models.

Maximum-a-posteriori (MAP) estimation is also based on likelihoods (Duda et al., 2001). It maximizes the likelihood of the model X, defined as $L(X) = P(X|x)$. Since only P(x|X) can be known, Bayes's law is used to express the probability of the most likely model given the sample as $P(x|X) = P(X|x)(P(x)/P(X))$. Note that maximum likelihood is a particular case of maximum-a-posteriori with $P(x)/P(X) = 1$. Maximum-a-posteriori estimation is employed when we have a-priori information about the studied phenomenon that can be quantified as an a-priori probability $P(X)$. The data is the same for all evaluated models, which means that the unconditioned probability of the sample P(x) does not participate in the maximization process, and can be ignored. Priors are not known, they must also be estimated. Priors cannot be estimated without making biased assumptions about the data. This statement is known as the Ugly Duckling theorem, and it is discussed in detail in the textbook by Duda et al. (2001).

Expectation Maximization (EM) is an iterative procedure for obtaining a ML or MAP when part of the data is missing (Duda et al., 2001). Didactical examples involve a data set collected from men and women, e.g. height, weight, or the proverbial time to find the butter in the refrigerator, for which it is not known if an observation stems from a male or female subject. We want to compute the statistics for each population separately. The time-to-find-butter constitutes the observed data $\mathbf{x}$, and the type, male or female, constitutes the hidden data y. Random variables

X and Y are associated to the observed and hidden data.

The idea behind the EM algorithm is to alternate between computing a model X and guessing what the missing data is. For this purpose it iterates two steps: 1) E-step: given parameter estimates and a guess about the unknown variable y, compute its expected value; 2) M-Step: with the computed expectation value, estimate the distribution parameters to maximize the likelihood of the data. For the didactical example, consider that the time-to-find-butter is normally distributed for both men and women, and consider the means and variations of the distributions known. In the E-step compute the expected class, male or female, for each data point. In the M-step, knowing the class for each data point, determine the means and variances that maximize the likelihood of the data.

To arrive at the two steps in the iterative EM-algorithm, we start with the log likelihood of the data given the model, and express the probability of the observed data as the integral of marginal probabilities of the joint observed and missing data $\log(L(x)) = \log P(x|X) = \log \int_y P(x,y|X)$ (Duda et al., 2001). The goal of the E-step is to determine an optimal lower bound on the log likelihood of the data. It can be shown that, for a given model X_i and a guess on Y

$$\log P(x|X_i) = \int_y P(y|x,X_i) \log \frac{P(x,y|X_i)}{P(y|x,X_i)} dy. \tag{2.7}$$

The lower bound is maximized in the M-step, which means we need to find X such that $\log P(x|X)$ is maximum. We write the maximization problem depending on the known X_i as

$$\begin{aligned} \max_X \log P(x|X) &= \max_X \int_y P(y|x,X_i) \log \frac{P(x,y|X)}{P(y|x,X_i)} dy \\ &= \max_X \int_y P(y|x,X_i) \log P(x,y|X) dy \\ &= \max_X E_Y \log P(x,Y|X) \equiv Q(X). \end{aligned} \tag{2.8}$$

The next model X_{i+1} is the one that maximizes the quantity $Q(X)$.

The necessary ingredient for all three methods is the probability density of the data given the model, which is not known. Section 2.2.4 discusses methods for density estimation from observed data.

2.2.3. Markov Random Fields

Spatial continuity is an important clue to observe in natural images. When pixels are modeled as independent, identically distributed random variables the spatial information is lost. The spatial relationships between pixels can be modeled using Markov random fields (MRF), (Geman and Graffigne, 1986; Bouman and Sauer, 1993). For didactical introductions on MRFs see for

example (Gardner, 1990) or (Ngan et al., 1999).

The pioneering works that involved Markov processes referred to time relations between elements of an ordered sequence of random variables. An ordered sequence has the Markov property if each random variable in the sequence depends only on the previous one, or, equivalently, the conditional probability distribution of a random variable given all previous random variables equals the conditional probability of that random variable given its direct predecessor. The Markovian property was first applied to random fields by the physicist Ernst Ising (in the 1920s). He was attempting to devise a mathematical model for the experimentally determined properties of ferromagnetic materials.

A random field X is defined on a finite rectangular lattice $\mathscr{L}$ of pixels by assigning a random variable X_t to each pixel t. A digital image is modeled as a *realization* of the random field obtained by observing all random variables X_t simultaneously. A neighborhood system $\mathscr{N}$ is defined on the lattice as a collection of neighborhoods $\mathscr{N}_t$ such that

$$\begin{aligned} \mathscr{N}_t &\subset \mathscr{L}, \\ t &\notin \mathscr{N}_t, \text{and} \\ (s \in \mathscr{N}_t) &\Leftrightarrow (t \in \mathscr{N}_s). \end{aligned} \tag{2.9}$$

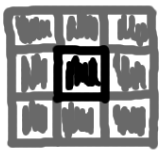


Figure 2.5.: The 8-pixels square neighborhood of the center pixel.

The most popular neighborhood in digital images consists of the 8 pixels connected to the center pixel as shown in Fig. 2.5. Having defined the notion of neighborhood, X is a Markov random field on the sample space Ω if

$$P(X_t|\{x_s\}_{s\in\mathscr{L}-t}) = P(X_t|X_s, s \in \mathscr{N}_t), \text{ and} \tag{2.10}$$

$$P(X = x) > 0 \text{ for all x} \in \Omega. \tag{2.11}$$

The first condition expresses the Markovian property of the random field. The second condition is true of any random field and it is called the *positivity condition*.

The conditional probabilities in a random field should be simplified to a MRF such that the resulting system is consistent. A consistent system is one where the probabilities in Markovian condition can be obtained from the joint pdf $P(X)$ of all the random variables in the system, by employing probabilistic inference rules. In order to have consistency in the MRF, the local

conditional pdfs must have a Gibbs functional form (Besag, 1974)

$$P(X = x) = \frac{1}{Z} e^{-\frac{1}{T}U(x)}. \tag{2.12}$$

The Gibbs distribution, e.g. (Ngan et al., 1999), was introduced in physics, where the constant T is a temperature parameter, and $U(x)$ is an energy function. Z is a normalizing constant that, usually, must not be evaluated. The Gibbs distribution relates the probability of a system state to the state's energy: states with high energy are less probable than those with low energy. The energy function depends on the potential of cliques, $V_{cl}(x)$, associated with the neighborhood system $\mathcal{N}$. Denoting a clique with cl and the set of cliques with Cl, the energy function dependent on the potential function is

$$U(x) = \sum_{cl \in Cl} V_{cl}(x). \tag{2.13}$$

For digital images, a clique contains either a single pixel, or more pixels such that every two pixels are neighbors. The cliques of one, two and three pixels that can be formed in an 8-connected neighborhood system are shown in Fig. 2.6.

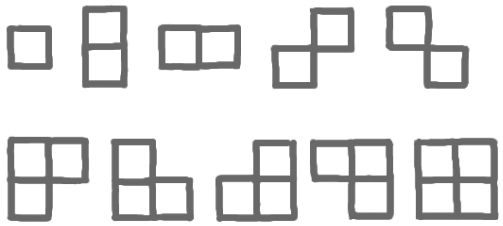


Figure 2.6.: All types of cliques that can be formed in an 8-connected neighborhood system.

To model an image, the parameters of the Gibbs distribution and its potential functions must be estimated from the data. In a maximum-likelihood approach, this means that the probability of the realization x of the random field X must be evaluated for each proposed model. This creates an intractable optimization problem for images of reasonable sizes, unless one makes simplifying assumptions or is content with approximate solutions. Optimization algorithms that find approximate maximum-likelihood estimators are iterative and they are either stochastic or deterministic in nature, see for example (Ngan et al., 1999).

Stochastic relaxation (also known as simulated-annealing) algorithms (Kirkpatrick et al., 1983) start from an arbitrary initial estimate and employ a partially random search on the sample space Ω. For example, the *Metropolis* algorithm (Metropolis et al., 1953) starts with an initial state at very high temperature and evaluates its "goodness". It then randomly generates a new state within some limits of the old state. The new state is accepted only if it is better than the old one. When a local optimum is approached, the temperature decreases. The algorithm stops at a minimum temperature. Building on the Metropolis algorithm, Geman and Geman (1984) have proposed the *Gibbs sampler* for image restoration problems. This algorithm entails that the local

conditional pdfs in the MRF are known and that the value of a pixel is estimated with participation of neighborhood pixel values. In practice, both algorithms are computation-intensive.

Deterministic algorithms accomplish faster convergence to a local optimum by starting with "smart" initial estimates. The *iterated conditional modes* (ICM) algorithm, e.g. (Ngan et al., 1999), follows a philosophy similar to that of the Gibbs sampler, but it only allows downhill moves. The image is viewed as a noisy version of an MRF realization, and the noise is Gaussian, independent, identically distributed. In each iteration, the algorithm cycles through the ordered indices of pixels, setting the value of the pixel according to the Markovian pdf, given the values of pixels in the neighborhood. The results of this algorithm are dependent on the pixel-indexing order. The *highest confidence first* algorithm (HCF), e.g. (Ngan et al., 1999), was proposed as a modification of the ICM algorithm. It keeps pixels in two classes: committed and uncommitted. Initially all pixels are uncommitted, and once a pixel has been committed to a value or image region it cannot go back to the uncommitted state. HCF solves two problems of ICM: an initial estimate is not required, and the order in which pixels are updated is not influenced by some arbitrary indexing.

One last issue concerning MRFs employed in image processing needs mentioning. In general, they are considered to be *stationary* and *ergodic*, e.g. (Gardner, 1990). A strictly stationary MRF is one where the Markov statistics are shift invariant, $P(X_t) = P(X), \forall t \in \mathscr{L}$. The ergodicity property justifies inferring the pdf of a MRF from a single instance, which is the digital image.

2.2.4. Probability density estimation

Probability density estimation refers to the process of constructing a probability density function from the observed data. Two main approaches to density estimation may be identified: parametric and non-parametric. The researcher chooses the type of density estimator taking into account his knowledge or assumptions about the studied phenomenon.

We may understand parametric density estimation by thinking of the photon reflexion phenomenon. If we bombard the glass with photons, in general, four out of a hundred will be reflected. Because the phenomenon is probabilistic and we can always count on measurement errors, we may count two reflected photons if we conduct the experiment once, and we may count six reflected photons if we conduct the experiment again. Intuitively, we assume that the more experiments we make, the average of reflected photons will be four. We also assume chances that we count four photons are the same with the chances that we count six photons, and that they are both larger than the chance that we count seven or three. We have the same intuition about many variables in physics and biology, like men's hight. If we plot the data we obtain curves of similar shapes, known shapes, which can be described by parametric functions.

Parametric density estimation is chosen by researchers when they assume that the data has been drawn from a known parametric family of distributions. The best known and most often used

parametric distribution is the Gaussian (normal) distribution, e.g. (Papoulis, 1990) with mean m and variance σ^2

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-m)^2}{2\sigma^2}}. \tag{2.14}$$

Examples of the bell-shape of the normal distribution is shown in Fig. 2.7(a), along with some other distributions that are sometimes used in computer vision. The Laplace distribution, e.g. (Gardner, 1990), in Fig. 2.7(b) is reminiscent of the normal distribution. Its pdf depends on the absolute distance to the mean, instead of the squared distance to the mean. Fig. 2.7(c) shows examples of the Rayleigh distribution. If the components of a random vector (X,Y) are random variables that follow a normal distributions, their Euclidean norm $\sqrt{X^2+Y^2}$ follows a Rayleigh distribution, e.g. (Gardner, 1990).

A discrete random variable with a distribution similar to the normal one, has a binomial distribution. The resemblance can be observed looking at Fig. 2.7(a) and 2.7(d). As the number of trials in the binomial distribution goes to infinity, the binomial distribution tends toward the Poisson distribution, e.g. (Gardner, 1990), which indicates the probability that a number k of events will occur in a given time-interval, knowing the average rate λ at which events occur in that time-interval

$$f(k;\lambda) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{2.15}$$

The length of the time-interval between two occurrences of a random variable with a Poisson distribution follows an exponential distribution. The shapes of the Poisson and exponential distributions are shown in Fig. 2.7(e) and Fig. 2.7(f), respectively.

The Poisson distribution is used to model probabilities of the number of calls per minute to a call center, or the number of new arrivals into a queue of people waiting to become the happy possessors of a signed copy of the book "365 Ways to Become a Millionaire" [2]. The Poisson distribution can also be used to approximate the cumbersome binomial distribution.

We conclude this section with two examples or random variables with parametric distributions. At high temporal resolution, the photons emitted by a light source follow a Poisson distribution. Integrated over comparatively large periods, the light intensity fluctuates according to a normal distribution. An example from biology involves the sizes of grown biological creatures, which follow a lognormal distribution.

[2] The book's complete title is "365 Ways to Become a Millionaire (Without Being Born One)", is written by Brian Koslow and can be bought for $0.89. The shape of the Poisson distribution intuited by the reader probably reflects the degree to which his view of his fellow men is optimistic.

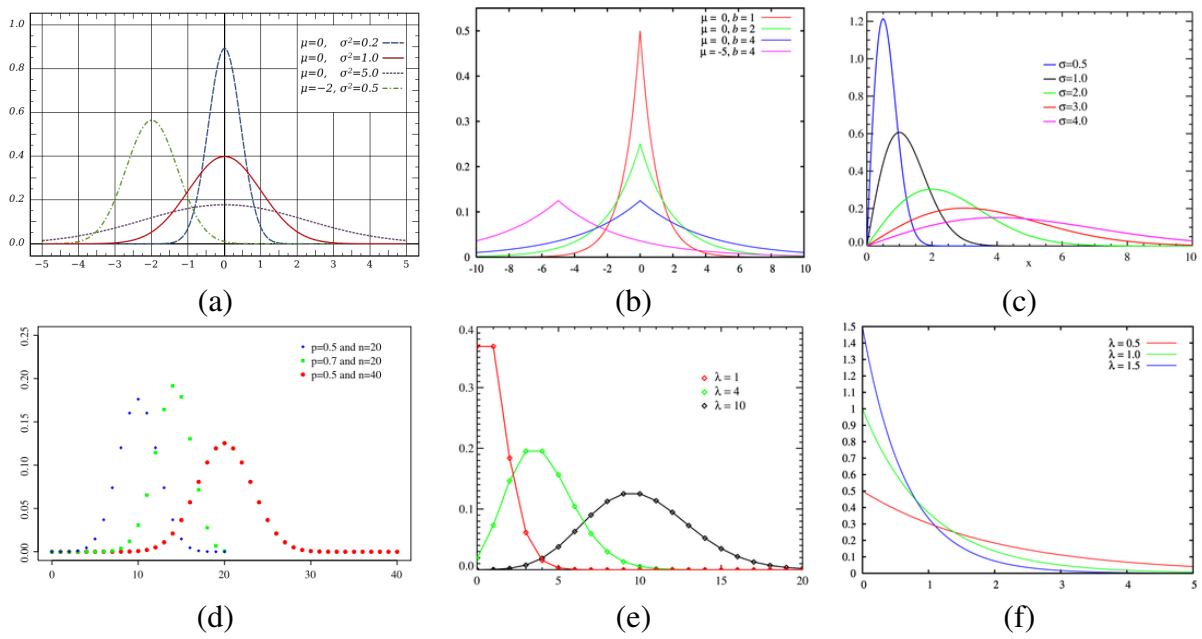


Figure 2.7.: Examples of parametric distributions shapes (**?**). (a) Normal distributions. (b) Laplace distributions. (c) Rayleigh distributions. (d) Binomial distributions. (e) Poisson distributions. (f) Exponential distributions. (Image sources:wikipedia.org)

2.3. Nonparametric Density Estimation

Virtually all parametric pdfs are unimodal. In many practical situations, it happens that the data exhibits multiple modes. Take, for example, the height in a group of basketball-players celebrating the team's victory with a large group of friends. The pdf that models this random variable will have at least three modes, one at a smaller height for females and one at a larger height for males and the largest one for the basketball-players.

A good probability model must both summarize and generalize the available dataset, i.e. it must abstract all information on the variation in the dataset, while being able to recognize which values not present in the data set fit the variation law. In these respects, multimodal distributions are badly modeled by unimodal parametric density functions. An alternative is to use a *mixture* of parametric pdfs, such that each mode is represented by one mixture component. However, an issue exists with this approach. Density estimation is often used to explore datasets when we do not know anything about them. Not knowing the number of modes may turn approximating the mixture of pdfs into an intractable problem. Nonparametric estimation methods have been designed for such situations by using the sample to directly define the pdf, while making the mild assumption that the pdf is smooth. Nonparametric distributions allow, so to say, the data to speak for itself. The following sections presents a short overview of some of the existing methods.

2.3.1. Histograms

The histogram is the oldest and most intuitive density estimator. It is the first tool that comes to mind when exploring a dataset. It helps to quickly gather an impression of both small and very large datasets.

Given a dataset $x_1 ... x_n$, a histogram is constructed by defining for the dataset an origin x_0 and a bin width h. These parameters determine the number of bins in the histogram. The bins of the histogram are defined as intervals of the form $[x_0 + mh, x_0 + (m+1)h)$ The histogram is a function that assigns to all observations in a bin the same value:

$$\hat{f}(x) = \frac{1}{nh}(\text{no. of observations in the same bin as x}), \tag{2.16}$$

where $\hat{f}$ denotes the estimate of the density f. One version of the histogram allows the bins to have different widths, to accommodate the fact that some regions of the sample space contain more data than others. Due to its simplicity, the histogram is computationally inexpensive compared to other methods. The two qualities confers the histogram's great popularity in computer vision applications. Nonetheless, histograms have some notorious drawbacks.

Some issues exist in summarizing the data. The choice of origin and bin width may have quite an effect on the shape of the estimate. This is especially true in higher dimensions, but can happen even in one dimension. On the other hand, the histogram may summarize too much detail from the data, which negatively affects its power to generalize. This problem can be alleviated somewhat by smoothing the histogram. Mathematically, histograms cannot be used when the derivative of the pdf is required, since histograms are discontinuous. In higher dimensions, the problems occurring in one dimension are exacerbated, and the computational advantage may be lost. For example, it is easy do decide that the histogram of a gray image should have 256 bins. For a color image, the number of bins of equivalent size is 256^3, and many of them are empty or contain very few elements. If the size of the bins is increased, the data may be irremediably distorted, and important details may be lost.

The alternative estimators presented here have been designed to solve one or the other problem inherent to histograms.

2.3.2. The kernel estimator (Parzen windows)

The kernel density estimator is a continuous function based on the definition of the pdf for a continuous random variable, which can be written as the limit of $f(x) = 1/2h \cdot P(x-h < X < x+h)$ when $h \to 0$. Every sample space point x becomes the center of a sampling interval of size $2h$. Observations within the interval contribute to the estimate $\hat{f}$ via a *kernel function* K which

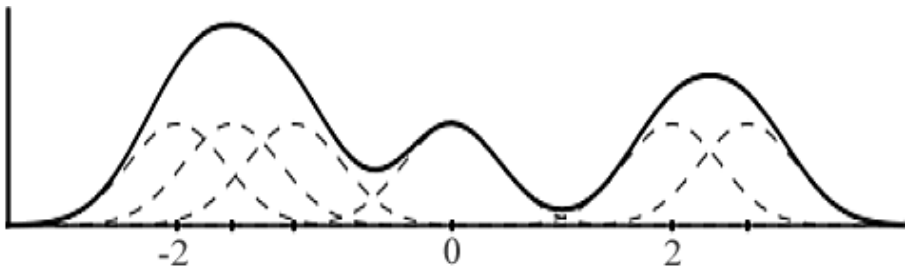


Figure 2.8.: Kernel density estimation: six observations used to construct the estimate (continuous line) from individual Gaussians centered on the observations (dotted line).

satisfies the condition $\int_{-\infty}^{+\infty} K(x)dx = 1$, and is in general symmetric

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x-x_i}{h}). \tag{2.17}$$

Parameter h is the *window width*, also named *bandwidth* (with no relation to the bandwidth in spectral analysis) or *smoothing parameter*. The kernel estimator can be viewed as a sum of n parametric functions of given shape, with width determined by the window width, and centered on the n data points. To illustrate the principle as in Fig. 2.8, the kernel function has been chosen to be a Gaussian (Silverman, 1986). The figure shows the individual functions, along with the estimate obtained by adding them up. Nonparametric kernel estimators based on Gaussian kernels are most often encountered under the name *Parzen windowing*. Another popular kernel function is the *Epanechnikov* kernel (Silverman, 1986)

$$K(x) = \begin{cases} \frac{3}{4}(1-x^2), & \text{for } |x| \leq 1, \\ 0, & \text{otherwise.} \end{cases} \tag{2.18}$$

By centering an interval on each sample point, the kernel estimator avoids problems introduced by rigidly choosing bins for the histogram. Since the kernel is everywhere positive, everywhere derivable, and a density function, the estimate itself will be a continuous, everywhere derivable density function. The kernel density estimate converges to the true density in the case of an infinite number of observations (e.g. see (Duda et al., 2001)).

The density estimate is dependent on the window width, the dataset being given. As the window width tends to zero, every data point introduces a spike in the density's shape. The shape becomes smoother as the band width increases, while details contained in small variations in the data are lost. The dependence is illustrated in Fig. 2.9, reprinted from (Viola, 1995). The plots have been generated from a hundred point sample of a zero-mean Gaussian with variance one. The kernel function employed to compute the estimate was also a Gaussian, the most used function in kernel density estimation. For this function, the window width is equivalent to the variance of

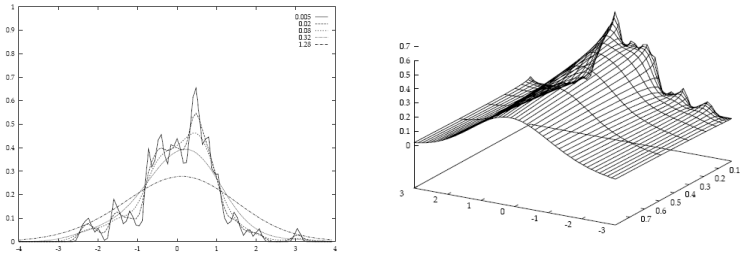


Figure 2.9.: Influence of the window width on the kernel density estimate. Five plots of the density estimate from the same 100 observations but with five different window widths (left), extracted from the surface plot of the Parzen density estimation versus variance, for the same sample (right). (Source: (Viola, 1995), pp. 46,47.)

the Gaussian, which, in the presented example, ranges from 0.005 to 1.28.

The kernel function and the window width should reflect any information available a-priori, given its large influence. However, in many situations, there is no such information available. Due to its importance, the problem of choosing the optimal kernel and window width has been intensively studied, and a number of methods for choosing the window width have been proposed; for a review see, for example, (Silverman, 1986). Most of them lead to complex solutions that either involve some new parameter, or impose additional constraints on the dataset. Here, we mention a rather simple method for choosing the window width that aims to maximize the log likelihood of the data, given the density estimate. Maximizing log likelihood can be shown to be well approximated by minimizing entropy (defined in Section 2.4.1) in this situation (Viola, 1995)

$$\log L(x) = \log P(x_1 ... x_n | \hat{f}) \approx \log \prod_{i=1}^{n} \hat{f}(x_i) = \sum_{i=1}^{n} \log \hat{f}(x_i), \tag{2.19}$$

and the last term dived by n is the expectation of $\log \hat{f}(x;h)$, also known as *empirical entropy* approximation of the entropy of the random variable X. If empirical entropy and density estimate both use the same observations, the minimization of empirical entropy with respect to window width has the trivial solution $h = 0$. To avoid this solution, when computing the estimate at observation x_t, empirical entropy is computed from a set of observation that does not contain x_t. Fig.2.10 shows plots of empirical entropy for a random vector with 9 components obtained from 3×3 image patches. The empirical entropy is estimated with a Gaussian kernel function, versus kernel variance, for two samples selected from the same image. The plot in the middle shows a broad minimum, which suggests that in this situation the density estimate is not extremely sensitive to the value of the variance. However, we have previously discussed that this need not always be so. For example, the plot on the right shows a decreasing entropy function with no minimum.

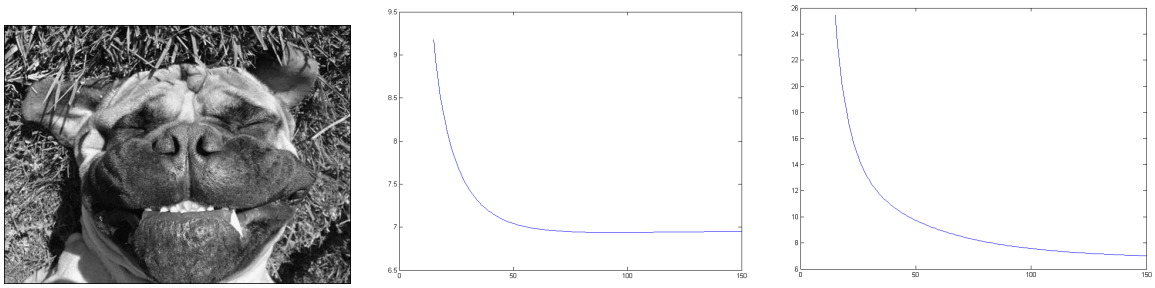


Figure 2.10.: Empirical entropy evaluated from two different samples for the image on the left. Observe that the one in the middle has a broad minimum, while the one on the right does not have a minimum.

2.3.3. The Nearest-Neighbor Method

The optimal window width for describing an entire data set may be inexistent. This is because observations are often disproportionately scattered over the sample space. Some regions in the sample space may be very densely populated, while others are much less so. The information content is higher in densely populated regions, and variation in the observations carry more importance than in sparsely populated regions. The k-nearest-neighbors method represents a potential remedy to the problem of finding the best window width (Duda et al., 2001).

The method estimates $\hat{p}(x)$ from an interval centered at x large enough to contain exactly k observations, known as the k nearest-neighbors of x. The parameter k is typically $\sqrt{n}$, where n is the total number of observation (Silverman, 1986). If the distance between x and the observation x_i is denoted with $d_i(x)$, the size of the interval will be $d_k(x)$. In regions of high density, d_k will be relatively small, preserving important detail information. In regions of low density it will be relatively large. The k nearest-neighbor density estimate is defined as

$$\hat{p}(x) = \frac{k}{2nd_k(x)}. \tag{2.20}$$

To understand what this definition does, we can compare it to the bin of a histogram. The bin of the histogram counts how many samples fall withing a given interval, and the density estimate is directly proportional to this number. The nearest-neighbor estimate determines the size of the cell that contains a given number of observations, and it is inversely proportional to it.

The nearest-neighbor estimate tends to yield functions with very long tails, since tail values are usually situated in sparsely populated regions of the sample space. Additionally, the obtained function is not a smooth curve, it is not a probability density because it does not integrate to one, and it is only piecewise derivable. These last issues are remedied by the variable kernel estimator described next.

2.3.4. The variable kernel estimator

The kernel density estimator has many desirable properties, but requires the choice of a fixed window width. However, there is no reason why we should not estimate a window width $h(x)$ at each x, like we did in nearest-neighbor estimation, and replace h with $h(x)$ in the kernel density estimator formula. We then obtain the *balloon density estimator*, sometimes denoted *generalized k nearest-neighbor estimate* (Silverman, 1986)

$$\hat{p}(x) = \frac{1}{nd_k(x)} \sum_{i=1}^{n} K(\frac{x - x_i}{d_k(x)}). \tag{2.21}$$

The balloon estimate still suffers from some of the issues in the nearest neighbor method, since it depends on $d_k(x)$. The balloon estimate is negatively affected by local noise, it has long tails and an infinite integral.

An estimator that is a probability density function emerges if the $d_k(x)$ is replaced with the distance $d_k(x_i)$, i.e. the distance between the observation x_i and its kth nearest neighbor. The new estimator is the *variable kernel estimator* or *sample point estimator* with smoothing parameter h (Silverman, 1986)

$$\hat{p}(x) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{hd_k(x_i)} K(\frac{x - x_i}{d_k(x)}). \tag{2.22}$$

All following facts referring to the variable kernel estimator can be found, for example, in (Silverman, 1986). The smoothing parameter h is often omitted, since it is constant and can be incorporated in $hd_k(x_i)$. Instead of selecting a window width for each *estimation point* x, the variable kernel estimator selects a window width for each *observation* (or sample point) x_i. The window width is no longer dependent on x, which substantially reduces problems in estimating the densities tail. Also, the density is not affected anymore by the position of the estimation point, being fully determined by the dataset.

There exists an optimal choice for the window width at an observation point x_i, which we may denote with $h(x_i)$

$$h(x_i) = h_0 \left(\frac{\lambda}{\tilde{p}(x_i)} \right)^{\frac{1}{2}} \tag{2.23}$$

where h_0 represents a fixed window width and λ is a proportionality constant. The estimate $\tilde{p}$ is called a *pilot density*. It is sufficient if the pilot density is only a rough approximation of the underlying density. Its purpose is to allow to compute the window widths $h(x_i)$ in order to obtain a more accurate estimate.

Parameters h_0 and λ allow a higher level of abstraction, but must still be chosen by the researcher. The estimate in equation 2.22 depends on the value of λ. This parameter represents the threshold between low and high densities. When the local density is low, the window width $h(xi)$ is larger

than h_0 leading to increased smoothing around the point x_i. Reciprocally, for regions of high density, the window width becomes narrower.

Automatic selection of window-width size is an intensively studied problem due to countless applications that profit from data analysis. A very interesting and recent theoretical development is (Katkovnik and Spokoiny, 2008), and it has been applied to image segmentation and denoising problems in (Polzehl and Spokoiny, 2003; Tabelow et al., 2006). The authors propose that the kernel function be viewed as model of the local data within each window centered on each observation. The best window-width is found iterating through a list of possible sizes. In each iteration, given the window size, the maximum-likelihood fit to the data in the window is computed. One then statistically tests the hypothesis that residuals between the model and the data can be treated as pure noise. The best window width is the largest window that passes the test. In their paper, the authors provide the theoretical bounds of such a statistical test.

The nonparametric methods in this chapter have been introduced while discussing random variables for the simplicity of illustrations (all illustrations are 1D, allowing the reader to concentrate on the introduced method). Their definitions for random vectors, i.e. in higher-dimensions, are the same. The formulas are only changed to replace the volume of the one dimensional interval $2h$ with the volume of a d-dimensional sphere and correcting factors that ensure pdfs will integrate to one. These formulas will be reminded in the latter chapters, as we will use them to develop our methods.

2.4. Information Theoretic Measures

Image processing tasks typically involve two or more random variables or vectors. For example, different image objects are associated with different random variables in order to accomplish segmentation. Objects may be assigned labels, which may be also viewed as random variables that transform the image segmentation problem into a data classification problem. Researchers are often interested to view the pdfs of the different random variables in relation to one another. For instance, how much does the pdf of a pixel feature tell us about the pixel's label.

Knowledge on the functional dependence between random variables and the amount of uncertainty associated with them is acquired from information theoretic concepts like entropy and pdf divergence measures. Some of the concepts mentioned in this dissertation will shortly be described in order to help the reader follow some of the discussion in following chapters.

2.4.1. Entropy, Conditional Entropy and Mutual Information

The entropy of a random variable is a measure of how random the variable is. Are the birthdays of babies more or less random than lottery numbers? Entropy, typically denoted $H(X)$ quantifies the answer to this questions

$$H(X) = \sum_{x_i \in \Omega} P(X = x_i) \log \left(\frac{1}{P(X = x_i)} \right) = - \sum_{x_i \in \Omega} p(x_i) \log p(x_i). \tag{2.24}$$

This entropy measure is known as *Shannon entropy*. Lower entropy values are associated with more certainty concerning the random variable.

From its formula, one observes that Shannon entropy is the expectation of the random variable $(-\log P(X))$. This expectation can be approximated by $1/n \sum_{i=1}^{n} -\log p(x_i)$, and it is also known under the name *empirical entropy*. Writing the sum of logarithms as logarithm of products, Shannon entropy is proportional to the negative of the log likelihood by a constant factor. A maximum likelihood optimization may thus be replaced with a minimum entropy optimization.

The concept of entropy also helps express the degree of certitude that two random variables are functionally dependent. For this purpose, we first define the conditional entropy of a random variable Y given random variable X

$$H(Y|X) = E_X[E_Y[\log(P(Y|X))]]. \tag{2.25}$$

Larger functional dependencies result in lower values for conditional entropy. However, a small value of $H(Y|X)$ can also result from the fact that the entropy of Y is small itself. To circumvent this problem, *mutual information* is introduced

$$I(X,Y) = H(Y) - H(Y|X). \tag{2.26}$$

to measure the amount by which the entropy of Y decreases when the entropy of X is known. It can be demonstrated that $I(X,Y) = I(Y,X)$. Methods based on maximizing mutual information have gained much popularity in computer vision, especially due to the work of Viola and Wells (1997). The work discusses in detail desirable propertied of mutual information and shows its efficiency in ill-posed alignment problems.

2.4.2. Divergence Measures

In most applications we are not interested in functional dependencies. Instead, we want to know if two pdfs are similar or not. For example, two pdfs are estimated and we need to know if they represent estimates of the same random variable. A well know similarity measure between pdfs

p and q is the *Kullback-Leibler Divergence* (Duda et al., 2001)

$$KL(p||q) = \sum_{x_i \in \Omega} p(x_i) \log \frac{p(x_i)}{q(x_i)}. \tag{2.27}$$

This quantity is not symmetric, hence it is not a true distance measure. A true distance that measures the mis-match between pdfs is the *Bhattacharyya distance* (Duda et al., 2001)

$$B(X,Y) = \sum_{x_i \in \Omega} \sqrt{p(x_i)q(x_i)}. \tag{2.28}$$

Another true distance is the Wasserstein distance. Unlike the Bhattacharyya distance, it is defined via cumulative distributions. If we denote the cdf of two random variables F_p and F_q, the definition for the continous case reads

$$W(p,q) = \int_0^1 |F_p(x) - F_q(x)| dx. \tag{2.29}$$

Many other divergence measures exist, and choosing the appropriate one for a given problem is a topic of intensive research in statistics. We mention that a relatively recent example is (Barth et al., 1999), and we do not further immerse in this topic.

2.4.3. Mann-Whitney-Wilcoxon test

Some of the algorithms mentioned in this dissertation incorporate divergence measures into an optimality criteria, typically minimization. Much fewer ones adopt a variant where the divergence measure is thresholded in order to obtain some result. One such example is the Mann-Whitney-Wilcoxon test. This is a nonparametric statistics test for assessing whether two samples of observations have been drawn from the same distribution. For instance, we would like to know if two image patches belong to the same object.

The null hypothesis is that the two independent samples (e.g. image patches) are drawn from the same variable (e.g. region). The test calculates a statistic, usually called U. For two small samples of possible different sizes, the statistic U is obtained in a few trivial steps: 1) sort both sample into a single ranked series, 2) decide which is the sample with smaller ranks, and let this be sample1, and the other sample2, 3) for each observation in sample2, count the numbers of observations smaller than it in sample1, 4) set the value of U to the sum of these counts. The distribution of U is known when the null hypothesis is true. The null hypothesis is accepted or rejected by checking the tabulated values of U.

3. Active Contours: Overview and State-of-the-art

> This fellow takes things seriously, it isn't safe to ask him a simple question.
>
> *Groucho Marx*

In this chapter we will answer the question: "What are active contours?". Active contours are methods based on models of physical phenomena. In the introduction, we have resorted to the rubber-band model for simplicity. However, a more appropriate model is that of moving (or evolving) interfaces (or curves). Both terminologies are often used. Intuitively, the evolving curves exist in a plane on top of the image. In order to find object boundaries, the curves move under the action of forces generated from image properties while staying smooth.

The forces are oriented in the direction of the contour's normal vectors. At each moment in time, the curve has a position and an energy. The curve moves in each time iteration at a speed dictated by image properties until it reaches a state of minimal energy. Physically speaking, the contour's energy and the motion forces directly determine one another. Mathematically, the energy is formulated first, and motion forces are the result of energy derivation for the purpose of gradient descent. For this reason, the active contours framework is a framework of *variational models*. Four issues must be considered to achieve segmentation with this physical model.

First, the visual properties of image regions must be described. This means selecting features to characterize image properties and generating energy terms from them. Visual properties may refer to the strengths of edges, yielding edge-based active contours. The contour is constructed to move toward and stop at image pixels where the measured edge strength is largest. The edge strength is often measured by computing the gradient of the image ∇I. The magnitude of the gradient is plugged into a strictly decreasing function g that contributes to the contour's energy. The energy $E(C, \nabla I)$ is the boundary integral over the contour C. Intuitively, to obtain the contour's energy, one adds the energy bits that exist at each point on the contour and are inversely proportional to the strength of the edge. A force proportional to the energy bit acts on each contour point, a fact which can be mathematically expressed in the form of a partial differential equation.

Alternatively, visual properties may refer to the characteristics of the region occupied by the imaged object, yielding region-based active contours. The contour is constructed to move such as to

encompass only image pixels with similar characteristics. The characteristics or features $f_r \in R^n$ may refer to intensity, color, texture or other measurable quantities. To decide which pixels are similar, the values of pixels in each region encompassed by each contour are analyzed statistically. Probability densities p_i are often estimated in the process (using, for example, histograms or Gaussian distributions).

The contour's energy $E(R_i, C_i)$ is a region integral over the region R_i encompassed by C_i. Intuitively, each pixel inside the region contributes bits of energy to yield the total energy. An energy bit is proportional to the probability that the pixel belongs to the region. This part of the energy is known as the image data term. To this term, one needs to add the energy that constrains the contour to having smooth shapes. This part of the energy is known as the smoothness term. Similar to edge-based contours, forces proportional to the energy bits act on contour points; they are also expressed in the form of partial differential equations. The image in Fig. 3.1 depicts both edge and region-based construction of active contours.

Second, the derivative of the energy is computed to obtain motion forces. The two main approaches for derivation, expectation maximization and shape gradients, will be discussed in Section 3.2. Third, the evolution process starts with initial contours. The positions of initial contours significantly influence the results for the vast majority of active-contour methods. When the user places the initial contours manually, he is able to provide the segmentation method with useful information. However, initialization should be automatic for autonomous computer vision. The best of both approaches is often combined in so-called semi-automatic initializations, as will be shown in Section 3.4.

Fourth, the contour motion caused by the computed forces must be implemented. The implementation algorithm is determined by the curve's representation, which can be parametric or implicit. *Parametric (explicit) models* represent the contour as a chain of particles (points). The contour is obtained by interpolating points between particles. In *implicit models* (or *level-set methods*), the contour is the intersection of a 3D surface with a 2D horizontal plane. The implementation itself may rely on numeric techniques or may be entirely algorithmic. Implementation algorithms are discussed at larger extent in Section 3.5.

The design of an active-contours method follows these four steps, which are illustrated in Fig. 3.2. This figure plays the role of a road-map for the seminal and state-of-the-art material presented in this chapter. The large amount of research reviewed here has been directed at tackling problems in the framework of active contours. Well-know problems include the fact that assumptions about the image do not always hold. For instance, the color gradient magnitudes for some of the edges in the grass and in the dog are stronger than those of edges between grass and dog in the image at the top of Fig.3.1. Also, the color of the fur largely varies, and pixels in the dog region are not similar to each other.

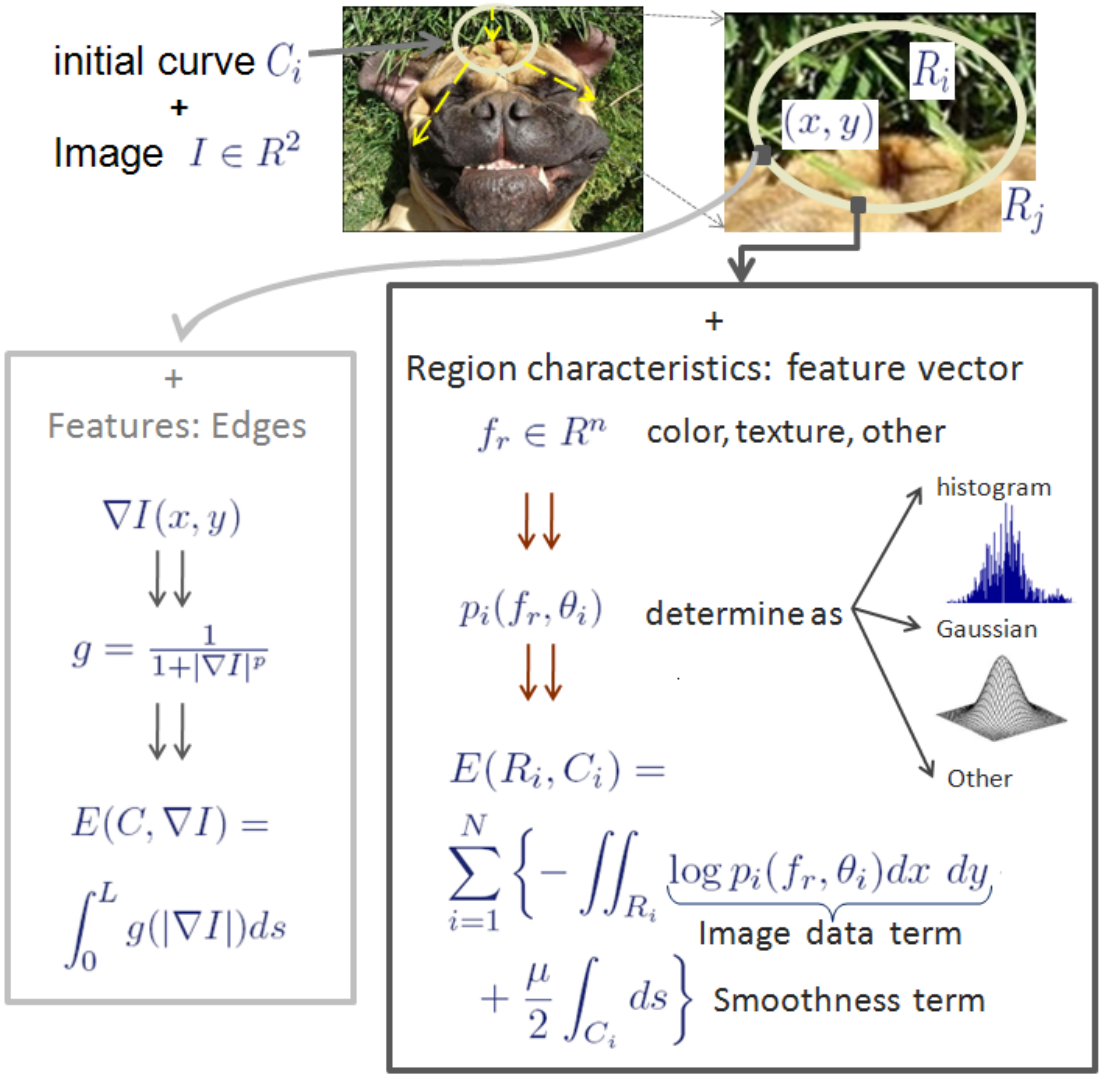


Figure 3.1.: Generation of energy terms from visual properties of the image. Initial image and illustration of features (top).The initial image (top) also shows the initial contour (solid circle) and some of the forces acting upon it (dotted lines). Concise flow for edge-based ACs (bellow left) and concise flow for region-based ACs (below right). The contours stay smooth during their evolution.

3.1. Active Contours Formulations

This section introduces the mathematical formulation and notation for edge-based, region-based and hybrid active contours. The generation of energy terms from image properties, succinctly illustrated in Fig.3.1, are described here in more detail. The presentation is slightly historical, and it starts with the first proposed active contours. This manner of presentation requires a few

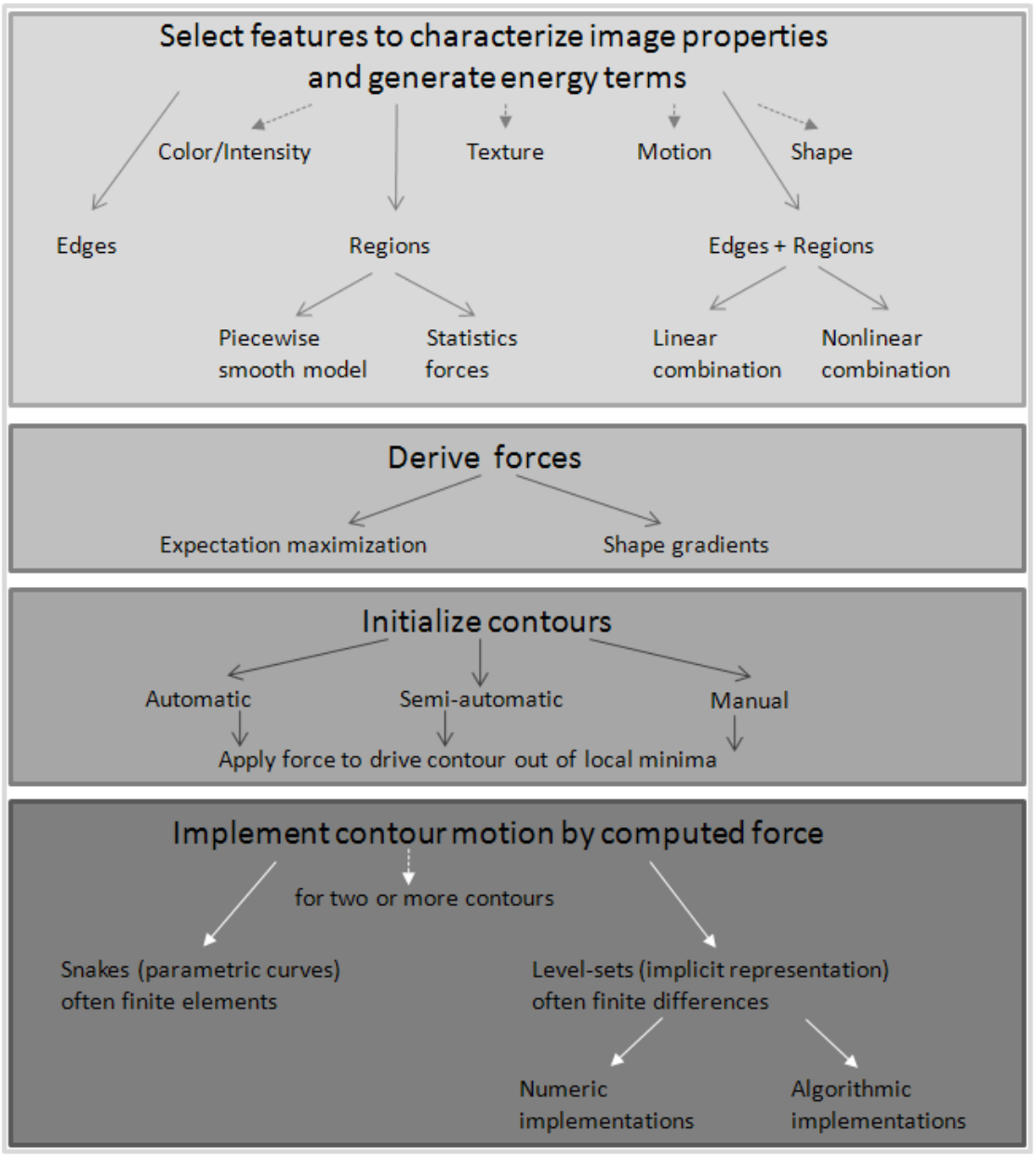


Figure 3.2.: Overview of the active contours framework. Active contours have an energy that depends on image properties of mostly two types: edges and region features. They move toward object boundary while staying smooth.

forward references.

3.1.1. Edge-Based Active Contours

Active-contour methods have been studied intensely since they have been first proposed. Edge-based active contours were first proposed by Kass et al. (1988), and they were called snakes. Their energy depends on edge strength and includes a term to reinforce the smoothness constraint. This type of constraint reinforcement is also known as regularization, and the term added to the energy for this purpose is known as regularization term. Let the contour be a curve C parametrized by p. The energy of a snake is written as the sum of internal and external energies

$$E(C(p)) = \underbrace{\alpha \int_0^1 \left|C'(p)\right|^2 dp + \beta \int_0^1 |C''(p)|^2 dp}_{Internal\ energy} - \underbrace{\lambda \int_0^1 |\nabla I(C(p))| dp}_{External\ energy} \tag{3.1}$$

I denotes the image to be segmented, and the gradient of the image ∇I measures edge strength. C' and C'' are the first and second derivatives of C and they are involved in the definition of the *internal energy*, which is the snake regularization term.

The nomenclature for the first and second internal energy terms are *smoothness term* and *stiffness term*, respectively. The external energy is often called *image data term*, or simply data term. The parameters α, β and λ control the contribution of each energy term to the total energy, and thus they influence on the behavior of the active contour. Edge strength may be measured with any edge filter, not just by computing image gradients.

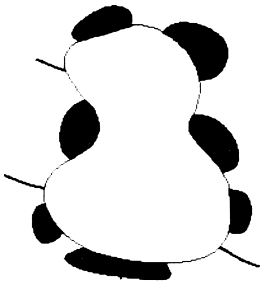


Figure 3.3.: A snake (Kass et al., 1988).

Snakes are very intuitive, easy to implement, and they are an excellent tool for interactive segmentation. They may also close gaps, as illustrated in Fig.3.3. However, the initial snake must be placed in the vicinity of the object to be segmented because its range of capture is small, which means that they converge to the closest edges, and these are not necessarily object boundaries. Solutions proposed to alleviate this problem, e.g. (Cohen, 1991; Xu and Prince, 1997, 1998) will be discussed further in section 3.4.

A snake is generally represented as a chain of particles. To implement snake motion, particles are moved under the action of forces computed from the energy; more details are given in Section 3.5. This makes it a daunting task to prevent a contour from self-intersecting or to deal with situations when contours merge or split. The level-set method (Osher and Sethian, 1988), described later in this chapter, represents curves implicitly and, as a result, does not suffer from any of these problems. It thus incited the idea of snake-like curves represented implicitly (Caselles et al., 1993; Kichenassamy et al., 1995; Malladi et al., 1995). These curves are known as *geodesic active contours* (Caselles et al., 1997), and they differ from snakes in two points: 1) the stiffness term is omitted, and 2) the data term is constructed via a non-linear function $g : [0, +\inf] \rightarrow R^+$ which has the property that $g(x) \rightarrow 0$ as $x \rightarrow \inf$, giving the energy more flexibility. The geodesic

active contour is formulated as

$$E(C(p)) = \alpha \int_0^1 \left|C'(p)\right|^2 dp + \lambda \int_0^1 g(|\nabla I(C(p))|)^2 dp \tag{3.2}$$

$$= \int_0^1 g(|\nabla I(C(p))|)C'(p)dp = \int_0^L g(|\nabla I(C(s))|)ds. \tag{3.3}$$

L denotes the length of the contour, and $C(s)$ is the arc-length parametrization of $C(p)$.[1] Let us denote the curvature of C with k, and its unit inward normal with $\vec{n}$. The curve flow is derived from the energy (3.3) by gradient descent and yields the following partial differential equation (pde) (Caselles et al., 1997)

$$\frac{\partial C}{\partial t} = g(I)k\vec{n} - (\nabla g\vec{n})\vec{n}. \tag{3.4}$$

Edge-based active contours are affected by problems of edge detectors. For example, the edge-filter response is weak where the human eye perceives a strong edge, and conversely, weak responses may occur where the human eye perceives a strong edge. Remember for example Fig.1.4 in the introduction. Consequently, classical edge-based contours leak in the first situation and stop before reaching the correct boundaries in the second.

Currently, two classes of solutions exist to make active contours more robust and overcome problems caused by edge detectors. One class consists of region-based active contours, the other consists of contours that combine region and edge information.

3.1.2. Region-Based Active Contours

The appearance of many objects in the world can be described by their color or texture, while their shapes are mostly composed of smooth surfaces rather than of randomly connected points. An image of an arrangement of such objects is often the union of homogeneous regions delineated by smooth boundaries. Visual characteristics change from object to object, rather then within a single object, while the largest differences are often perceived at boundaries. Mumford and Shah (1985) have formulated these assumptions mathematically, modeling images as functions on a grid domain. Consider an image I approximated by a piecewise smooth function $u(\mathbf{x}) = u(x,y)$ defined on the domain $R \subset R^2$ and discontinuous at the set of boundary points C parametrized by the arc length s. Let us denote the absolute value with $|\cdot|$ and with ∇u the gradient of function u. The best-approximating piecewise smooth function is determined by minimizing the energy functional

$$E_{MF}(u,C) = \int_R |u-I|^2 d\mathbf{x} + \mu \int_{R-C} |\nabla u|^2 d\mathbf{x} + \nu \int_C ds. \tag{3.5}$$

[1] By the chain rule for derivation, we have $C'(p)dp = C(s)$.

The parameters μ and ν weigh the contributions of smoothness of the approximating function and the smoothness of curve, respectively. The smoothness of the approximating function is known as *total variation norm* in a similar and also very well known variational image model, the ROF nonlinear total variation model (Rudin et al., 1992). This model is mainly employed in image denoising, but it also has applications in image segmentation. The form of the energy functional is similar the one in the Mumford Shah model, but does not depend on position of the contour

$$E_{ROF}(u) = \underbrace{\int_R |\nabla u| dx}_{TV(u)} + \lambda \int_R |u-I|^2 dx, \tag{3.6}$$

which explains that its main application is to image denoising. However, we mention it due to its importance and its resemblance to the Mumford Shah model. $TV(u)$ is the total variation norm of the function u. The least squares approximations $|u-I|^2$ is sometimes replaced with the L_1 norm $|u-I|$. A problem in these frameworks is that, although it is possible to model textured regions, this is, in general, a very tedious task (Liu et al., 2002). Also, minimizing the energy functional is not trivial, involving a Poisson-type equation (Tsai et al., 2001).

For textured images, the description is simpler and more compact when the image is modeled statistically. However, statistical modeling is just as suitable for color-based segmentation. In both situations, it is important to capture the information that best distinguishes between different objects. Region information can be gained from the object's color or intensity, or from any number adequate features obtained with any number of arbitrary image filters. In general, the variation of features is formalized and measured using one of the probability density estimators presented in Chapter 2. A recent review of statical methods can be found in (Cremers et al., 2007).

The mathematical framework for active contours segmentation of statistically modeled images was proposed, for example, in the seminal work by Zhu and Yuille (1996), which we shortly describe. Pixel features are assumed to be identically, independently distributed in each image region R_i. The total number of regions in the image is N. The feature vector $\mathbf{f}(\mathbf{x})$ at a pixel $\mathbf{x} = (x,y)$ is the realization of a random vector with pdf $p(\{\mathbf{f}(\mathbf{x}) : \mathbf{x} \in R_i\} \,|\, \theta_i)$. The pdf for each region, shortly denoted p_i, depends on the unknown parameters θ_i. The energy functional for the N region-based contours is

$$E(R_i, \theta_i, N) = \sum_{i=1}^{N} \left\{ \underbrace{\left(\frac{\mu}{2} \int_C ds \right)}_{Smoothness\ term} - \underbrace{\iint_{R_i} \log p_i d\mathbf{x}}_{Data\ term} + \beta \right\}. \tag{3.7}$$

Parameter β penalizes a large number of regions to prevent over-segmentation, while parameter λ controls the influence of the smoothness term. The number of regions N is decreased by merging two adjacent regions, so that the merging causes the largest decrease in energy. The process is stopped when no merge can further decrease the energy. The energy is minimized by

evolving initial contours. The contour that separates regions R_i and R_j is denoted $C_{i,j}$. Its motion is derived using Green's theorem and the Euler-Lagrange equations, as in (Zhu and Yuille, 1996), while considering the parameters θ_i constant. At every contour point, the contour moves in the direction of its outward normal, denoted as $\vec{n}_i$, under the action of a force $F_k = -\mu k_i \vec{n}_i$, where k_i denotes the curvature. We may say that the contour moves at a speed that is directly proportional to the curvature. The speed also depends on the values of the pdfs of regions R_i and R_j at that point

$$\frac{\partial C_{i,j}}{\partial t} = \underbrace{-\mu k_i \vec{n}_i}_{Smoothness\,force} + \underbrace{\left(\log p_i - \log p_j\right)\vec{n}_i}_{Statistics\,forces}, \tag{3.8}$$

The term $F_i = \log p_i \vec{n}_i$ represents the *statistics force* exerted on the contour by region R_i. The force points in the direction of the contour's normal, and its magnitude depends on the confidence that the pixel should be assigned to region R_i. The total of statistics forces thus assigns the pixel to the region where the value of pdf is highest. In other words, one can say that regions compete over each contour point. For this reason, the statistic forces generated by the two regions are often called *competition forces*. Analogously, when the two regions that most decrease the total energy (3.7) are merged, this action is considered to be accomplished by *merging forces*. The contour flow by equation (3.8) is illustrated in Fig.3.4. Shown are a contour $C_{i,j}$ between regions R_i and R_j, and the statistics forces corresponding to the regions. Smoothing forces F_k at points of high and low curvatures are also shown. All forces act in the direction of the contour's normal.

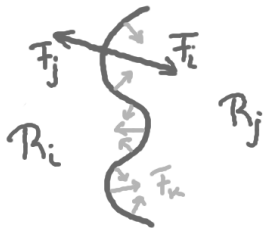
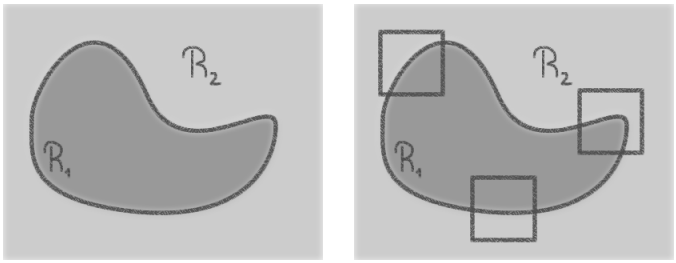


Figure 3.4.: Force illustration.

Force computation involves the unknown parameters θ_i. In order to determine them, some assumptions about the image need to be made. The assumptions about the image are reflected in the model chosen to estimate probability densities and the observations selected to estimate their parameters. The most common assumption about the image is that features are normally distributed in each image region (Paragios and Deriche, 2002a; Rousson and Deriche, 2002; Chan and Vese, 2001; Jehan-Besson and Barlaud, 2003). Other parametric distribution may be assumed, especially when - otherwise not too complex - images are very noisy and the type of noise is known. For instance, a Gaussian Markov random field model was employed to segment an image in (Chakraborty et al., 1996), while (Chesnaud et al., 1999) experiment with probability densities of the exponential family.

Arbitrary densities of real world objects often cannot be approximated with parametric density models. In these situations nonparametric density estimation methods may be more suited, as discussed in Chapter 2. Widely used non-parametric methods are smoothed histograms and Parzen density estimation (Brox and Weickert, 2004a; Rousson et al., 2003; Kim et al., 2005; Herbulot et al., 2004). When nonparametric estimators are employed, the statistics force is not always generated as the logarithm-of-probability, like in Eqn.(3.8). Instead, it relies on similarity measures between pdfs. Similarity is quantified by computing a divergence measure between the



(a) (b)

Figure 3.5.: Acquiring global and local information. (a) Image regions for acquiring global region information. (b) Rectangular image parts for acquiring local region information.

histograms of regions, like the Kullback-Leibler divergence or the Battacharya distance (Freedman and Zhang, 2004), or the Wasserstein distance (Chan et al., 2007). The statistics force that merges two regions is largest between the two regions that are most similar.

Independent of the type of model employed for pdf estimation, the pdf over a set of chosen feature describes the region. The ensemble of chosen features and their estimated pdf is sometimes called a *region descriptor*. Region descriptor will be specified more accurately in Chapter 4, but for now this definition suffices.

3.1.3. Local and Global Region Information in Contour Evolution.

We distinguish between two types of region descriptors, depending on the size of the of the image part modeled by the pdf. Most often, the pdf associated with an image region summarize the variation of features within the entire region, because it is computed from all observations in the region (Chan and Vese, 2001; Cohen et al., 1993). This is illustrated in Fig.3.5(a). The information contained in the pdf is global in nature and omits the spatial relations between pixels. For example, the sky immediately after sunset is globally described as blue. However, looking at it in more detail, one observes that it is dark blue above one's head and light blue where the sun just set. If details are to be described accurately with pdfs, statistics have to be computed over image parts smaller than the entire region, in order to capture local information. A possible type of local image patches is shown in Fig.3.5(b). Methods based on local information are much less encountered than the ones based on global information. We review the main methods that use local information in the following.

The region-competition method proposed by Zhu and Yuille (1996) relies in principle on the global information contained in the pdfs $p(\{\mathbf{f}(\mathbf{x}) : \mathbf{x} \in R_i\} \,|\, \theta_i)$. However, they have noticed that local information can increase the confidence about a pixel's true region if the joint distribution

of pixels in a local window $W(\mathbf{x})$ centered on pixel $\mathbf{x}$ is considered, instead of the pdf at pixel $\mathbf{x}$. This means that in the energy equation 3.7, the logarithm of the pdf p_i is replaced with the joint pdf of the w pixels in window $W(\mathbf{x})$, i.e. $\frac{1}{w}\iint_{W(\mathbf{x})} \log p_i d\mathbf{x}$.

The feature vector $\mathbf{f}(\mathbf{x}) \in R^d$ is assumed to have a multivariate normal distribution within region R_i. The density has a global mean denoted as μ_i, a covariance matrix denoted as Σ_i, a local mean denoted as $\bar{\mathbf{f}}$, and a local variance denoted as S. Means and variances are given by the formulas

$$\mu_i = \frac{1}{n}\sum_{\mathbf{x}\in R_i} \mathbf{f}(\mathbf{x}), \tag{3.9}$$

$$\Sigma_i = \frac{1}{n}\sum_{\mathbf{x}\in R_i} (\mathbf{f}(\mathbf{x}) - \mu_i)(\mathbf{f}(\mathbf{x}) - \mu_i)^T. \tag{3.10}$$

The joint probability of pixels in window W is written in terms of these multidimensional parameters as (Zhu and Yuille, 1996)

$$\begin{aligned}\prod_{j=1}^{w} p_i(\mathbf{x}_j) &= \frac{1}{(2\pi)^{w/2}\|\Sigma_i\|^{w/2}} \exp\left\{-\frac{1}{2}\sum_{j=1}^{w}\left(\mathbf{f}(\mathbf{x}_j) - \mu_i\right)\Sigma_i^{-1}\left(\mathbf{f}(\mathbf{x}_j) - \mu_i\right)^T\right\} \\ &= \frac{1}{(2\pi)^{w/2}\|\Sigma_i\|^{w/2}} \exp\left\{\frac{w}{2}\left(\bar{\mathbf{f}} - \mu_i\right)\Sigma_i^{-1}\left(\bar{\mathbf{f}} - \mu_i\right)^T + tr(\Sigma_i^{-1}S)\right\}.\end{aligned} \tag{3.11}$$

With this arsenal, the statistics force generated by region R_i in Eqn. (3.8) is deduced (Zhu and Yuille, 1996)

$$f_i = -\frac{1}{2}\left(\log(2\pi) + \log\|\Sigma_i\| + \left(\bar{\mathbf{f}} - \mu_i\right)\Sigma_i^{-1}\left(\bar{\mathbf{f}} - \mu_i\right)^T + tr(\Sigma_i^{-1}S)\right). \tag{3.12}$$

Theoretically, the feature vector $\mathbf{f}$ may have any number of components. We have already discussed that, in practice, this is rarely feasible, and indeed, in their paper, Zhu and Yuille (1996) have successfully experimented with two dimensional features for color and texture.

A modification of the region-competition method, one which loosens the Gaussian assumption, is proposed in (Kadir and Brady., 2003). To this end, the global pdfs are estimated nonparametrically, but local regions are still modeled with normal distributions. Unlike the region-competition method, the local information is employed not only in computing competing forces, but also in computing merging forces.

The anticipating snake introduced in Ronfard (1994) is another pioneering work that involves local information. The particularity of this work is that information for generating statistics forces is mainly local. Also unusual, the method employs a parametric curve representation to evolve a region-based active contour. The curve partitions the image into an inside- and an outside-region. The principle is to start with a possible deformation of the contour that leads to a region deformation δR, in either the inward or the outward direction, in order to generate a

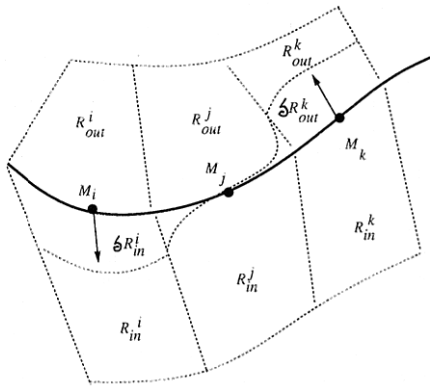


Figure 3.6.: Deformation hypothesis for contour evolution. (Source:(Ronfard, 1994))

region force. In the particular implementation provided by Ronfard, the contour deformation δR is divided into sections. For this purpose, the parceling in Fig.3.6 is created. The local regions are image strips aligned with the normal to the contour, and the deformation δR is also a rectangular window aligned with the normal.

The region force generated by the region variation δR is computed using the Ward distance. To compute the Ward distance, one must first compute the sum of squared errors between the original image intensity, $I(x,y)$, and a statistical model of the region, $I_{model}(x,y)$, over all pixels in a given region R, $W(R) = \int_R (I(x,y) - I_{model}(x,y))^2 dxdy$. The Ward distance between two region R_1 and R_2 represents the amount of energy needed to merge the two regions, and it is defined as $D(R_1,R_2) = W(R_1 \cup R_2) - W(R_1) - W(R_2)$. The region force equals the difference between two Ward distances $D(R_{in}, \delta R) - D(R_{out}, \delta R)$. Intuitively, the force tells us if δR should be merged with R_1 or with R_2.

Active contours based on local information only, have recently been given more attention (Piovano et al., 2007; Lankton et al., 2007; Li et al., 2007; Mansouri and Mitiche, 2002). A more detailed discussion follows in section 4.1 and 4.2. Here we point out the important results from (Brox and Cremers, 2007). If only local information is considered, and it is modeled with a normal distribution with fixed variance, the Mumford-Shah functional (3.5) is equivalent to the statistic functional (3.7). A piece u_i of the smooth approximating function u is then equal to the mean of the normal pdf p_i.

3.1.4. Contours Based on Combined Edge and Region Information

Intensity information gained from regions can be combined with edge information to create active contours that impose less constraints on the segmented image. The question is how to construct the associated energy functional. A straight forward solution is to sum a weighted edge-

term with a weighted region-term (Cohen et al., 1993; Paragios and Deriche, 1999b, 2002b; Ecabert and Thiran, 2002; Besson et al., 2000). A functional obtained as a boundary integral over local region integrals was proposed in (Lankton et al., 2007). The resulting curve flow has two components: a weighted curve shortening flow similar to the first term of the geodesic active contour (3.4), and a region flow similar to the region-competition term in (3.8). The difference is that the partial differential equation for updating the curve C depends on two arc-length parameters: the first traverses the curve, the second localizes the region information.

Other solutions for constructing energy functionals are nonlinear. The region term, written as a region integral, is divided by the boundary term, written as a boundary integral in (Jermyn and Ishikawa, 1999). The weighting parameters present in linear energies disappear from the energy proposed in that method, and the length of the curve is optimized in relation to its area, with the potential for producing smoother curves. Intuitively, this can be understood by remembering the circle, which has maximal area reported to its boundary length. As a distinctive feature, the authors suggest that the method can find the global optimum for an image.

Bresson et al. (2007) also propose a solution to finding the global optimum for a contour energy by combining region and edge information. The idea of the method is to minimize the length of a geodesic active contour while approximating the image with a piecewise constant function (using the L_1 norm in the ROF model). The paper is particularly interesting because it proposes a fast alternative to the standard minimization based on partial differential equations, which does not require setting the position of the initial contour. The so called *dual formulation* represents the function u as the sum of two artificial matrices. One captures the constant or region component of u, while the other capture the noise or texture component of u. Instead of iteratively updating u according to a pde, in each iteration, each matrix is computed while considering the other one constant. This means that the geometry (the regions) of the image are approximated considering the amount of noise known. The amount of noise is then determined considering the geometry known. The exact numerical scheme for the iterative process is given in (Bresson et al., 2007).

3.2. The Derivation of Motion Forces

For the majority of level-set methods for image processing, the flow equation is derived by gradient descent. The gradient of the energy is determined by the Euler-Lagrange equations. For this purpose, probability densities may be considered constant or variable. When probability densities are involved, the derivation may thus occur in one of the following manners.

One can take an expectation maximization approach (Dempster et al., 1977), in which the regions R_i, and thus probability densities p_i, are considered to be constant in time when writing derivatives. Once the level-set function has been updated, new probability densities p_i are computed considering the contour to be fixed Zhu and Yuille (1996); Chan and Vese (2001); Samson et al. (2000). This type of derivation has been employed to obtain flows (3.8) and (3.18) from energies

(3.7) and (3.17).

Alternatively, one can take into consideration that pdfs of region descriptors do in fact depend on the regions R_i. The flow can then be derived using shape-derivation tools as shown in (Aubert et al., 2003; Jehan-Besson and Barlaud, 2003). The resulting flow has additional terms when compared to the flow obtained in an expectation maximization manner. The additional terms may improve the segmentation, and, at the same time, they increase the complexity of the implementation and its computational cost. Shape derivatives are most often used in combination with the level set framework, which is also described latter in this chapter. Rousson and Deriche (2002) have shown that these terms reduce to zero if pdfs are considered to be Gaussian and that they are negligible when pdfs are assumed to be Laplacian (Heiler and Schnoerr, 2003). In this thesis, we opt for the expectation maximization manner of deriving the flow, because accurate segmentation results can be obtained without the additional terms, and the evolution is faster.

3.3. Features for Describing Objects

The overwhelming majority of references discussed up to this point propose active contours based on pixel intensity or color. However, in order to deal with complex images, additional information is typically needed.

The image segmentation task is a very difficult one, partly because the raw image data is very difficult to interpret. Raw image data consisting of intensity or color information often does not provide sufficient information to achieve correct segmentation. Human vision excells in integrating multiple cues in order to solve this difficult problem. Inspired by biological vision systems, researchers have worked on integrating into machine vision systems cues or features, like shape, texture and motion. The active-contours framework is well suited for integrating multiple sources of information into the contour's evolution.

3.3.1. Integrating Texture, Motion, and Other Features

Texture and motion are very important visual cues that help biological systems identify objects in the world (Regan, 2000). "Texture is a phenomenon that is widespread, easy to recognize and hard to define" (Forsyth and Ponce, 2003), which is why a standard definition of texture does not exist. We explore the concept of texture as a regularity and a constancy of the appearance of a surface that does not have a block-color. Quantifying texture for computational purposes has been a constant topic of research in computer vision, because texture is constant in an organic sense, not a mathematical one. For the overwhelming majority of contours, texture is considered a random vector (Zhu and Yuille, 1996; Sandberg et al., 2002; Popat and Picard, 1997; Varma and Zisserman, 2008; Wolf et al., 2006), and it is integrated into the energy as a statistical data

term (by means of the pdf), like in equation (3.7).

There are mainly three types of texture-quantifying methods, and they can all be found in the active contours segmentation framework. A texture feature vector may originate in filtering the image with a filter bank (Paragios and Deriche, 1999a, 2002a), or it may be a small linearized image patch (Awate et al., 2006). Alternatively, a structure tensor is defined from the image gradients. The structure tensor usually undergoes a nonlinear diffusion process to smooth image noise without smoothing edges before it functions as texture quantifier (Brox and Weickert, 2004b; Brox et al., 2003a; Rousson et al., 2003).

Biological vision systems perceive continuous motion. The brain assumes that objects change minimally from frame to frame, both in their positions and their appearance. Researchers make the same assumption when they design methods for determining the apparent motion of objects, known as optical flow. Some optical flow algorithms aim at determining the motion of individual points between frames. Corresponding points can be found by starting with the result from the previous frame, and searching for edges along the normals to the contour in the current frame (Isard and Blake, 1998b,a); due to problems that accompany edge detection, the shape of the tracked object needs to be constrained. Ideas for estimating the motion vector field have been introduced in a few seminal papers (Lucas and Kanade, 1981; Horn and Schunck, 1981). In the framework of active contours, the motion vector field can be employed to set the initial position of the contour in a frame, knowing its position in the previous frame (Ha et al., 2004). Also, motion vectors can be modeled statistically, like any other pixel feature. For example, vector components have been integrated with texture and color to obtain robust tracking methods that have been applied to traffic scenes (Brox et al., 2003a; Clark and Thomas, 2001).

There are mainly two different ways to apply statistics for determining motion in active contours. The first relies on modeling the background and the foreground of the scene. For example, if it is possible to learn a model of the background image, this image is subtracted from every frame in order to determine the foreground. The outcome is most often not the desired one because it does not contain only foreground pixels. One can then try to learn a pdf for static pixels and one for dynamic ones, and subsequently use these pdfs in an active contour (Paragios and Deriche, 2000; Yilmaz et al., 2004; Besson et al., 2000; Mansouri, 2002). The second is to find those regions in two frames that are delimited by active contours, and that have most similar statistics. In (Chang et al., 2005a) similarity was evaluated with the mean-shift procedure, while in (Freedman and Zhang, 2004) the Kullback-Leibler divergence was proposed for this purpose.

An alternative class of optical flow methods aims at estimating the motion of an object as a whole, while making assumptions about the law of motion followed by the object. Parameters of the law of motion are usually determined by minimizing the least-squares-error between the image predicted by the model and the observed image using an active-contours flow (Paragios and Deriche, 2005; Dorettoy et al., 2003). The optimization problem is increasingly difficult, as motion models have to grow more complex and the number of parameters grows, similar to the optimization problems posed by integrating shape information. Thus, the simplest type of model

that can be integrated into active contours is rigid or affine motion (Paragios and Deriche, 2005). A more complex Gaussian Markov Random Process model for the arrangement of pixels in time and space was employed for tracking textured objects, like smoke or fire on water (Dorettoy et al., 2003).

Optical flow, and other types of information may be abstracted as vector or tensor fields, which in turn can be viewed as vector valued images to be segmented. Often, the goal of the segmentation is to delineate regions of vectors that point mainly in the same direction. Possible applications are object segmentation from motion information only (Roy et al., 2006, 2003), or the segmentation of brain structures in diffusion tensor images (Wang and Vemuri, 2004).

Some authors have proposed to generate forces from image data by a physical-process model (Jalba et al., 2004). For instance, Xiang et al. (2005) use long range elastic interactions between image edges to let contours detect thin, weak blurred structures like blood vessels. Edge points and the evolving contour can be seen as electrostatic particles that generate interaction vector fields between the two entities; as the contour evolves, the interaction vector field is updated with the goal of robustifying the evolution (Yang et al., 2006).

In (Xie and Mirmehdi, 2006, 2008), the authors hypothesize the existence of magnetic forces between the active contour and object boundaries detected with edge filters. Due to the magnetic field, edges can interact across the image, providing global information in the segmentation process. These types of contours can easily close small gaps caused by weak edges, but they are still negatively affected by the problems of edge detectors.

3.3.2. Integrating Shape Priors

The shape of an object is a very important clue that helps detecting it in a visual scene. Shape reflects prior knowledge about the world. Depending on the amount of knowledge about the object, one can distinguish between *geometric priors* and *specific priors*. Geometric priors, also known as *generic priors*, mathematically formalize the general properties of shape. Specific priors mathematically formalize the shape of a particular object.

Generic (geometric) priors. All active contours include a smoothness constraint that prevents the curve from forming sharp corners, as pointed out, for example, for the region competition model in Section 3.1.2. The constraint $\int_C ds$ reveals that the curve is required to have minimal length. If data terms are ignored, the fastest curve flow that minimizes the length of the curve is in the direction of the curve's normal at a speed proportional to the curve's curvature. This flow will drive any contour into the shape of a circle which shrinks until it disappears. A mathematical derivation of this flow in terms of level sets can be found in Appendix A.

Another common generic prior is a maximal-area constraint. This constraint takes the form $\int_{R_i} dx$, and if forces the region enclosed by the contour C_i to occupy the maximum possible

space. This constraint adds a balloon force to the other forces that determine curve flow and helps drive the contour out of local minima (Cohen, 1991; Chan and Vese, 2001).

Recently, *higher order active contours* have been introduced in (Rochery et al., 2006). The shapes of these contours are constrained by generic priors that model long-range interactions between contour points. The concept is exemplified for the segmentation of thin structures. For thin structures, in a small neighborhood, points have either (relatively) parallel or antiparallel normal vectors, as illustrated in the figure.

Normal vectors

Interactions between every two points on the curve are expressed using tangent vectors $\vec{\mathbf{t}}$, which are perpendicular to the normal vectors. Mathematically, the geometric prior energy-term E_g associated with the contour is a double integral of quadratic form

$$E_g(C) = \alpha \underbrace{\int_C ds}_{Length} - \beta \underbrace{\int_R dx}_{Area} - \gamma \int_C \int_C \vec{\mathbf{t}} \cdot \vec{\mathbf{t}}' \Psi(||s - s'||) ds\, ds'. \tag{3.13}$$

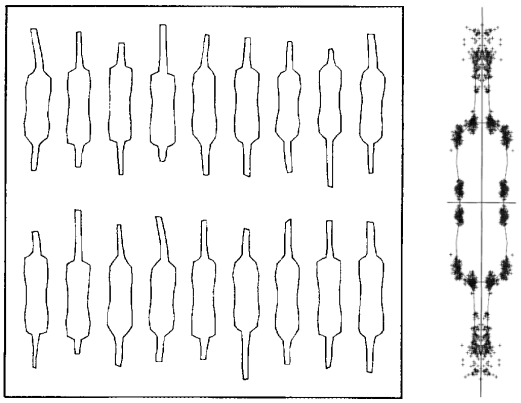
As previously, s is the arclength parametrization, and $||\cdot||$ represents is the Euclidean distance. Function Ψ is a real valued function chosen such that $\Psi(x) \to 0$ as $x \to \infty$, due to the fact that local interactions should be emphasized over global ones. Otherwise, the geometric prior depends on the scale (size) of the image, which is an undesired property. One can note that the linear length and area terms can be added to the quadratic term to enforce all three constraints.

The data term in the energy of a higher order active contour is modified to reflect interactions between points, in a manner analogous to the geometric prior term. For instance, a data term that depends on the gradient of the image, $|\nabla I|$, is written as

$$E_d(C) = \int_C \vec{n} \cdot \nabla I\, ds - \int_C \int_C \vec{\mathbf{t}} \cdot \vec{\mathbf{t}}' (\nabla I \cdot \nabla I') \Psi(||s - s'||) ds\, ds'. \tag{3.14}$$

The first term encourages the normal to be opposed to the image gradient, while the second allows the data and the contour to interact nonlinearly. Applications of these complex generic priors include the segmentation of elongated thin structures like roads in aerial maps (Rochery et al., 2003) and blood vessels (Nain et al., 2003), or circular structures like tree crowns (Horvath, 2007).

Finally, the active contour may be forced to encourage shapes that are similar to an initial contour's shape by minimizing the distance between the two shapes. This method is applicable to tracking objects that minimally change shape from frame to frame (Gastaud et al., 2004). For this purpose, a distance measure between shapes needs to be defined taking into account the way shapes are represented. The method is very adequate for shapes represented as distance transforms, as is the case when curves are represented implicitly in a level-set implementation.



(a) (b)

Figure 3.7.: Active shape models ingredients: (a) learning examples, (b) mean shape with overlaid example landmarks. (Source: (Cootes et al., 1995))

Specific priors. An intuitive way to learn the shape of an object is from examples. The active shape model is a framework that is widely used for learning specific priors (Cootes and Taylor, 1992; Cootes et al., 1995). In essence, the method computes an average shape and the degree in which examples differ from this average from a set of aligned examples. The same principle can be applied to learn the appearance of an object (Cootes and Taylor, 1992). Fig.3.7 (a) shows a set of *aligned* example-shapes for resistors. The fact that the shapes are aligned means that they have approximately the same orientation and size. To learn an average shape, we need to pick the same number of landmarks (special points) from each example shape and index it. Landmarks with the same index constitute a set of *corresponding* landmarks. The average shape, exemplified in Fig. 3.7(b), is obtained by computing the average position of each landmark. The degree to which the average shape varies is illustrated by the scatter of example landmarks overlaid on the average shape. The variation is quantified by computing the covariance matrix between the sets of corresponding landmarks.

Specific priors are typically discussed taking into account the contour representation, because the latter guides the selection of landmarks. Parametric models have a very large computational advantage over level sets when integrating shape priors, because the representation is much more compact, since parametric models are in fact a chain of particles. In comparison, level sets offer a dense representation in the form of distance transforms of the contour. In consequence, a large number of landmarks points must be selected in order to capture the information necessary for learning a good shape model. The landmarks must be uniformly scattered over the region represented by the distance transform.

Landmarks can be used for parametric representations. They are thus suited for real time tracking of objects with relatively constant shape (Cremers et al., 2002). For the same reason, shape alignment can be achieved much faster with explicit than with implicit representations. For an excellent treatment of parametric representations see (Blake and Isard, 1998; Isard and Blake,

1998a,b).

The implicit representation allows a straight forward computation of the mean shape. Covariance matrices can be seldom computed because the number of landmarks involved is so large. As a result, a mean shape plays the role of a template shape to which a segmented shape is compared using a distance measure. The larger the distance, the less the found shape looks like the template shape and the energy term associate with the shape prior is larger. Computed distances are informative only if the two shapes are aligned. To align shapes, one must have a model of the geometric transformation that connects them. The transformation's parameters are determined by minimizing the distance between the two shapes. However this minimization problem is not symmetric, meaning that the mapping that transforms shape A in shape B, is, in general, different from the mapping that transforms shape B in shape A. Additionally, the problem may have many local minima and it involves nonlinear optimization, which is particularly difficult. For these reasons, the selected transformation model is often rigid (Rousson and Paragios, 2002; Chen et al., 2002; Rousson and Paragios, 2008).

Alignment is important, for instance, in object recognition and image understanding. In many applications, manually positioning the template over the object to be segmented seems advantageous, since accurate segmentation has the highest importance. A trade-off between dense representation and ease of manipulation must be analyzed at times. For example, in medical applications, organ shapes may benefit from the dense distance transform representation, but it is unclear which transformation best models the variation of the shapes of different individuals' organs. The integration of shape priors into level-set methods is thus an actively studied topic, and we present here some state-of-the-art methods.

Pohl et al. (2006) proposed the concept of Logarithm of Odds (LogOdds) for shape representation, with application to the segmentation of brain structures. The method starts by computing probability maps from distance transform examples. These probabilities are used to construct a log odd maps. Unlike the mean shape, which is a linear combination of learning examples, the log odds map is nonlinear in nature, due to the nature of the distance transform and the fact that the probability maps can be estimated nonparametrically. The log odd maps have the advantage that they exist in a vector space, which means that the maps can be added and multiplied. The construction of a distance measure between template and segmented shape is eased considerably. Overall, prior shape knowledge can be captured more accurately. To achieve segmentation with a level-set algorithm, the level set itself is a log-odds map, and the smoothness term is also defined in a probabilistic manner (Pohl et al., 2007). The resulting method is very powerful and has been applied to the segmentation of brain structures. However, the appearance of objects should be such that it can be modeled well with a normal distribution.

Shape models have a good potential to grow more and more complex. For example, prior shape may be combined with motion information for tracking deformable objects in video sequences. These problems become interesting when the objects are deformed nonlinearly. Examples that are often researched are the deformation of hearts in ultrasound images and deformations of sil-

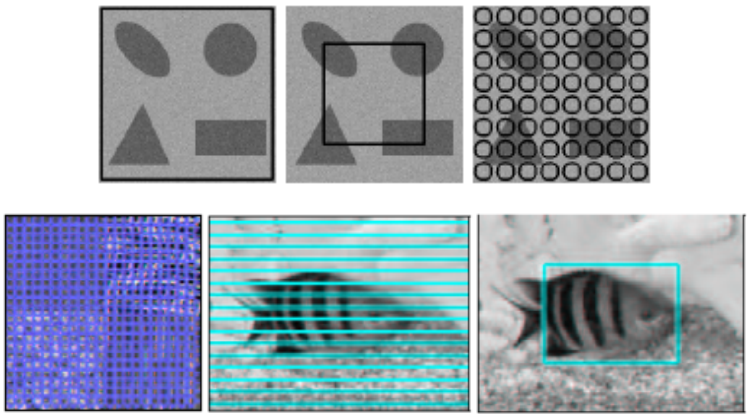


Figure 3.8.: Automatic initializations - first, third, fourth and fifth images - and initializations obtained from user interaction - second and sixth images. (Top image source: (Paragios, 2000); bottom image source: (Brox et al., 2003b))

houettes of walking people. Cremers and Rousson (2007) thus propose to approximate the complex statistics of a set of moving heart shapes or moving silhouettes employing Parzen-windows density estimation. In a further development, the resulting shape priors are integrated into a motion model of Markovian type (Cremers, 2008), where the probability of the current shape is conditioned on the shape observed at a previous time. This means that the parameters of transition probabilities between shapes must be estimated. The model ends up with and optimization problem over a very large number of parameters from a vast amount of data. Such problems are well known for being extremely difficult to solve, or very sensitive to initial conditions.

3.4. Active Contours Initialization

The active-contours segmentation process starts with an initial contour obtained automatically or with the help of user interaction. Automatic initialization is pursued in the larger frame of artificial vision research, a field with myriad of engineering problems in need of a solution. For many computer vision applications, it is absolutely unproblematic to require the user for minimal interaction. Through interaction, the user provides valuable information for the segmentation process. In many cases, segmentation fails when this information lacks. On the other hand, even when the user draws a circle or a square, as shown in Fig. 3.8, segmentations depend on the initial contour's position. Initialization is thus a very important issue in the active contour framework. All other properties of the active contour method being constant, the segmentation of an image may succeed starting with one initialization, while failing for another.

Initialization sensitivity is a problem especially for edge-based active contours. A corresponding amount of research has dealt with finding remedies for this problem. Each method solves some

problem which might occur in reality. However, the real world has a plethora of ways for not being conform to researchers' assumptions.

The range of capture of original snakes was enlarged placing the contour in a vector field obtained from the contour's distance transform in (Cohen and Cohen, 1993). A constant balloon force meant to shrink or grow the contour was proposed in (Cohen, 1991). The edges of an object generally have different strengths, and a constant balloon force may lead to missing some of them.

The range of capture is significantly enlarged by diffusing the edge map of an image (Xu and Prince, 1997, 1998). A vector field, known as *gradient vector flow*, is obtained in which vectors point toward the nearest boundary. As a result, snakes driven by this gradient vector flow can be initially positioned such that they cross the boundary. However, when the contour is tangent to the force vector, the gradient vector flow cannot converge to the correct solution. An attempt to alleviate this problem involves adding a balloon force to the gradient vector flow (Paragios et al., 2004).

In (Li et al., 2005a), an edge-preserving gradient vector flow is first segmented with a graph partitioning algorithm, yielding regions of interest around objects. The boundaries of these regions constitute initial snakes made to shrink toward objects. This method fails when snakes are initialized within or across object boundaries.

The initialization problem is not as severe with region-based active contours, since this type of contours relies on global image characteristics. Automatic initialization is encountered much more often for this type of contours (Brox and Weickert, 2004a; Paragios and Deriche, 2002a). In order not to favor a local minimum over another, the requirement is that initial patches are evenly distributed across the image, as shown in some of the examples in Fig. 3.8. Even so, one can never guarantee that the correct segmentation will be reached from automatic initialization. The segmentation result can only be determined by experimenting, and, probably, tuning parameters.

3.5. Methods for Flow Implementation

The speed and quality of segmentation with active contours largely depend on the implementation of the theoretically derived flow. Implementations may rely on graph algorithms, or, more often, curves are represented parametrically, or as level sets. The first two alternatives are shortly discussed, before level-set methods are presented.

An image can be transformed into a graph, where pixels are nodes, and differences between pixels, e.g. image gradients, are edges (Jermyn and Ishikawa, 1999). For instance, the weight of an edge can be set inverse proportional to the gradient magnitude. Well-studied graph algorithms can then be used to find the shortest way between a starting and an ending point. Both points

can be interactively selected, as is the case with intelligent scissors (also known as livewire) (Mortensen and Barrett, 1995), a tool that requires intensive interaction, but offers very good results.

A large class of algorithms model images as connected graphs and use graph-theoretic algorithms to solve the segmentation problem, as described, for example, in the seminal paper (Shi and Malik, 2000). Boykov et al. (2001) extend graph-methods to address the minimization of a large class of energies containing smoothness constraints with graph-cuts algorithms; the algorithms find approximate solutions very fast. These graph-cut algorithms are employed for active contours energy minimization in (Ning Xu and Ahuja, 2003; Kim and Hong, 2007). In the first reference, the active contour is dilated to obtain an outer and an inner boundary, that represent sink and source nodes in a graph. The new position of the contour is determined by the minimal cut in the graph, and it can overcome minima, but the size of the neighborhood obtained by dilatation is be user-specified. The second reference proposes to remove initialization, similar to (Jermyn and Ishikawa, 1999), by successively finding minimal cuts in partitions obtained in a previous step, until the energy does not further decrease. It is interesting to note here, that the authors of (Jermyn and Ishikawa, 1999) and (Kim and Hong, 2007) claim their algorithms find the global energy minimum for a given image, due to graph algorithm properties. The shortest cuts, however, do not always yield accurate segmentations.

Referring to active contours, one generally expects algorithmic implementations to be parametric snakes or level-set methods. Parametric snakes (Kass et al., 1988; Zhu and Yuille, 1996; Ronfard, 1994; Brigger et al., 2000; Blake and Isard, 1998) represent curves as chains of particles. Particle positions are iteratively updated according to the partial differential equation describing the curve flow. At every step, curve points are interpolated between particles, using for example radial or B-spline functions. Handling snake points may be a daunting task, because particles may come too close together or grow too far apart causing numerical problems. Also, it is very difficult to handle topological shape changes, like the merging and the splitting of contours. On the other hand, particles are a less dense representation which allows for faster computation and easier integration of shape-prior knowledge, as mentioned in subsection 3.3.2.

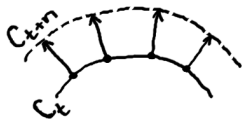


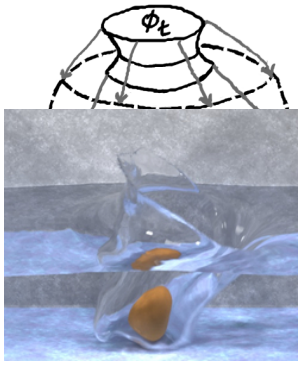
Figure 3.9.: Chain of particles.

For both parametric snakes and level-sets, the partial differential equation for the flow can be solved with the Finite Difference Method. For snakes, the larger the distance between particles, the less the amount of information from the image that flows into the equation. For a coarse sampling, the problem can be alleviated with Finite Elements Methods, where one works with continuous functions, generated from a basis of functions to pass through the particle chain (Cohen and Cohen, 1993). Further mathematical issues concerning the equivalence between snakes and level-set methods can be found in (Xu et al., 2001).

Before attending to the level-set method, we may note that designing implementations for active

contours flows is an active field and alternatives are often proposed. We mention here two examplesS. In the first one, stochastic dynamics were adopted for interface propagation in order to deal with local minima (Juan et al., 2006). In the second example, global minimizers for edge-based active contours are proposed for images approximated with a piecewise smooth function (Chan et al., 2004; Bresson et al., 2007). These methods assume that the piecewise smooth approximation is a good one. In the second reference, a fast numerical scheme for minimization is based on introducing noise into the energy functional. Instead of solving the partial differential equation to determine the piecewise smooth image approximation, two easier equations are solved. In each iteration, the amount of noise is evaluated first, and then the piecewise smooth function is approximated taking the calculated noise into account.

3.6. Level-Set Methodology



Simulation using level sets

The level-set method was proposed by Osher and Sethian (1988) as a solution to the problems of mathematically modeling moving fronts and simulating the motion using computers. The method is employed by large communities of researchers from computer graphics to image processing, due to some very appealing properties. First, large changes in front shape, including merging and splitting, are intrinsic to the method. Fronts may completely disappear, or new ones may be created and these situations also need not be handled separately. One fully appreciates these intrinsic properties, when faced with the task of simulating a stone splashing into water, like in the excerpt in the figure (Fedkiw, 2008). Second, the numerics of front evolution are easy to implement, and increasingly faster algorithms have been designed for this purpose. As a result, the level-set method has enjoyed increasing popularity in image processing fields. It is the method of choice in this thesis. As such, this section is dedicated to presenting the method and its importance in active contours segmentation. Finally, the ideas of this thesis are all implemented with a recent level-set algorithm, and thus this algorithm be described as well.

3.6.1. The Level-Set Method

Mathematically, a 2D-curve is a continuous map from a one-dimensional space to an 2-dimensional space. It can be represented parametrically (or explicitly) by specifying the coordinates $x(s)$ and $y(s)$, i.e. $C(x(s),y(s))$. Curves can also be defined implicitly, taking the form $C(x,y) = 0$. A curve that varies in time is written explicitly as $C(x(s),y(s),t)$. The same curve can be represented implicitly, embedding the curve into a function ϕ defined on a 3-dimensional space

$$C(t) = \{(x,y)|\phi(x,y,t) = 0\}. \tag{3.15}$$

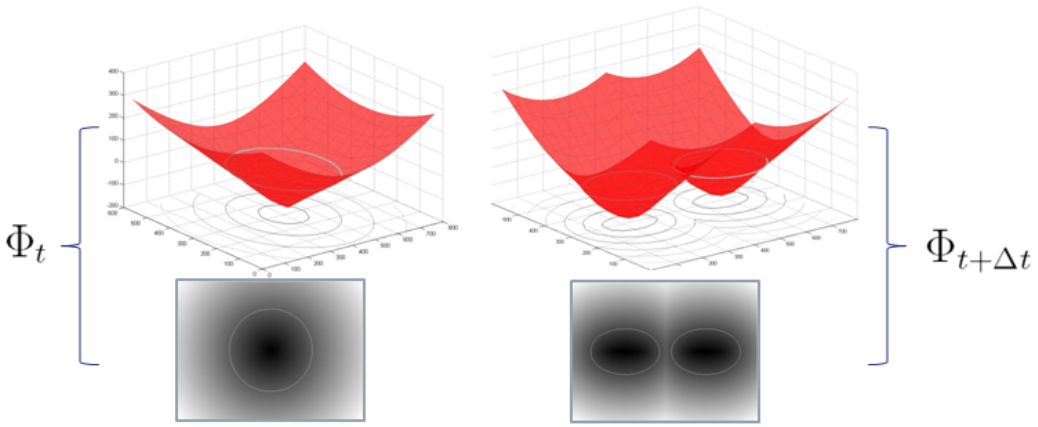


Figure 3.11.: Level-set representation of a curve. The surfaces show level-set functions. A level-set function is, in general, the signed distance transform of the curve, as depicted in the lower half of the image. The curve may change topology, as the level-set function is updated from the image on the left, to the image on the right.

At a fixed moment in time t_0, the set of points (x,y) that evaluate to the same value c, i.e. $\phi(x,y,t_0) = c$ constitutes a level-set for the surface $\phi(x,y,t_0)$. $\phi(x,y,t)$ is named the level-set function and the curve is the zero level-set of this function at any moment in time. The surface on the left of Fig. 3.11 depicts a level-set function at time t. Cutting the surface with the xy-plane at hight zero, one obtains the evolving curve in the shape of a circle. On the right, the level-set function is shown at time $t+n$, after n iterations. One can observe that the shape of the surface has substantially changed, and the curve has split into two circles at the zero level-set.

For implementation purposes, the level-set function is taken to be the morphological distance transform, see, for example, (Tönnies, 2005). The distance transform of a binary image is a real-valued image, where the value of each pixel is the distance between the pixel itself and closest nonzero pixel in the binary image. The distance transform of two contours is shown as 3D surface and gray image in Fig. 3.11. Conventionally, distances are negative inside and positive outside the contour.

The level-set function is initially set such that $\phi(x,y,0) = C_0$, where C_0 denotes the initial curve. An active contour is moved in the direction of its normal, and it involves, in general, the contour's curvature. Both the normal and the curvature can be expressed in terms of the level-set function ϕ, taking into account the sign definition for the function, as shown in Appendix B. In this thesis, the signed distance function ϕ is taken to be positive on the inside and negative on the outside of the curve. Let us denote the time derivative of the curve C and the level-set function ϕ with ϕ_t and C_t, respectively. A partial differential equation of the form $C_t = F \cdot \vec{\mathbf{n}}$, where $\vec{\mathbf{n}}$ denotes the outward normal, can be written as $\phi_t = F(-|\nabla\phi|)$. A partial differential equation of the form

$C_t = k \cdot \vec{\mathbf{n}}$, where k denotes the curvature can be written

$$\phi_t = div\left(\frac{\nabla\phi}{|\nabla\phi|}\right)|\nabla\phi|. \tag{3.16}$$

3.6.2. Formulations Based on Level-Sets

Level sets were introduced into the variational approach to image segmentation with edge-based functionals (Caselles et al., 1993; Malladi et al., 1995; Caselles et al., 1997), as already mentioned in subsection 3.1.1. The level-set function ϕ was defined as the signed distance transform of the active contour. For region-based contours, a contour's energy and flow were formulated with the help of level sets in (Chan and Vese, 1999, 2001) as the special two-region case of equations (3.7) and (3.8) as follows

$$E(\phi, p_1, p_2) = -\iint_R \left(\underbrace{H(\phi)\log p_1 + (1-H(\phi))\log p_2}_{Data\ term} - \underbrace{\mu|\nabla H(\phi)|}_{Smoothness\ term} \right) dxdy, \tag{3.17}$$

$$\frac{\partial\phi}{\partial t} = H'(\phi)\left(\log p_1 - \log p_2 + \mu\left(div\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\right)\right), \tag{3.18}$$

where $div(\cdot)$ is the divergence, $|\cdot|$ is the magnitude of a vector, and all other notations are as before. The function H is the Heaviside function,

$$H(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{3.19}$$

Since this function is discontinuous, its derivative $H'(x)$ needs to be defined; commonly, it is defined as the Dirac delta distribution: $H'(x) = \delta(x)$.

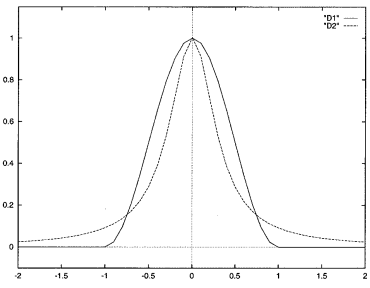


Figure 3.12.: Regularized versions of the Dirac delta with ε small (continuous line) and large (dotted line) (Chan and Vese, 2001).

Alternatively, to avoid introducing an infinite term at $\phi(x,y) = 0$, one can use a regularized

version H_ε of the Heaviside function and obtain its derivative $H'_\varepsilon = \delta_\varepsilon$, similar to Chan and Vese (2001), with a small $\varepsilon > 0$. The Dirac delta is then obtained for $\varepsilon \to 0$. The choice of ε has an influence on the behavior of the update equations. When $\varepsilon \to 0$, $H'(\phi)$ is non-zero only at points around the zero level-set, and thus the level-set function will be updated only at those points. This in turn means that existing curves can only merge and split.

When ε has a large value, $H'(\phi)$ is different from zero everywhere, and the level-set function is updated everywhere. This means that the function may change its sign outside a narrow band around the zero level set, and thus new contours are created. Setting ε to a large value is computationally much more expensive, since the level-set function must be updated for a larger number of points.

Concerning execution speed, one may also note that the curve normal need not be computed in order to apply the statistics force to the level-set. The gradient of the level-set function is needed only for curvature computation, which can then be optimized (Shi and Karl, 2005a).

Recently, some authors have dealt with the optimization of the functional over multiple regions. A large part of this progress is due to the level-set implementation which makes it possible to include the condition of non-overlapping complementary regions as artificial parameters (Aubert et al., 2003; Jehan-Besson and Barlaud, 2003; Paragios and Deriche, 2002a; Vese and Chan, 2002) or Lagrange multipliers (Zhao et al., 1996; Samson et al., 1999) in the energy functional. An simple mathematical formulation to this problem is given in Brox and Weickert (2004a). If each of N regions is represented by its own level set ϕ_i, the functional in (3.17) and its corresponding equation of motion can be written by analogy as follows (Brox and Weickert, 2004a):

$$E(\phi_i, p_i, N) = \sum_{i=1}^{N} \left(- \int\int_R \underbrace{H(\phi_i) \log p_i}_{\textit{Data term}} - \underbrace{\frac{\mu}{2} |\nabla H(\phi_i)|}_{\textit{Smoothness term}} \, d\mathbf{x} \right). \tag{3.20}$$

$$\frac{\partial \phi_i}{\partial t} = H'(\phi_i) \left(\log p_i - \max_{j \neq i, H(\phi_j) > 0} (\log p_j) + \frac{\mu}{2} \left(div \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \right) \right). \tag{3.21}$$

In this equation, one may note that the statistics forces generated by competing regions, $\log p_i$, $\log p_j$, uniquely assign pixels to regions. The smoothness term, however, can cause regions to slightly overlap, but this can be corrected in a post-processing step. Motion forces are derived either in an expectation-maximization manner, or using shape gradients, as described in Section 3.2. Equations (3.18) and (C.1) have been both derived using an expectation-maximization approach.

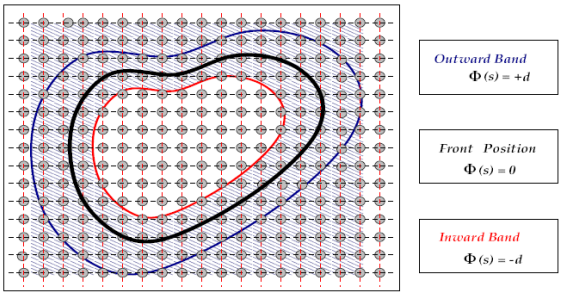


Figure 3.13.: Narrow Band algorithms principle. The finite difference numerical schemes are applied only for grid points in the narrow band (Paragios, 2000).

3.6.3. Implementations for Level-Set Methods

Many level-set implementations are based on Finite Difference Methods (Adalsteinsson and Sethian, 1995; Sethian, 1999). These require numerical approximations of the first and second order derivatives of the level-set function. Numerical schemes are effectively simplified when the function's gradient has value 1 everywhere. The reason for taking signed distance transforms of contours is that they have this property. Most often, the level-set function does not remain a distance transform, causing numerical instability. The solution is to reinitialize the function by recomputing the distance transform.

Finite difference implementations underly a constraint, known as Courant-Friedrich-Levy (CFL) condition, that the speed multiplied with the time step must be smaller than the size of a grid cell at every point on the curve in order to obtain a stable and accurate solver. The advantage is that boundaries are located with sub-pixel accuracy. The disadvantage is that the computational cost is, in general, very large. This adds to the the fact that reinitialization is time intensive. The problem is alleviated starting from the observation that one is often interested only in the position of the zero level set. This means that it is sufficient to solve the partial differential equation in a narrow band around the zero level-set (Adalsteinsson and Sethian, 1995), as shown in Fig. C.2. A particular case of narrow-band algorithms are fast marching methods (Sethian, 1999; McInerney and Terzopoulos, 1996), but they require that the direction of propagation does not change. These methods are often used for fast reinitialization (Peng et al., 1999). The time step limiting CFL condition can be circumvented with well known implicit numerical methods, like the additive operator splitting scheme (Lu et al., 1991; Goldenberg et al., 2001; Weickert et al., 1998), at the cost of increased computational complexity per iteration.

However, the principle for some methods is to allow for the formation of new interfaces away from the narrow bands of current ones (Chan and Vese, 2001; Li et al., 2007). In this situation the level-set is updated over the entire grid. Re-initialization can be avoided if the level-set is constrained to remain a distance function (Li et al., 2005b). For this purpose, a least-squares

term of the form $\int_R \frac{1}{2}(|\nabla\phi| - 1)^2$ is added to the energy of the level set. This technique also needs large computation times.

A way to design fast algorithms is to use the phase-field method for approximating motion by mean curvature. In this model, the level-set function takes value one inside the curve, zero outside and has a smooth transition from zero to one in a narrow band around the zero level set (Rochery et al., 2005). The corresponding shape energy has the form of a double-well potential. The level-set function can be initialized with the value that yields maximum energy. Already in the first step, the value of the level-set function at each image pixel is pushed toward zero or one by the information in image data. This is not only meaningful, but makes the method initialization invariant. The implementation may be fast when using a two-step algorithm based on threshold dynamics (Esedoglu et al., 2005; Esedoglu and Tsai, 2006). In each iteration, the level-set function is updated according to a phase field flow equation. Subsequently, the function is thresholded such that all pixels get assigned either to the interior or the exterior. Another method that is initialization invariant and does not require re-initialization is proposed in (Xie and Mirmehdi, 2004), and it is based on radial basis function interpolation of the level-set function.

Alternatively, if the level-set function is updated in a narrow-band around the contour, fast algorithms can be designed by keeping lists of grid points equidistant to the zero level-set, and by computing only an approximate distance transform Whitaker (1998). This idea is the basis for probably the fastest level-set implementation available at present, which will conclude this chapter.

In this section, we have described the main choices of a programmer that has the task of implementing the evolution of an active contour. There exists a considerable amount of literature in which solutions are proposed for a particular class of problems solved with level-set methods. We mention a situation where one would like to keep the advantages, but fix an issue that is disadvantageous for the problem at hand. The topology should be preserved when the shape of the object is known, for example when segmenting an opened human hand or a cortical surface in the brain (Han et al., 2003a,b). The authors propose to check at each grid point whether the topology will be changed or not, before allowing the level-set to be updated at that point.

3.6.4. The Fast Level-Set Method by Shi and Karl

The flows proposed in the following chapters of this dissertation are implemented with the fast level-set method (Shi and Karl, 2005a,b, 2008), for two main reasons. First, this method is very fast: it was proven to be two orders of magnitude faster than some numerical schemes, and, it is suitable for real-time tracking of objects based on the pdf of their color feature. In our research, we found no other level-set methods applicable for real-time tracking. Second, segmentations differ only in detail from those obtained with a numerical method; for example, the fast level-set method does not have sub-pixel accuracy.

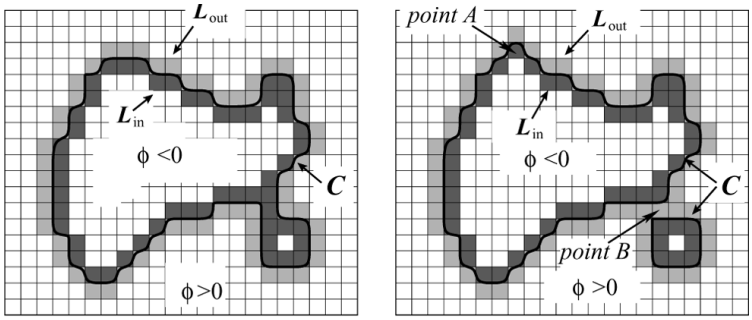


Figure 3.14.: Left: Double representation of a curve (Shi and Karl, 2008), as 1) an integer-valued level-set function ϕ and 2) two lists of pixels neighboring the curve, called L_{in} (illustrated in dark gray) and L_{out} (illustrated in light gray). Right: The curve is propagated by switching points between L_{in} and L_{out}.

The method does not implement the partial differential motion equation numerically. It uses instead a double curve representation: 1) the curve is the level-set of an *integer* valued function, and 2) the curve lies in a very narrow-band, between inside and outside pixels. The first idea, of indicator level-set functions with values in a small integer set, had been inspired by (Gibou and Fedkiw, 2005). In that work, the level-set function takes values -1 for pixels outside and 1 for pixels inside the curve, and it is initialized on an image pre-segmented with a K-means algorithm. The second idea, of memorizing pixels equidistant to the zero level-set in lists had been previously proposed in (Whitaker, 1998). Under the action of image and smoothness forces, the front is evolved by switching pixels between lists and correspondingly updating the level-sets. In the following, we present the details of the algorithm closely following the original papers.

For segmentation purposes, the sign of the level-set function is the one that codes for the region a pixel belongs to. When forces act on the level-set function, the outcome of main interest is the sign of the function. To track the moving curve, a narrow band of bandwidth two pixels is memorized in two lists: a list of pixels that neighbor the zero level-set from the inside, named L_{in}, and a list of outside neighboring pixels, named L_{out}, as depicted in Fig. 3.14.

A curve is propagated outward by switching a pixel from L_{in} to L_{out} and while ensuring that L_{out} will not be disrupted, as is the case for point A shown in Fig. 3.14. To propagate the curve outward the switching need to be from L_{out} to L_{in}, as shown for point B. Switching is caused by forces acting on the contour. Again, only the sign of the force is of interest. If the force at a pixel in the outer list is positive, the curve will evolve outward at that pixel. Similarly, if the sign of the force at a pixel in the inner list is negative, the curve will move inward at that pixel. (Of course, one could take the opposite convention.)

For the switching procedure, it is important that the level-set function has information about whether pixels are list pixels or not. The integer valued level-set function is defined to code for four types of pixels, with values of -3 in the interior, -1 at pixels in the interior list Lin, 1 at pixels

in the exterior list Lout, and 3 in the exterior:

$$\phi(\mathbf{x}) = \begin{cases} 3, & \text{if } \mathbf{x} \text{ is an exterior point} \\ 1, & \text{if } \mathbf{x} \in L_{out} \\ -1, & \text{if } \mathbf{x} \in L_{in} \\ 3, & \text{if } \mathbf{x} \text{ is an interior point.} \end{cases} \tag{3.22}$$

A pixel $\mathbf{x}$ is switched from L_{out} to L_{in} in a procedure named *switch_in*, having two steps: 1) delete $\mathbf{x}$ from L_{out} and insert it in L_{in}, and set $\phi(\mathbf{x}) = -1$; 2) update the level-set function - $\forall mathbfy \in N_4(\mathbf{x})$ satisfying $\phi(\mathbf{y}) = 3$, $\mathbf{y}$ is added to L_{out}, and set $\phi(\mathbf{y}) = 1$. $N_4(\mathbf{x})$ denotes the four-connected neighborhood of pixel $\mathbf{x}$.

A pixel $\mathbf{x}$ is switched from L_{in} to L_{out} in a procedure named *switch_out*, having two steps: 1) delete $\mathbf{x}$ from L_{in} and insert it in L_{out}, and set $\phi(\mathbf{x}) = 1$; 2) update the level-set function - $\forall mathbfy \in N_4(\mathbf{x})$ satisfying $\phi(\mathbf{y}) = -3$, $\mathbf{y}$ is added to L_{in}, and set $\phi(\mathbf{y}) = -1$. Once the switching procedures have been completed, it may happen that a pixel in L_{in} has only negative neighbors in its four-connected neighborhood in the level-set function. However, these cannot be boundary pixels, since they don't have neighbors of the other sign, and must be thus removed from L_{in}. Similarly, pixels in L_{out} that only have positive neighbors must be deleted from L_{out}.

Propagating a curve under the evolution of forces F is summarized as follows

Propagate-curve procedure

- Initialize arrays ϕ, F, and the lists L_{out} to L_{in}.
- Compute forces F at all pixels in L_{out} and L_{in}.
- Outward evolution: for each $\mathbf{x} \in L_{out}$, if $F(\mathbf{x}) > 0$ $switch\_in(\mathbf{x})$.
- Delete redundant points from L_{in}: for each $\mathbf{x} \in L_{in}$, if $\forall y \in N_4(\mathbf{x})$ $\phi(\mathbf{y}) < 0$, delete $\mathbf{x}$ from L_{in}, and set $\phi(\mathbf{x}) = -3$.
- Inward evolution: for each $\mathbf{x} \in L_{in}$, if $F(\mathbf{x}) > 0$ $switch\_out(\mathbf{x})$.
- Delete redundant points from L_{out}: for each $\mathbf{x} \in L_{out}$, if $\forall y \in N_4(\mathbf{x})$ $\phi(\mathbf{y}) > 0$, delete $\mathbf{x}$ from L_{out}, and set $\phi(\mathbf{x}) = 3$.
- If the stopping condition is not satisfied, start new iteration.

The algorithm stops after a fixed number of iterations, or when the forces are positive for all pixels in L_{in}, and negative for all pixels in L_{out}.

The algorithm poses no restrictions on the method for generating forces from image data. The

smoothness force is implemented with a simple Gaussian filter. Curvature calculations on the coarse level-set function in the two pixel narrow band badly approximate the smoothing forces. The solution to this problem is based on the idea that diffusion generated mean curvature motion can be implemented using convolution operations (Merriman et al., 1994). Linear diffusion on an image is equivalent with smoothing the image with a Gaussian kernel. In the previous subsection, it has been shown that the curvature of an implicitly represented curve is the divergence of the gradient of the level-set function, i.e. it is the linear diffusion term in the diffusion equation [2].

One can deduce that in order to accomplish motion by mean curvature, the level-set function can be smoothed with a Gaussian kernel, and consequently thresholded to preserve its integer-valued definition. At a pixel $\mathbf{x}$ in L_{out}, the smoothing force F_s is 1 if $G * \phi(\mathbf{x})) < 0$, and 0 otherwise. G denotes the Gaussian kernel, and $*$ denotes convolution. The definition expresses the intuition that if most pixels in a mask centered on $\mathbf{x}$ are inside the curve, yielding the negative convolution result, pixel $\mathbf{x}$ itself represents a "dip" in the contour, and the contour is thus not smooth. A "bump" is smoothed by evolving the curve inward. By analogy, at a pixel $\mathbf{x}$ in L_{in}, the smoothing force F_s is 1 if $G * \phi(\mathbf{x})) > 0$, and 0 otherwise.

Similar to (Gibou and Fedkiw, 2005), the authors of the fast level-set method propose to run the algorithm in two cycles, to economize on computations for the smoothing term. In the first cycle, the curve is evolved according to the image data forces for a number of iterations N_a, as described in the propagate-curve procedure. In the second cycle, the curve is smoothed for a number of iterations N_s with the same propagate-curve procedure, but with forces computed from Gaussian convolution as previously described. The amount of smoothness is controlled by filter size and the number of filtering operations.

Multiple curves are evolved by modifying the basic algorithm. Given M curves, all are represented with the same level-set function, but each curve administers its own set of lists, L_{in}^i and L_{out}^i. Additionally, a matrix of region labels ψ indexes the region each pixel belongs to. The *switch_out* procedure is as previously, except that the region indicator $\psi(\mathbf{x})$ is also set to zero. The *switch_in* procedure responsible for outward evolution is modified to handle region competition. Two curves are prevented from assimilating the same pixel by means of topological numbers. The topological number of a pixel in respect to a region indicates the number of connected-components in the eight-neighborhood of a pixel intersected with the region. If there are more regions in the eight-neighborhood of a pixel, and the topological number in respect to each is larger than 1, switching the pixel would cause different regions to merge. In this situation no switching is performed. In all other situations, the *switch_out* procedure remains as previously, except that the region indicator is set to the region corresponding to the contour, $\psi(\mathbf{x}) = m$. After a few evolution steps, the boundary between contours is listed in only one of the two L_{out} lists involved, and representation symmetry is broken.

In summary, the properties of the algorithm that make it fast and simple are as follows. All

[2] Linear diffusion involves computing the Laplacian of the level-set function. The curvature, which is equal to the divergence of the normalized gradient is the Laplacian of the function.

computations for level-set propagation are reduced to integer arithmetics, which is notoriously faster than float arithmetics. One only need compute the sign of the image data force, since the contour can move exactly one pixel at a contour point in each iteration. As a result, force computation may also be reduced to integer operations. Forces are computed only at pixels on the inner and outer borders. And finally, curvature computation may be replaced with a fast smoothing with an integer Gaussian kernel.

This elegant algorithm has been slightly modified in the course of this work. The changes will be presented in the following chapters, in sections that disclose implementation details on the proposed methods, according to the chronology of changes. This algorithm and our methods were implemented in C++ in the course of this work.

The closing note of this chapter is that, the overview given herein allows us to place the methods of this dissertation in the AC framework. They are ACs based on novel local region descriptors and a new type of equation of motion from spatially-variant definitions of energy terms.

4. Local Region Descriptors for Active Contours

It is important to prove, but it is more important to improve.

popular wisdom

The overview of active contours in the previous chapter showed that image segmentation methods currently rely on edge information, region information or both. While edges characterize small neighborhoods of pixels, region descriptors characterize entire image regions. This chapter proposes to characterize local image regions by defining Local Region Descriptors (LRDs); these are essentially statistics of features located within windows centered on the evolving contour. Active contours driven by only LRDs are a very recent development in active contours. We use LRDs to define general-form energies based on level sets in a novel manner, such that the number of energy local minima is reduced. For this purpose, a particular energy is associated with an active contour by specifying the function in the energy functional that takes LRD values as argument. For example, this function can be the logarithm of the probability density of features conditioned on the region. We introduce two other such functions based on the assumption that local densities of features are approximately Gaussian. The first is based on a similarity measure between features of pixels that involves confidence intervals and the second is based on a local Markov Random Field (MRF) model. By minimizing the associated energies, the proposed active contours can segment objects that have largely overlapping global probability densities. In experiments, this method can accurately segment natural large images in very short time when using the fast level-set implementation.

4.1. Rationale for Active Contours Based on Region Information

Historically, region intensity information has been added to edge-based energy functionals in order to obtain more robust contours (Paragios and Deriche, 1999b; Cohen et al., 1993). Zhu and Yuille (1996) assumed it to be sufficient to consider only the region information and designed a general energy functional for segmenting an image in N regions. Information about a region

is usually captured by selecting features that quantify its visual characteristics and statistically modeling their variation. The construct obtained by combining features and their statistics is named region descriptor.

Features for region descriptors may refer to intensity or color, texture (Paragios and Deriche, 2002b), motion (Brox et al., 2003a) and optical flow field (Roy et al., 2006). Image features that describe a region usually vary within this region, and the goal of a region descriptor is to formalize and measure the variation. This is commonly achieved by interpreting the varying values of a feature as realizations of a random variable with a probability density function (pdf) that needs to be determined. Some methods take a supervised approach to learning the pdfs. in (Paragios and Deriche, 2002a, 1999b; Ecabert and Thiran, 2002) image intensities are modeled as a Gaussian mixture and its parameters are learned beforehand with an expectation maximization (EM) algorithm; in (Chakraborty et al., 1996) the user selects image samples thus making it possible to compute a mean and a variance for each region. Other methods (Chan and Vese, 2001; Chesnaud et al., 1999; Zhu and Yuille, 1996) take an unsupervised approach, and approximate the parameters of a Gaussian pdf for each region, and at each evolution step.

Real world objects may be not uniformly illuminated, have large stripes, spots and fur, or wear glasses or beards, and their features may thus have arbitrary densities. Pdfs are then better approximated by nonparametric kernel density estimators (KDEs) (Brox and Weickert, 2004a; Kim et al., 2005). With an appropriate choice of kernel window width, nonparametric KDEs can describe the data closely, but because of this, new data points not present in the learning set may have low probabilities. Pdfs of two regions described non-parametrically may be compared by computing a statistical divergence, like the Kullback-Leibler divergence or the Battacharya distance (Freedman and Zhang, 2004).

Generally, a region descriptor is estimated from all samples within the region delimited by the active contour (Paragios and Deriche, 2002a; Chan and Vese, 2001; Brox et al., 2003a), which means that one obtains a global region descriptor in this manner. Active contours based on global region descriptors are negatively affected when the support of distributions of different regions overlap. In this situation, values which fall on the tail of the distribution might be classified based on the maximum likelihood criterion into the wrong region. This is an omnipresent problem in classification tasks; however, an overview of possible solutions is beyond the scope of this paper.

Chances are that the feature varies less when its variation is analyzed in a local region. This means that the overlap between pdfs is likely to decrease if only observations from around the contour are the basis for density estimation. For this purpose, we introduce the concept of local region descriptors (LRDs) computed from samples within windows centered on contour pixels, along with a formalism for constructing energy functionals from LRDs. These functionals have many local minima. This issue is addressed by changing the function that takes LRD quantities as argument. First, a balloon force is added to a region competition equation defined via the function "logarithm-of-pixel-probability" with pdf determined from a LRD. Secondly, we introduce two novel functions based on a Gaussian similarity measure and on local modeling of the image as a

Markov Random Field (MRF). We name all these functions segmenter functions. The proposed active contours are tested on various synthetic and real, gray and color images by placing initial contours inside the objects to be segmented and letting them grow toward object boundaries. In experiments, objects with visual characteristics that create the impression of an approximately uniform appearance can be segmented correctly and rapidly; these objects may be neighbored by other objects that are, at least partly, similar in appearance.

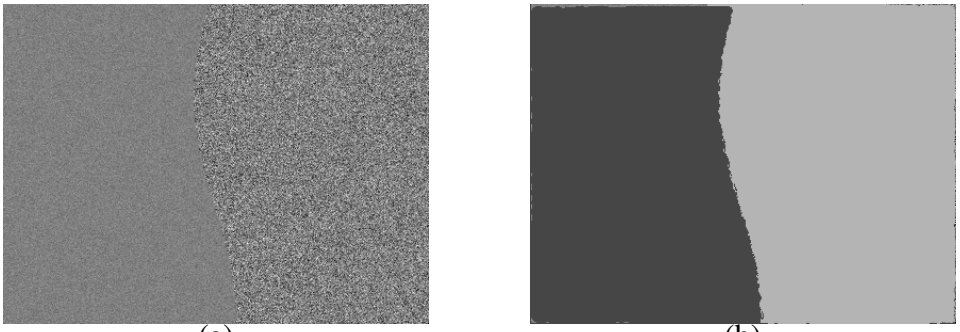
4.2. Local Region Descriptors for Active Contours

Strongly overlapping probability densities usually lead to poor segmentation results. The extreme example of an image with two regions having normal distributions with the same mean but different variances is shown in Fig. 4.1(a), and has been discussed in (Zhu and Yuille, 1996). The authors correct the evolution equation (3.8) by analyzing two sets of parameters in order to solve the problem. One set of parameters is computed for each region from all pixels within this region. The other set is computed from the pixels within a window $W(x,y)$ centered on each pixel on the evolving curve. The probability of a boundary pixel is then replaced with the probability of the window considered to have m independent pixels as shown in equation(3.11). The motion force then includes a term that compares the mean and variance over samples in the window with the global region mean and variance in a statistic-test-like manner.

Global region descriptors decouple pixel intensities from their spatial positions. Visually, more elements form a group not only if they look alike, but also if they are in close proximity of each other, whereas similar elements which are further away will not belong to the group. For a very large number of objects, their visual properties change in a relatively slow manner; a comparatively sudden change in color and texture is very often accounted for by the presence of a boundary between objects. We assume for our method that this situation exists in the image to be segmented.

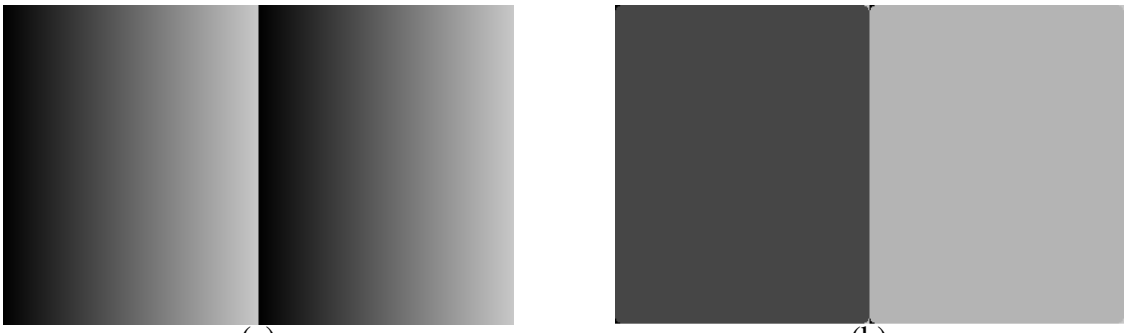
Global descriptors may hamper an active contour. In the example in Fig. 4.1(a), regions are well characterized by their respective variances, but this need not be the case for all objects with overlapping distributions. Consider the synthetic image in Fig. 4.2(a). It shows two rectangles filled with the same color gradient that clearly are two separate regions. The intensities in the two rectangles not only overlap, they even follow the same discrete uniform distribution. However, an observer perceives two rectangles because the intensity varies slowly within each rectangle, but suddenly at the visual boundary.

Describing each region only locally seems more appropriate for this image. In the following we show how both synthetic and natural images can be correctly segmented by minimizing energies given by different segmenter functions that depend on local information only. First, LRDs are defined, then the concept of segmenter functions is introduced along with different segmenter functions.



(a) (b)

Figure 4.1.: Exemplary segmentation of a synthetic image. (a) Synthetic image showing two regions with mean 128 and variances 10 and 35 and (b) its segmentation by (4.3) (using the segmenter function g_{mrf} of LRDs). The segmentation is initialization invariant, as long as an initial contour is placed inside each region.



(a) (b)

Figure 4.2.: Exemplary segmentation of a synthetic image. (a) Synthetic image showing two regions with the same discrete uniform distributions and (b) segmentation obtained with LRDs plus balloon force according to (4.3). The segmentation is initialization invariant, as long as an initial contours is placed inside each region.

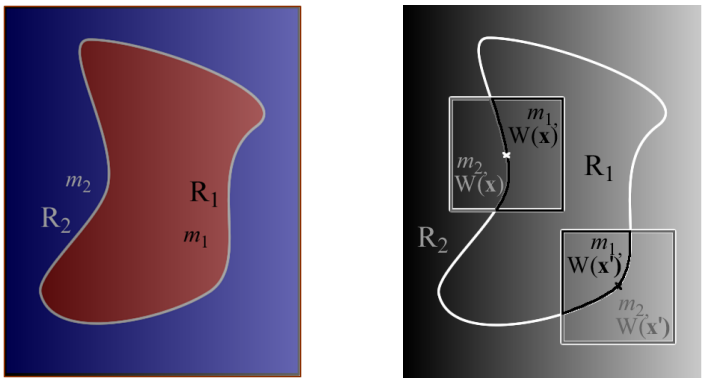
4.2.1. Local Region Descriptors

A region descriptor is defined by choosing or computing features that quantify the visual characteristics of a region, and by choosing a probability model to express the variation of those features within the region. Features may be chosen from intensity, color or quantities that measure texture properties, e.g. values obtained with Gabor filters. Probability distributions on features may be modeled parametrically or non-parametrically.

Local region descriptors are region descriptors with probability distributions computed from feature samples *within regions that lie inside windows centered on an active contour point*. At each pixel on the contour, statistics that describe a region are computed only from samples in this region. The LRD is defined by specifying the size of the window, the features that describe a region and the type of pdf underlying the feature samples in the window. Features may be

computed by either taking into account the position of the contour or not. Choosing the window's shape is also part of an LRD's definition. Most obvious choices for the shape of a window with the purpose of describing the image locally are squares (Pappas, 1992; Zhu and Yuille, 1996) or circles (Zhu and Yuille, 1996; Lankton et al., 2007).

Two square-shaped windows used for computing LRD values are depicted in Fig. 4.3. An initial curve divides the image into an inner and an outer contour region, denoted in the image with R_1 and R_2. Each window includes patches from both regions. We observe that a LRD value describing R_1, for example the mean pixel intensity, can vary significantly along the contour; the same is true for an LRD describing R_2. For example, the inside mean m_1 in the window on the left of the image is closer in value to the outside mean m_2 in the same window than to inside mean m_1 in the window on the right.



(a) (b)

Figure 4.3.: Contour separating the image in an inside region R_1 and an outside region R_2. (a) Global region descriptors m_1 and m_2 are computed from all observations in R_1, respectively R_2. (b) Two windows $W(\mathbf{x})$ and $W(\mathbf{x}')$ centered on contour pixels. Region patches included in each window are highlighted. Each includes the pixels for computing the LRDs m_1, m_2 in $W(\mathbf{x})$, and m_1 and m_2 in $W(\mathbf{x}')$.

We were inspired by Pappas' adaptive clustering algorithm (Pappas, 1992) in our decision to determine the statistics of each region separately. That algorithm estimates the local pdf of a class of pixels only from pixels that already belong to this class with very promising results.

In (Zhu and Yuille, 1996), samples within a window are assumed to follow a Gaussian distribution, but in the authors's algorithm the position of the contour is not taken into account when computing statistics. Because statistics are computed without taking the boundary position into account, the detected boundary may differ from the image boundary by an amount proportional to the local window size. This fact influences the choice of window size: windows too small do not include enough samples to reliably compute statistic forces; windows too large are associated with large uncertainty about real boundary positioning. Unlike in (Zhu and Yuille, 1996), *features pdfs are computed for each region in the local window, a separate pdf is computed only*

from the the observations in this region, as illustrated in Fig. 4.3. Another difference to (Zhu and Yuille, 1996) is that here global descriptors are not computed. With LRDs the boundary is generally accurate. The window size influences the result minimally, as long as it contains enough samples from each region and the assumptions about the local pdfs are true; this can be seen in the results of experiments with different window sizes (Figs. 4.8, 4.12 in Section 4.4.3). These results are due to the fact that, as the contour approaches the boundary, only samples from one image regions are used to compute the statistics for this region. The possibility of separately describing patches of regions is theoretically mentioned in (Ronfard, 1994), but implemented is a method that compares all samples in a small window centered on a snake point with region patches outside and inside the point.

A method that involves local region processing and takes into account the contour's position has been presented very recently in (Piovano et al., 2007); an independent development, the method turns out to be based on a special type of LRD. The feature used to describe the region is the local weighted average of the intensity inside the region. This feature is obtained at every pixel by convolving the image with a Gaussian kernel that is strictly positive only inside the region containing the pixel. The window W is reduced to a pixel and there is no probability distribution model involved.

Another interesting result is that the piecewise smooth Mumford-Shah energy is equivalent to a likelihood-based energy (3.20) based on local region descriptors (Brox and Cremers, 2007). The local region descriptors must be composed of a normally distributed intensity feature with constant $\sigma = \sqrt{0.5}$ in windows of given size W. This is an alternative interpretation of the LRD in (Piovano et al., 2007). Another LRD for which the feature is pixel intensity is used in (Lankton et al., 2007) to formulate a hybrid region-edge-based active contour; the local statistics computed for this feature are the local region sample mean and variance.

For the images segmented in this paper, we choose simple region descriptors that rely on intensity or color and we assume the distributions of these features are locally Gaussian; intensity is modeled with univariate normal distributions and color with multivariate ones. We make this choices because we want to emphasize the advantages that come from local modeling.

4.2.2. Segmenter Functions

The majority of energies associated with active contours have either a form similar to (3.5) or to (3.20); (3.5) expresses the assumption that the image can be approximated with piecewise smooth functions, while for (3.20) it is assumed that each pixel is assigned a region label and the energy represents the logarithm of the joint pdfs of pixel features conditioned on their region label. Both energy types thus include a double integral over the image domain of functions of values depending on pixel features. This suggests that we can introduce general-form energies based on LRDs by allowing a flexible form for the function under the double integral.

Formally, consider a LRD composed of a feature vector $f(x,y)$ and a pdf model with a different set of parameters $\theta^{W(x,y)\cap R_i}$, shortly θ_i^W, in each window $W(x,y)$ centered on a contour pixel (x,y). Also consider a real-valued function g denoted as *segmenter function*. Each region R_i is represented by a level-set function Φ_i. We propose the minimization of the following energy functional, considering the number N of regions present in the image known:

$$E(\Phi_i,\theta_i^W,N)=\sum_{i=1}^{N}\left(-\int\int_R\Big(H(\Phi_i)g(f_i(x,y),\theta_i^W)-\frac{\mu}{2}|\nabla H(\Phi_i)|\Big)dxdy\right) \tag{4.1}$$

The equation of motion for the level set function Φ_i can be derived from the associated Euler-Lagrange equation (as shown in the Appendix A):

$$\frac{\partial\Phi_i}{\partial t}=H'(\Phi_i)\left(g(f_i,\theta_i^W)-g(f_j,\theta_j^W)+\frac{\mu}{2}k_i\right), \tag{4.2}$$

where we denote the curvature term by $k_i=\left(div\left(\frac{\nabla\Phi_i}{|\nabla\Phi_i|}\right)\right)$.

To obtain an energy similar to the energy in (3.20) we have to set the segmenter function to $g=\log p$. However, the parameters of p are not computed from all values in a region R_i, but as LRDs from local observations.

Very often, the result of applying a filter to an image region is considered to be normally distributed, and thus g is the logarithm of a Gaussian pdf of a random vector (with more components when more filters are employed), i.e. $g=\log p_{\mathcal{N}}$ with $p_{\mathcal{N}}\sim\mathcal{N}(\mathbf{m}_i^W,\Sigma_i^W)$ and where $\mathbf{m}_i^W$ is the mean and Σ_i^W is the covariance matrix. The image of a spiral (Chan and Vese, 1999) is initialized with a grid of circles. It is segmented by a contour moving according to (4.2) with the segmenter function $g=\log p_{\mathcal{N}}$. The LRD is pixel intensity assumed to be a feature which locally follows a normal distribution. The result can be seen in Fig. 4.4.

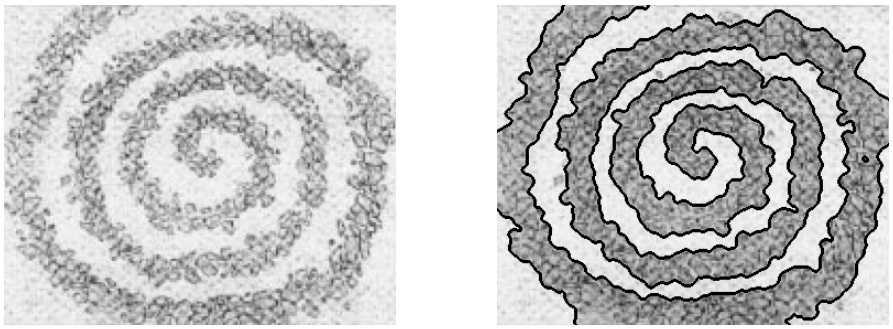


Figure 4.4.: A spiral image presented in (Chan and Vese, 1999) (left) segmented with LRDs and segmenter function $g=\log p_{\mathcal{N}}$ (right).

Contours are best initialized within real objects with the proposed method, similar to (Zhu and Yuille, 1996). If this requirement is not met, and a real boundary cuts through an initial patch, it

still may be possible to obtain a correct segmentation by evolving the contour according to the region competition equation (4.2). For this purpose it is necessary that part of the object to be segmented occupies a larger area in the initial patches. This situation occurs for the spiral in Fig. 4.4.

Energies based on LRDs are prone to have more local minima than those based on global descriptors (depending on the degree of separability between descriptors), especially when an initial contour lies within a real object. Since LRDs characterize the image locally, they are "short sighted". They tend to be equal on both sides of a contour making the motion force approximately zero. Not being able to use information from image parts outside the windows, the contour cannot escape a local minimum.

One alternative is to rely on the curvature term to evolve the contour when it lies in a homogeneous region, as in (Piovano et al., 2007). This alternative requires balancing the influence of the data term and of the curvature term in the energy functional (3.20): the less homogeneous the regions, the larger μ needs to be; however, this is known to be an unreliable solution. Because of the large number of local minima, the method in (Piovano et al., 2007) is very sensitive to initialization and so is the method in (Lankton et al., 2007). For both methods, local image data must have unimodal distributions that are well characterized by the local mean. The variance of the data is not taken into consideration, but it can often be an important source of information (e.g. in the image in Fig. 4.1). Such probability models are often not suitable for descriptors based on more than one feature.

In the previous section we have noted that special cases of LRDs are used to construct the energies in (Piovano et al., 2007; Lankton et al., 2007); we may now show the segmenter functions used there. We have discussed in the previous section that the LRD in (Piovano et al., 2007) can be interpreted in two different ways. If the interpretation is that the LRD relies on the feature intensity, the segmenter function that needs to be used is $g = \log p_{\mathcal{N}}$ with $p_{\mathcal{N}} \sim \mathcal{N}(m_i^W, \sqrt{0.5})$ (Brox and Cremers, 2007). Note that the Mumford-Shah functional in (3.5) cannot be expressed, in general, as a particular case of proposed energy (4.1), since the function smoothness term $\int_{R-C} |\nabla u| d\mathbf{x}$ is missing.

If the LRD is considered to be based on the local weighted mean intensity inside a region, denoted here by $f(\mathbf{x})$, the energy proposed by (Piovano et al., 2007) is obtained with the segmenter function $g = (I(\mathbf{x}) - f(\mathbf{x}))^2$; $I(\mathbf{x})$ denotes image intensity at pixel $\mathbf{x}$. In this case, g resembles the term under the integral in the Mumford-Shah functional from (3.5). The segmenter function in (Lankton et al., 2007) also resembles this term, since $g = (I(\mathbf{x}) - m_i^W(\mathbf{x}))^2$, where $m_i^W(\mathbf{x})$ is computed as the local mean intensity inside region R_i. We note here that the function in (Ronfard, 1994) may also be seen as segmenter function. It is based on the Ward distance and its role is to decide whether a small rectangular patch around a snake point should be merged with the foreground or the background.

Another known solution to the local minima problem involves adding a small constant force term

λ to the motion of the level set Φ_i:

$$\frac{\partial \Phi_i}{\partial t} = H'(\Phi_i)\left(g(f_i, \theta_i^W) - g(f_j, \theta_j^W) + \lambda + \frac{\mu}{2}k_i\right). \tag{4.3}$$

The parameter λ can be seen as a maximal area constraint (Chan and Vese, 2001) or plays the role of a balloon force (Cohen, 1991). In the context of LRD-based contour evolution, it can be interpreted as follows: λ indirectly gives a measure of the minimal difference that needs to exist between two local image regions in order to consider that they describe the sides of a real image boundary. As such, it may reflect the a priori knowledge about the smallest local image variations created by real edges. Equation (4.3) describes a *local* region competition equation with an added balloon force that yields a method for segmenting images with strongly overlapping distributions for foreground and background objects, as will be discussed in Section 5.5 and exemplified in Fig. 4.5.

For the moment, we turn to the synthetic example in Figure 4.2(a): the segmentation in Figure 4.2(b) is obtained according to (4.3). The feature for the LRD is pixel intensity, which is assumed to be normally distributed in local regions, and the segmenter function $g = \log p_{\mathcal{N}}$; this particular form of (4.3) will be referred to as *LRDs plus balloon force*. A circular initial contour is placed inside each region. While the means and variances computed for the outside and inside of the contour in each window are minimally different, balloon force λ drives each contour to grow. When the contour reaches a bright-to-dark boundary, the local regions are different enough for the contour to stop. We thus observe that at the boundary, distributions of LRDs do not overlap. Since intensity in these regions varies linearly, the initial contours may be placed anywhere inside the respective regions. However, if global descriptors are computed, the result of the segmentation will depend on the initialization. For example, if the initial circles are centered within each region and include no dark pixels, the dark pixels will remain in the background since their initial probability is larger in the background, a fact that does not change when region histograms are reestimated at each iteration.

4.2.3. Novel Segmenter Functions

The search for other solutions that avoid local minima led us to develop two novel segmenter functions as will be shown in the following.

Consider an initial contour included in an image object. Intuitively, it should grow as long as pixels just outside the contour are similar to pixels just inside the contour; then, instead of describing both sides of the contour for region competition, only the local regions inside the contour need to be described. A similarity test then checks whether outside pixels match the description. This means that only the object to be segmented needs to be described, while its background need not be described.

In order to describe this intuition formally, we assume that the LRD has a normal distribution and we define the segmenter function g to be the confidence that a samples was drawn from this distribution. For a one-dimensional feature f_1 we know the percentage α of values that fall within an interval of length c_α around the mean m^W, measured in standard deviations s^W. Chosen a confidence level α, the function g will be written:

$$g_{sim}(c_\alpha, f_1(\mathbf{x}), \theta^W) = c_\alpha \cdot s^W - |f_1(\mathbf{x}) - m^W|, \tag{4.4}$$

where $|.|$ denotes the absolute value; we remind here that $\theta^W = (m^W, s^W)$ and $W = W(x,y) \cap R_{in}$. The smaller the difference between the feature value and the window mean the higher is the confidence that the pixel at the center of the window belongs to the inside of the contour. For $\alpha = 0.68$, the values accepted as similar fall within one standard deviation from the mean ($c_\alpha = 1$). This means that if the difference is larger than one standard deviation, the pixel is considered to belong to the outside

The confidence for observing given values of a multivariate normal distribution can be expressed using the value of the χ^2_p distribution, with p degrees of freedom (Duda et al., 2001) p.629. Thus, if more features are observed, g is formulated using χ^2_3. For a chosen confidence $1-\alpha$, values similar to the curve's inside will fall within a given interval of size $\chi^2_p(\alpha)$ around the mean. Let $f(\mathbf{x})$ be the feature vector, $\mathbf{m}^W$ its mean, and Σ its covariance matrix. g_{sim} is then written as

$$g_{sim}(\alpha, f(\mathbf{x}), \theta^W) = \chi^2_p(\alpha) \; - \; (f(\mathbf{x}) - \mathbf{m}^W)' \, \Sigma^{-1} (f(\mathbf{x}) - \mathbf{m}^W). \tag{4.5}$$

and it can be interpreted similarly to the one-dimensional case. Only LRDs which describe the inside of the object are computed, and letting all the variables describe only one object, the energy to be minimized is modified to refer to only one object:

$$E(\Phi, p) = -\iint_R \Big(H(\Phi) g_{sim}(f(\mathbf{x}), \theta^W) - \frac{\mu}{2} |\nabla H(\Phi)| \Big) d\mathbf{x}. \tag{4.6}$$

The energy is minimal when $g_{sim}(\mathbf{x}) > 0$ for each $\mathbf{x}$ with $H(\Phi(\mathbf{x})) = 1$, i.e. each pixel in the segmented object must be similar to its neighbors in the object, with similarity measured by g_{sim}. The associated Euler-Lagrange equation for the level-set function Φ can be derived (see the Appendix), leading to the following equation:

$$\frac{\partial \Phi}{\partial t} = H'(\Phi)(g_{sim}(f, \theta^W) + \frac{\mu}{2} k). \tag{4.7}$$

This equation can be interpreted as follows: the level set will move to include a contour pixel $\mathbf{x}$ if $g_{sim}(\mathbf{x}) > 0$ and the curve stays smooth; it will do the opposite if $g_{sim} < 0$. Since only one object is described, the background can be complex and have any sort of empirical distribution as long as it does not match the local description of the object to be segmented. It is thus still possible to segment the object with g_{sim} even if object edges are weak (e.g. Figs. 4.7, 4.9). This is not possible with local region competition plus balloon force that needs minimally strong

edges, i.e. edges are stronger than edges inside the object to be segmented. Also, this method has difficulties when one of the objects separated by the boundary has a local probability distribution with multiple modes (see the incorrect segmentation of the camera in Fig. 4.10(c)).

The similarity-based function g_{sim} has difficulties to segment objects that are very noisy (e.g. Figs. 4.1 and 4.15) or when a small percentage of the pixels in a local window does not match the local Gaussian description (e.g. the background of Fig. 4.7 is slightly textured and there are some highlights in the arm and racket in Fig. 4.14).

Methods that deal with this type of problems are often based on Markov Random Fields (MRF) (Pappas, 1992), and we will now show how a more robust segmenter function can be obtained by using MRF modeling.

A MRF models the intuitive idea that the intensity of a pixel depends on the intensities of neighbored pixels. Let the current segmentation of image I be C, i.e. C is the characteristic function of the current local segmentation. If this segmentation is considered to be the realization of a MRF, the pdf for a pixel's segmentation label can be approximated by a Gibbs distribution as follows (Pappas, 1992):

$$P(C(\mathbf{x}) = c) = \frac{1}{Z} exp\left(-\frac{1}{T}\sum_{cl \in Cl(\mathbf{x})}(V_{cl}(c))\right), \tag{4.8}$$

with $Cl(\mathbf{x})$ the set of pairwise cliques that include pixel $\mathbf{x}$, and $V_{cl}(c)$ the potential of a clique cl of two 8-connected pixels $\mathbf{x}$ and $\mathbf{y}$:

$$V_{cl}(c) = \begin{cases} -\beta, & \text{if } c(\mathbf{x}) = c(\mathbf{y}) \text{ and } \mathbf{x}, \mathbf{y} \in Cl(\mathbf{x}) \\ \beta, & \text{if } c(\mathbf{x}) \neq c(\mathbf{y}) \text{ and } \mathbf{x}, \mathbf{y} \in Cl(\mathbf{x}). \end{cases} \tag{4.9}$$

T is a parameter considered here constant. In the modeling process, the original image is assumed to be a noisy version of the segmented image. The goal is to maximize the conditional pdf $p(I(x,y)|C = c)$ assuming that it is normally distributed with mean m_i^W and standard deviation $\mathbf{s}_i^W$. This is equivalent to minimizing the cost

$$Cost(\mathbf{x} \in R_i) = \frac{(I(\mathbf{x}) - m_i^W)^2}{2(s_i^W)^2} + \frac{1}{T}\sum_{cl \in Cl(\mathbf{x})} V_{cl}(c). \tag{4.10}$$

It may be observed that minimizing (4.10) is equivalent to maximizing (4.4) if the potential term is ignored. One can conclude that it is possible to write a regularized version of (4.4) by taking into account the potential term:

$$g_{mrf}(c_\alpha, f_1, \theta^W, C) = c_\alpha \cdot s^W - |f_1(\mathbf{x}) - m^W| - \nu \sum_{cl \in Cl(\mathbf{x})} V_{cl}(c). \tag{4.11}$$

Parameter ν controls the influence of the regularization term. The purpose of this regularization is to ensure a smooth segmentation; from this point of view, it may be seen as a binarized version

of the smoothness constraint in the Mumford-Shah functional in (3.5).

The MRF cost function g_{mrf} can replace g_{sim} in the energy of a growing curve (4.6) and in the corresponding equation of motion (4.7). It has been employed to segment the synthetic example in Fig. 4.1(a). The image was first filtered to replace each pixel's intensity with the value of the standard deviation of pixels in the 8-neighborhood; subsequently a Gaussian filter was applied and the resulting image was segmented employing the segmenter function g_{mrf}. The result is shown in Fig. 4.1(b). An initial circular contour was placed inside each image region and the two contours evolved independent of one another; one can thus observe for each region a stable segmentation result, independent of the other region.

Knowledge about the image can influence not only the choice of the pdf model for computing LRDs, but the choice for the segmenter function g as well. A more detailed discussion follows in Section 5.5, after mentioning one possible implementation for the method presented in this thesis.

4.3. Fast Implementation by Approximations

Local region descriptors for active contours are implemented with the fast level-set method. Like many other computations in this method, calculating g_{mrf} is reduced to comparing the number of cliques assigned a positive β when the pixel belongs to the contour's inside with a threshold. Even with such a strong approximation, segmentation results are very good.

In order to reduce the number of iterations needed for convergence, several initial patches can be placed inside one object, since level-set methods can deal with topology changes automatically (Osher and Sethian, 1988). Finally, LRDs can be computed every few steps during curve evolution, instead of every step; their values then determine motion forces not only for the center of the window, but also for pixels in its immediate neighborhood N_s of range s; when $s = 0$ the computation is carried out only for the contour pixel. The algorithm may be summarized as follows:

- Initialize contour C, level set Φ, and select g and W
- For each pixel $\mathbf{x}$ on C
  - Compute θ_i^W, the LRD for each neighboring region i.
  - For each $\mathbf{y} \in N_s(\mathbf{x})$, compute the sign of $g(\mathbf{y}, \theta_i^W)$.
- Evolve C according to the **Propagate-curve procedure** in Section 3.6.4.
- Stop if convergence or maximum number of iterations reached.

With this algorithm, short times for the segmentation of large images have been obtained, as will be shown in the next section.

4.4. Results and Discussion

In this section, we present results for the proposed method and discuss the effects of different choices for parameters and window sizes. The method was implemented in C++ and all computations have been reduced to integer arithmetic. Contour initialization is started by the user, by selecting a few pixels in each object to be segmented and a circular contour is generated around each selected pixel.

4.4.1. Experiment Settings

In our experiments natural gray-level and color images were segmented. Features for LRDs are pixel intensity or color unless otherwise specified and their local pdfs are assumed to be Gaussian. We have experimented with LRDs, $g = \log p_{\mathcal{N}}$ and balloon forces according to (4.3), and with the two novel segmenter functions g_{sim} and g_{mrf}. The properties of evolution based on LRDs plus balloon forces are discussed together with the influence of the magnitude of the balloon force in Section 4.4.3. For the novel segmenter functions, c_α was set to 1 for all images. Experimentally it was found that a smaller value of 0.75 will often create holes in the segmentation, and a larger value of 1.5 or 2 will cause leakage, especially for regions with large variances. Both g_{sim} and g_{mrf} use only information about the object. Due to this fact, the boundary between background and silhouette in Fig. 4.7 can be segmented although the hair and shirt contain the intensities present in the background because these regions do not fit the local normal distribution for the background close to the boundary. Starting from the silhouette, with an initialization that is just-as-far away from the boundary, we could not achieve this segmentation because there are too many edges present.

With a few exceptions, boundary smoothness was implemented by filtering the level-set function with a Gaussian filter of 5×5 pixels (standard deviation of 1.5), because a larger filter can better prevent leakage; it also prevents the formation of small holes in the segmentation (e.g. of noisy images). However, a larger filter does not allow for an accurate segmentation of corners and for such regions it is better to employ a smaller filter. For example, due to sharp corners in the woman's shirt and the table, a 3×3 pixels filter has been employed to segment images in Fig. 4.15 and Fig. 4.14.

The size of the window W and the magnitude of the balloon force were varied in the experiments; segmentations change gradually as these parameters are varied; in some situations, these changes are minimal. The parameter β introduced to compute g_{mrf} has also been varied. It was found

that it has a much larger influence on the segmentation result, since it is very coarse.

4.4.2. Comparison with The Region-Competition Method by Zhu and Yuille

LRDs plus balloon force (with window side length of 11px and $\lambda = 5$) were used to segment and track the hands shown in Fig. 4.5; the difficulty of these images lies in the overlapping support of the histograms for the hands and the background (intensities between 50 and 75). For illustration purposes, two such regions are marked with ellipses in Fig. 4.5. Edge detectors, like Canny and Sobel (Gonzalez and Woods, 2002a) yield edge maps that lead to leakage or false boundaries. The lightest patches in the hands, as shown in Fig. 4.5, are detected with an adaptive-threshold segmentation of the background-subtracted image. The binary image is repeatedly eroded (Gonzalez and Woods, 2002a) to ensure that the initial contour is well inside the hands; using connected components, only the largest two patches from the eroded image are kept and used to initialize the contour. Subsequent frames are initialized by combining the information from the background-subtracted current frame and optical flow computed for pixels on the final contour in the previous frame. The full tracking application will be presented in detail in chapter 7.

The images of hands in Fig. 4.5 have also been segmented with a fast level set implementation of the method introduced in (Zhu and Yuille, 1996), starting with the same initial contours. Since neighbored regions on the arms and sleeves are not well characterized by their variances, the method (Zhu and Yuille, 1996) cannot achieve an accurate segmentation of the hands, as shown in Fig. 4.5(c). Dark pixels are mostly in the background and such pixels on the hand also get assigned to the background.

With the method in (Zhu and Yuille, 1996), further segmentation problems created by overlapping global distributions can be seen in Fig. 4.6. For the image in Fig. 4.6(e), the initial contour grows to include all dark objects, instead of stopping at the silhouette. All buildings are included in the background of Fig. 4.6(f). One can observe that many contours in segmentations in Fig. 4.6 (g) and (h) are correct; however the segmentation has leaked in Fig. 4.6(g), and it stops before reaching the boundary in Fig. 4.6(h). In this last image, pixels in the face region distort the PDF of part of the background region. Combined with the intensity gradient in the background itself and the noise in the image, this creates the conditions for the contour to stop prematurely. These problems are not present in the segmentations obtained with the method presented in this paper, as can be seen in Fig. 4.6 (i)-(l).

(a) (b) (c)

Figure 4.5.: Segmentation of two images of hands. (a) Initial contours. In the top image two regions with largely overlapping pdfs are marked with ellipses. (b) Segmentation with the method described in (Zhu and Yuille, 1996). (c) Segmentation with LRDS and the constant speed λ.

4.4.3. Influence of Parameter Choices and Initialization

Fig. 4.7 shows the influence of different initializations for the Miss America image. We employ the segmenter function g_{mrf} because the background is slightly textured [1]; the lighter pixels are outliers to a local normal distribution with very small variance. The segmentation in Fig. 4.7(d) has been obtained by setting the initial contour as depicted in Fig. 4.7(a) and using a window side length of size 21×21 pixels, $W = 21px$. Setting the contour as depicted in Fig. 4.7(b),(c), yields segmentations that are within few pixels from the contour presented in Fig. 4.7(d).

For the same image, the influence of the window size is shown by varying the length of the window side from 11px to 31px. The corresponding segmentations are shown in Fig. 4.8. Here, one can observe that larger windows are more prone to forming stable holes. Altogether, we may conclude that initialization and window size have little influence on the final segmentation. Inaccuracies at the image boundaries are partly due to pixels that do not match the interior LRD, e.g., pixels become lighter at the top of the image, and partly due to boundary effects. Critical pixels in boundary regions have less neighbors in their corresponding windows than those in the interior. The latter problem can be solved when the image is extended by repeating border pixels prior to segmentation and, after segmentation, cropping the result to the original size.

[1] Details can be seen by zooming in the electronic version

We now compare segmentations with g_{mrf} and LRDs plus balloon force, still looking at the Miss America image. The background in Fig. 4.7 has a homogeneous texture; the lighter pixels in the texture can be included in the curve due to g_{mrf}. The background, shirt and arm are all dark; these regions have quite similar values, but locally they are slightly different. The differences can be detected with g_{mrf}, as shown in Fig 4.7(d). LRDs plus balloon force in (4.3) do not rely as much on representing the region locally with the correct model, but on what we perceive as strong edges; this approach cannot detect the difference between the two regions, as can be observed in Fig. 4.9(b) (initialization as in Fig. 4.7(a)).

In Fig. 4.10, we show the result for segmenting the Cameraman image with g_{sim} and LRDs plus balloon force, respectively, starting from two different initializations. All segmentations were obtained with $W = 11px$; the balloon force was $\lambda = 5$. Segmentations with g_{sim} are visually the same, while results for the two initializations for LRDs plus balloon force differ very little.

Looking at the results in Fig. 4.10 one may compare the properties of g_{sim} and LRDs plus balloon force. g_{sim} is able to detect the faint edges between the sky and the buildings because the function is sensitive to subtle image changes in regions with small local variances; this also explains the hollow around the camera man. LRDs plus balloon force ($\lambda = 5$) is less sensitive, but more accurate in finding boundaries: the contour around the camera man is very accurate, but part of the camera was segmented into the background and some of the building edges are not detected (Fig. 4.10(c)).

The number of detected building edges depends on the value of the balloon force λ. The influence of this parameter is shown in Fig. 4.11. The window width was $W = 11px$ and λ was varied from 2 to 10 (initialization as in Fig. 4.10(a) top). Fig. 4.11 (a) shows segmentation with $\lambda = 2$; observe that more building edges are found. As λ increases, more of the buildings are segmented into the background. For $\lambda = 8,9$ the segmentation is very similar to the one for $\lambda = 10$ shown in Fig. 4.11(d).

Fig. 4.12 shows the influence of window size on segmenting the background in the cameraman image. The top row shows segmentations with g_{sim} and window side lengths of 13px, 17px and 21px. The source of the differences in segmentation is the color gradient in the sky. For a larger window, the differences between bright outside pixels and the LRD are larger than for a smaller window. The part of the luminous sky center not included in the final contour grows with the window size.

The bottom row in Fig. 4.12 shows segmentations with LRDs plus balloon force with $\lambda = 5$ and window side lengths of 7px, 13px and 17px. In this situation, the larger the window the more inaccurate the segmentation and more of the buildings are assigned to the background, because parts of different buildings are represented with the same region descriptor. This decreases the sensitivity to small changes, like the ones between the sky and the buildings. Since the difference between grass and buildings is large enough, that boundary can still be found.

Fig. 4.13 shows the influence of the threshold β for g_{mrf}. This threshold is set to 4 for most

images: if 4 or more cliques have positive β for the inner region, the speed is positive. Setting this threshold to 3 usually causes leakage, as can be seen for the background in Fig. 4.13(a) and for the racket in Fig. 4.13(c). Setting it to 5 usually prevents the contour from reaching object boundaries, as can be seen looking at the arm shown in Fig 4.13(c). However, there are exceptions. Setting this threshold to 5 may prevent leakage as in Fig. 4.7(d); this is due to the fact that dark pixels in the hair, with intensities similar to those in the background, can be excluded from the segmentation since many have 5 or more neighbors that have intensities that do not match the background. In some situations, it may be useful to set the threshold to 3; highlights on the arm initialized as in Fig. 4.13(d), are better segmented with $\beta = 3$ than 4 (segmentation in Fig. 4.14(b) bottom). When using LRDs plus balloon force in Fig. 4.14, two different initializations result in different segmentations for the arm, but not for other objects. This is due to the lighter crease in the shirt not included in the initial contour. This crease has a smooth shape and forms an edge - pixels on the inside of the final contour are darker, pixels on the outside lighter. This fits the definition of a real boundary and thus the final contour stops at this crease. When the novel segmenter functions are used, the different initializations lead to very similar segmentations.

From our first experiments we conclude that, for LRDs plus balloon force, a window side length of 11px and $\lambda = 5$ are good candidates for starting the segmentation of gray images. Segmentations in Figs. 4.14(c) and 4.15 (c) were obtained with these values. For the novel segmenter functions, a good candidate for the window side length is 21px. Objects in Figs. 4.14(b) and 4.15 (b) were segmented with this window size. An exception was the racket; since this object is small, a smaller window had to be chosen in order to keep the size of the window comparable to the initial patch. The visual characteristics of the background in Fig. 4.14(b) change minimally such that the pdf remains constant over large image patches; the segmentation of the background with $W = 21px$ is thus very similar to the segmentation with $W = 41px$.

A good candidate threshold for g_{mrf} is $\beta = 4$. This threshold has been used to obtain the segmentations for the arm, racket and table in Fig. 4.14(b) top and the racket and table in Fig. 4.14(b) bottom; for the arm in this last image $\beta = 3$. For Fig. 4.15 (b) β was also 4. g_{mrf} was chosen for the arm racket and table in Fig. 4.14 (b) in order to deal with the highlights on these objects; g_{sim} was sufficient in dealing with the background.

4.4.4. Examples of Color Image Segmentation

We have also tested our methods on color images. Figs. 4.16 and 4.17 present the initial contour and the result of the segmentation of frames from two color video sequences, commonly referred to as Akyio and Erik. For these images, we choose the LRD to be a normally distributed RGB color vector. Both segmentations were achieved by LRDs plus balloon force, with a 5×5px smoothing filter, the size of the window W was 11px and $\lambda = 350$. These images cannot be segmented correctly with the function g_{sim} or g_{mrf} because the local regions that include the

eyes and eyebrows have two modes with different basins of attraction (one represents skin, the other eyebrows, eyelashes and iris). This type of distribution is badly approximated by a normal distribution. The eye regions are close to the real object boundaries, and the large variances of the Gaussians fitted to those regions cause background pixels to be segmented into the foreground. The contour thus leaks around the eyes.

The multivariate version of g_{mrf} was used to segment the grass in Fig. 4.18(a); an inferior result obtained with g_{sim} contains more small holes in the segmentation. The image was first smoothed to decrease the variance in the grass. Otherwise, the variance of the window sample is too large and causes the contour to leak into the region occupied by the squirrel. In regions around the squirrel's bottom and tail, grass overlaps fur and there is not a clear smooth boundary. Both local regions include many light pixels. The variance in the local region in the fur is larger than in the grass, and thus, if we start the segmentation from the squirrel, it will leak into the grass at the bottom.

The algorithm's implementation is fast: the 397×499 noisy Claire image and the 409×518 table tennis scene have been segmented in under 5 seconds to obtain each of the two results in Fig. 4.14(b) and (c) and Fig. 4.15(b) and (c), respectively. The faces in the 409×500 and 327×400 color images in Figs. 4.16 and 4.17 were both segmented in under 0.5 seconds. The number of computations for the algorithm depends the number of evolution steps, the number of contour pixels at each step and the square of the window side length. The first two variables make up for the largest amount of computation time. The Cameraman image is segmented in 0.61sec with $W = 11px$ and in 0.71 sec with $W = 21px$, with the initialization from Fig. 4.10(a),top. The image in Fig. 4.15 is segmented in 5 sec with $W = 11px$ and in 7 sec with $W = 21px$, with the initialization from Fig. 4.15(a). However if the background of this image is initialized with 4 instead of 2 circles, the computation time can be reduced to under 1 sec, because the number of evolution steps decreases considerably. The C++ application runs single threaded (it uses only one of the processor cores) on an Intelő Core 2 Duo E6600.

In summary, we have concentrated on region-based active contours and region descriptors. Specifically, we have suggested that the problematic overlap of pdf for different regions can be reduced if pdfs are computed for local regions. We have thus proposed to replace global region descriptors with local ones in the framework of active-contours segmentation. For this purpose, LRDs have been defined by choosing representative region features and selecting a probability model to represent their variation. Parameters of the local pdf are computed from samples that lie in a region within a window centered on the active contour. We have proposed to associate energies based on LRDs to active contours by defining segmenter functions for a general-form energy. This procedure has been exemplified with three segmenter functions. The results are a region competition equation and an object-oriented approach that proved to have different properties in experiments. Segmentations were accurate in situations where the global distributions of foreground and background overlapped, and other methods failed. Results could be obtained very fast by reducing calculations to integer arithmetics and choosing the fast level-set algorithm.

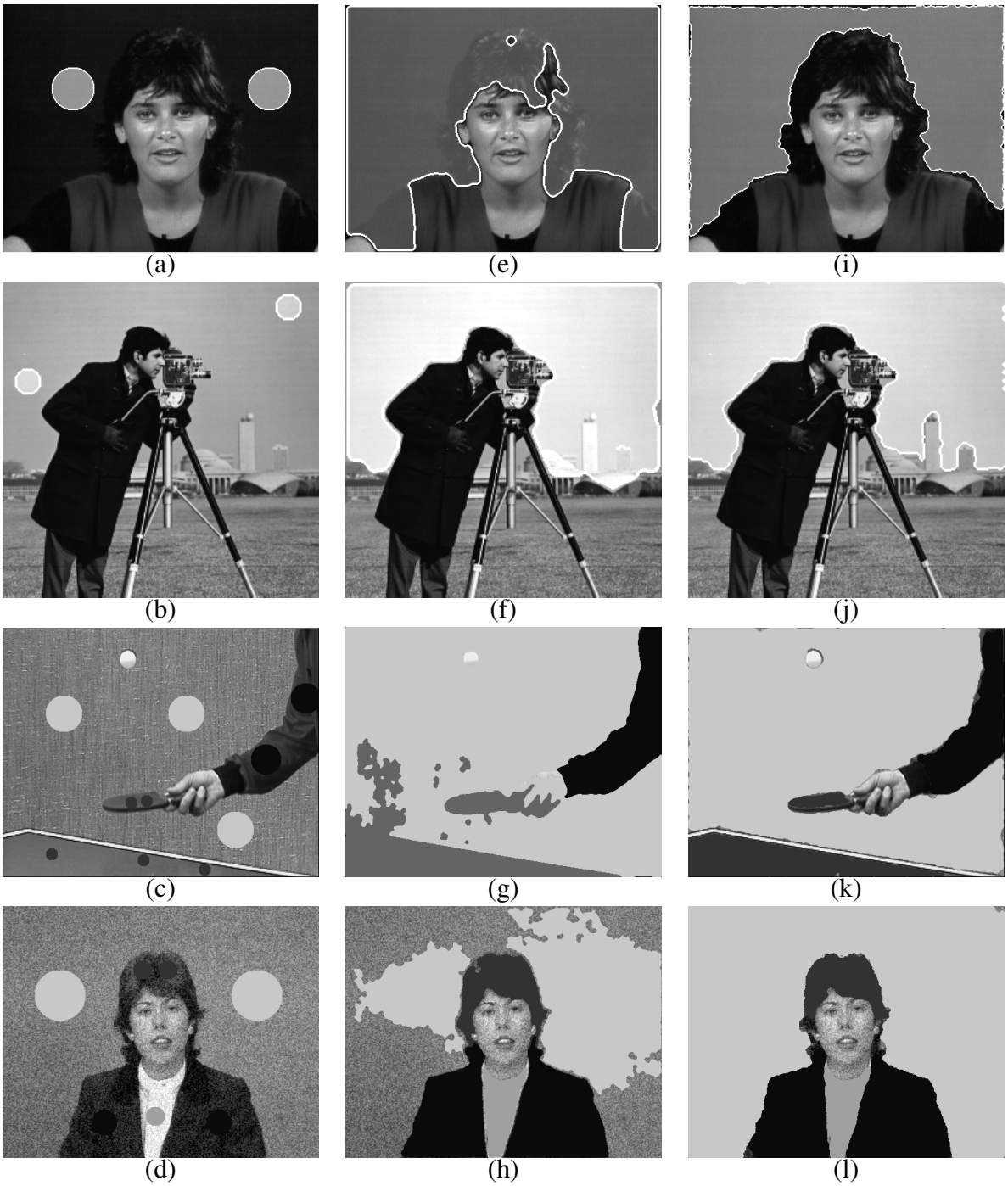


Figure 4.6.: Comparison of two segmentation methods. (a)-(d) Initial contours for both segmentation methods. (e)-(h) Segmentations obtained with the method in (Zhu and Yuille, 1996) (superimposed on initial images). (i)-(l) Segmentations obtained with the proposed methods (superimposed on initial images). Note that the proposed method yields better results for all examples.

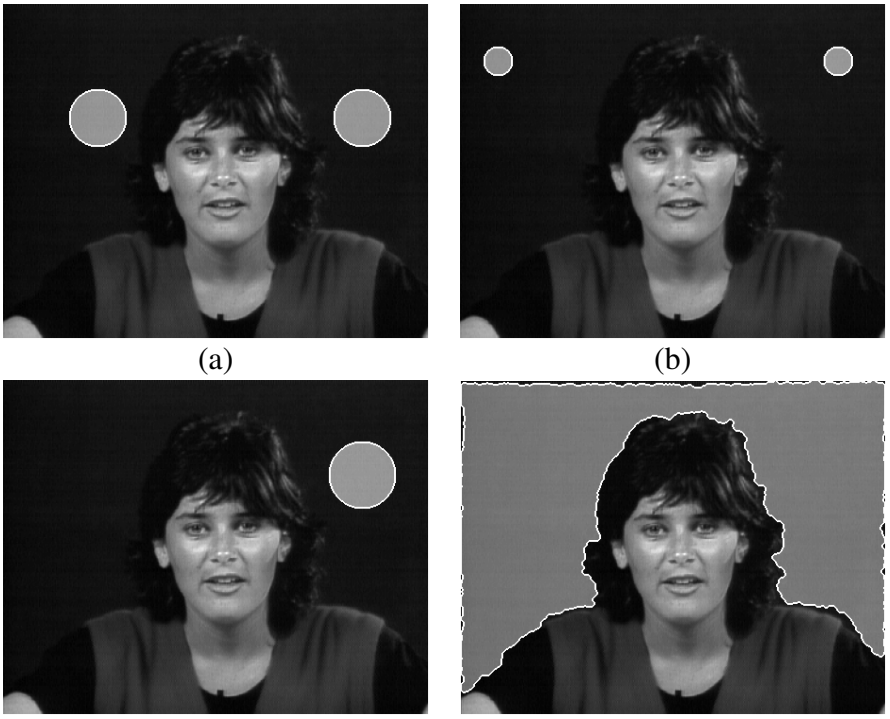


Figure 4.7.: Segmentations obtained with different initializations. (a)-(c): Different positions of initial contours for a popular image. (d) Segmentation with the function g_{mrf} ($\beta = 5$) for initialization (a) and $W = 21px$; the other initializations result in contours within a few pixels difference from the one in (d).

Figure 4.8.: Segmentations obtained with g_{mrf} and $\beta = 5$ and different window side lengths: (a) 11px, (b) 15px, (c) 25px, (d) 31px. Initialization as in Fig. 4.7(a). Note that the different window sizes result in very similar segmentation results, which are within a few pixels from one another.

Figure 4.9.: Intermediary step (a) and segmentation obtained with LRDs plus balloon force, $W = 11px$ and $\lambda = 5$ (b) (initialization as in Fig. 4.7(a)). Note that the LRDs plus balloon force method is not sensitive enough to detect the boundary between the female subject and the background.

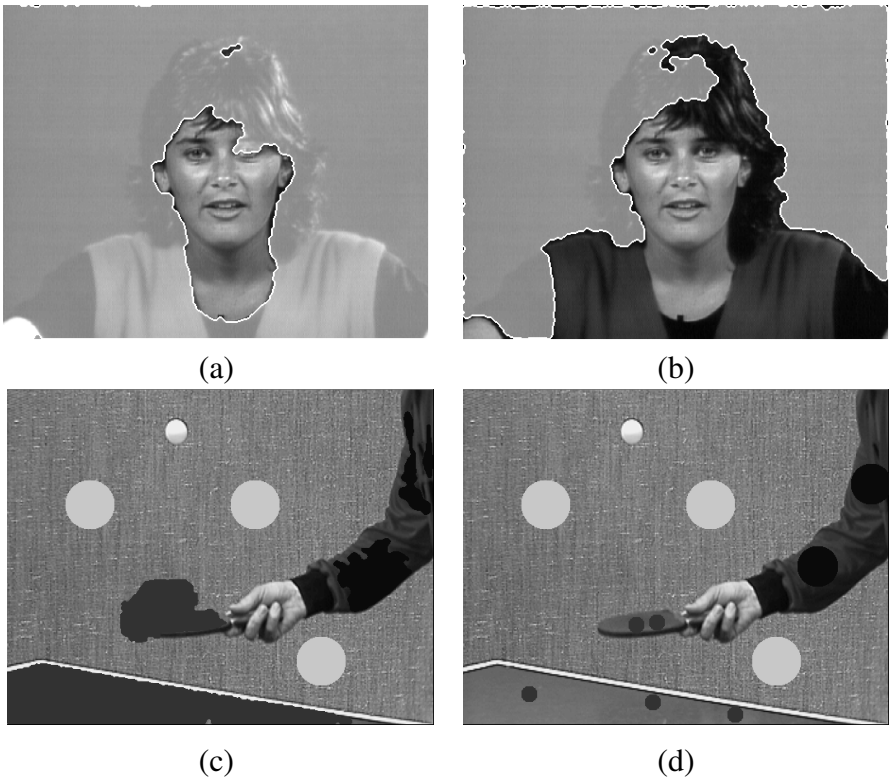
(a) (b) (c)

Figure 4.10.: (a) Two different initializations. (b) Corresponding segmentations obtained with g_{sim} and $W = 11px$. (c) Corresponding segmentations obtained with LRDs plus balloon force and $W = 11px$ and $\lambda = 5$. Note that g_{sim} detects the presence of the smaller building, while LRD plus balloon force does not. g_{sim} is thus more sensitive to changes in locally Gausian regions. Note also that the two different initializations result in very similar segmentations, which are within a few pixels of one another in both situations.

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

Figure 4.11.: Segmentation results for different values of λ: (a) $\lambda = 2$, (b)$\lambda = 6$, (c) $\lambda = 7$, (d)$\lambda = 10$ (with $W = 11px$ and initialization as in Fig. 4.10(a)top.) Note that as λ increases more of the buildings get drawn into the background due to faint edges between the sky and the buildings.

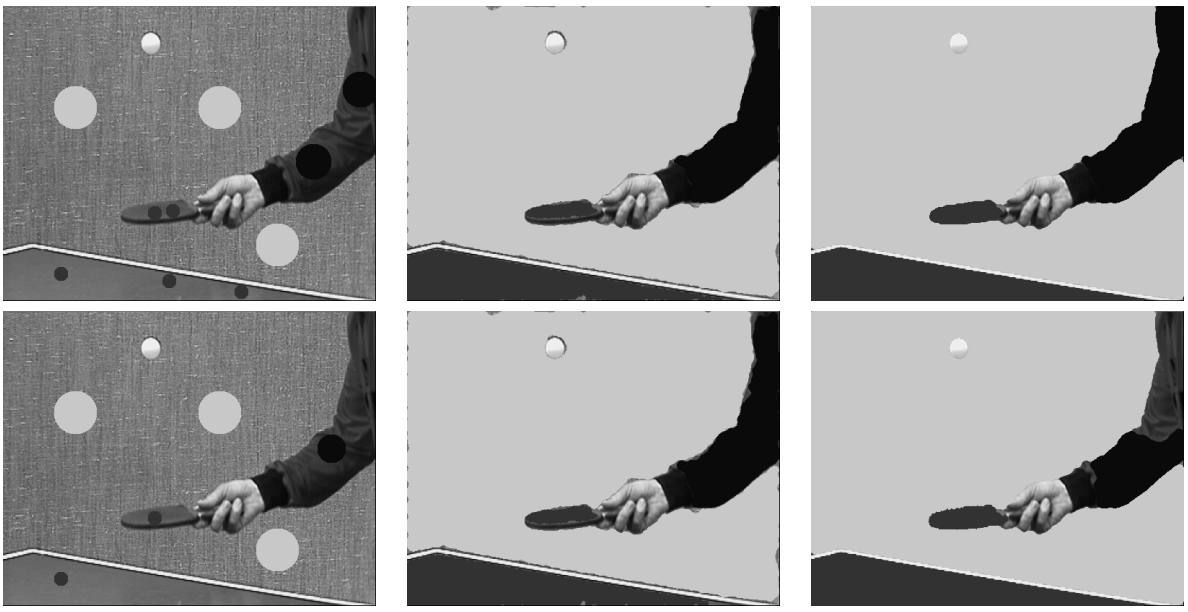
Figure 4.12.: Top row: segmentation with window side lengths of 13px, 17px and 21px and g_{sim}. Bottom row: segmentation with window side lengths of 7px, 13px and 17px and LRDs plus balloon force. (Initializations as in Fig. 4.10(a)). Note that as the size of window increases, the variation of features in the described local region is larger and less well described by a Gaussian model. g_{sim} then tends to separate the sky into dark and light regions, while in LRD plus balloon force result become less accurate.



(a) (b)

(c) (d)

Figure 4.13.: Segmentations obtained with different threshold values for g_{mrf}: (a) $\beta = 3$, (b) $\beta = 4$, and (c) $\beta = 3$, for the racket and table, and $\beta = 5$ for the arm; the background contours were not evolved so that the over- and under-segmentations can be observed. Initializations: for (a),(b) as in Fig. 4.7(a), for (c) as in (d). Note that the influence of a unit change in β has a large effect on the segmented image, due to implementation approximations. For these examples, the particular choices of β lead to bad segmentations.



(a) (b) (c)

Figure 4.14.: Segmentations of a table tennis scene. (a) Two initializations for the table tennis scene. (b) Corresponding segmentations with g_{mrf} for the arm, racket and table; g_{mrf} was chosen to deal with the highlights on these objects; g_{sim} was sufficient in dealing with the background. (See text for parameters). (c) Corresponding segmentations with LRDs plus balloon force ($W = 11px$, $\lambda = 5$).

(a) (b) (c)

Figure 4.15.: Segmentations of a noisy image. (a) Initialization for a noisy image called "Claire". (b) Its segmentation with the segmenter function g_{mrf}, $W = 21px$ and $\beta = 4$. (c) Its segmentation with LRDs plus balloon force, $W = 11px$ and $\lambda = 5$.

Figure 4.16.: A frame from the color sequence Akyio segmented with LRDs plus balloon force ($W = 11px$ and $\lambda = 350$) taking the RGB vector as feature. Initialization (left) and final segmentation (right).

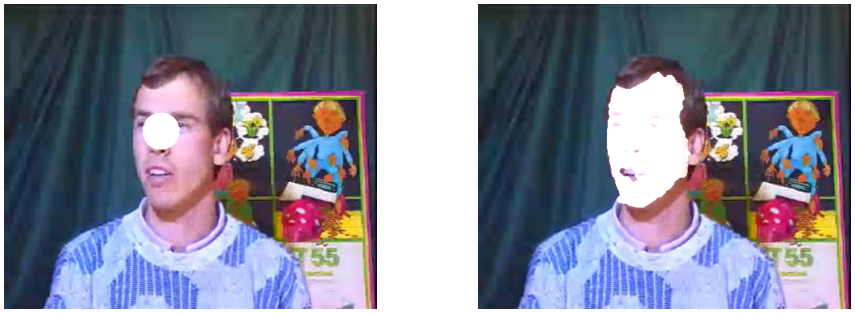


Figure 4.17.: A frame from the color sequence Erik segmented with LRDs plus balloon force ($W = 11px$ and $\lambda = 350$) taking the RGB vector as feature. Initialization (left) and final segmentation (right).

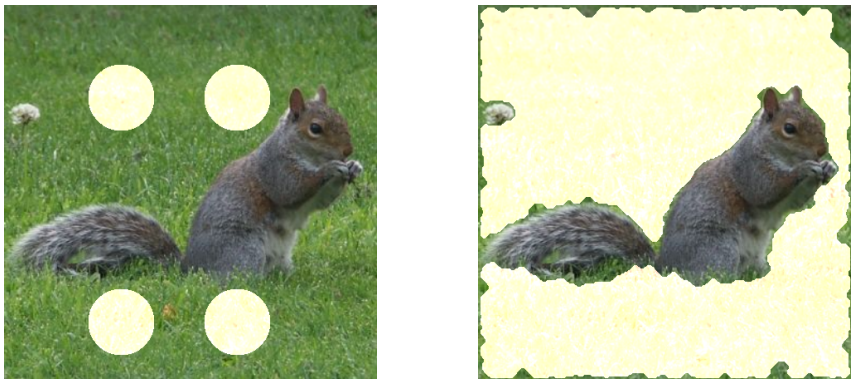


Figure 4.18.: An image of a squirrel segmented with with g_{mrf} taking the RGB vector as feature. Initialization (left) and final segmentation (right).

5. Active Contours with Space-Variant Definitions of Energy Terms

> The reasonable man adapts himself to the world; the unreasonable one persists in trying to adapt the world to himself. Therefore, all progress depends on the unreasonable man.
>
> *George Bernard Shaw*

Active contours are a method for solving the intensively studied image segmentation task. Traditionally, the definition of energy terms for active contours is space invariant, i.e. forces driving the contour are computed the same way at every contour point. In contrast, we propose a framework where the definition of energy terms is space-variant. For this new type of contours, forces at different contour points are computed in different ways. As an example, we introduce contours for which energy terms are defined in one of two ways. These contours are driven by local-statistics forces that are estimated according to either a Gaussian density model, or a kernel density estimator. For a contour point, one of the two forces is chosen depending on the point's position relative to other contours. Multiple such contours allow for segmentation of images based on local region descriptors. Unlike other active contours based only on local information, these contours are initialized far away from object boundaries. Our evaluations showed accurate segmentations on complex natural scenes, based solely on color. Parts of this chapter have already been presented in (Darolti et al., 2008a).

5.1. Spatially-Invariant and Spatially-Variant Definitions of Energy Terms

The data term of a region-based active-contour is defined by quantifying the confidence that every image pixel has been assigned to the correct region (Zhu and Yuille, 1996). The following short overview will show that starting from this fundamental form of the data term, the energy terms grow more and more complex, but their definition remains spatially-invariant.

Researchers have proposed behaviors for active contours which allow them to adapt to more

information from the image or to prior knowledge, or to control the way a solution is formed. For example, linear terms have been added to the fundamental form in order to include in contour evolution additional information about shape (Rousson and Paragios, 2002; Paragios et al., 2002), motion (Mansouri and Mitiche, 2002) or texture (Rousson et al., 2003). The behavior of an active contour is determined by the way the evolution equation is derived, and by the way this equation is implemented. To derive the equation, region descriptors may be considered to be constant during gradient descent (Samson et al., 2000). Region descriptor do in fact depend on the position of the contour, and the shape derivative tool has been introduced to provide an exact equation of motion (Jehan-Besson and Barlaud, 2003). Both types of equations are very often implemented with the level-set method (Osher and Sethian, 1988), which has the advantage that topology changes are intrinsic to the method. This is not the case for the alternative snake method (Kass et al., 1988).

Level-set methods often update the level set according to the equation of motion in a narrow band around the zero level-set (Adalsteinsson and Sethian, 1995). As an effect, new regions cannot form far away from the narrow band, e.g. the white center of a black circle on a white background cannot be segmented if an initial contour is not set inside the white circle. The level-set model introduced in (Chan and Vese, 2001; Li et al., 2005b) provides the possibility for new contours to form far away from the zero level set.

Active contours based on the phase field model are explicitly designed to allow regions to quickly form according to image data starting from a neutral initialization (Rochery et al., 2005). Phase fields have been proposed recently to construct non-linear energies for active contours specialized in segmenting elongated or circular structures (Rochery et al., 2006; Nain et al., 2003; Horvath, 2007). Another special type of non-linear energies are those that combine region and edge information in a non-linear fashion (Lankton et al., 2007; Jermyn and Ishikawa, 1999).

In the methods mentioned above, energy terms remain constant once they have been defined, which means they have spatially-invariant definitions. We propose active contours for which the definition of energy terms is spatially-variant and use them to create a robust method based only on local region descriptors.

Local descriptors must only represent local regions accurately, while global descriptors must model entire image regions. The first assumption (requirement) is less restrictive than the second one. Fig. 5.1 was constructed in order to show an example where objects are characterized by features with pdfs that overlap. The global densities of the star, square, and the background in this image strongly overlap. The boundaries of objects are however very clear, because local regions of each side of a boundary are very different in appearance. If the image is segmented employing global descriptors, Fig. 5.1(d) shows that the image gets separated into dark and light regions, and the shapes cannot be found. The shapes can be retrieved as in Fig. 5.1(c) when employing local region descriptors and contours with space-variant definition of energy terms.

Local region descriptors are dependent on the positions of the initial contours Piovano et al.

(2007); Lankton et al. (2007). These need to cross or be in the vicinity of image boundaries, due to the fact that contours based on local descriptors have many local minima. In the previous chapter, two types of forces were proposed to alleviate this problem, and allow initial contours to be placed well inside the segmented object (for instance as in Fig. 5.1(a)). These solutions yield very good segmentations for images with features that can be locally approximated with Gaussian densities. However, when local pdfs are badly approximated with normal distributions results degrade. In these situations, contours often leak through some pieces of image boundaries, while correctly finding others.

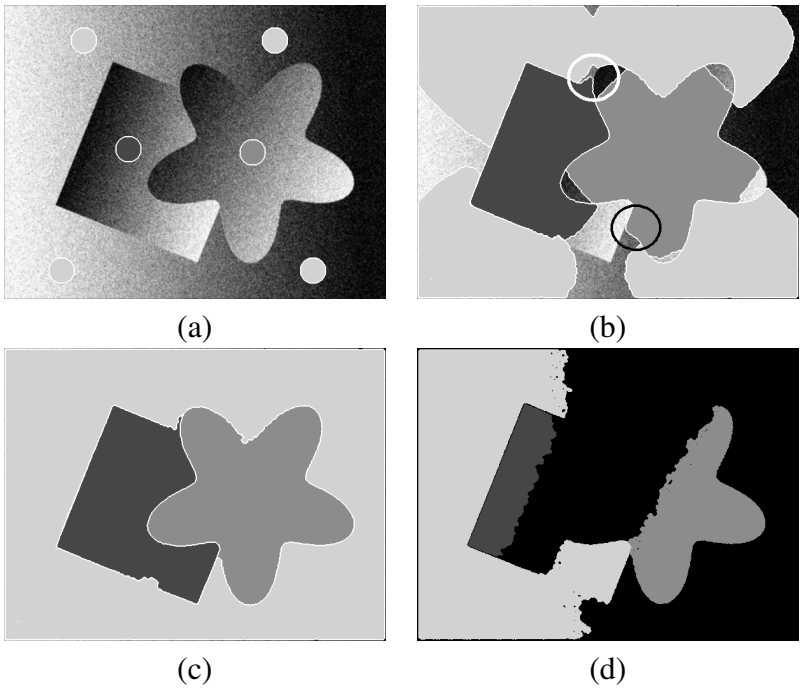


Figure 5.1.: A noisy synthetic image where the global distributions of objects strongly overlap, but most object boundaries are very clear. (a) Initial contours. (b) Intermediary evolution step showing leakage (marked by circles). (c) Segmentation with the proposed contours based on the intensity feature. (d) Segmentation with the region-competition approach (Zhu and Yuille, 1996) that uses global and local statistics.

Active contours with space-variant definition of energy terms are designed to correct leakage by better approximating probability densities that are not Gaussian. The densities of non-Gaussian, multimodal data can be well approximated employing kernel density estimators. However, kernel density estimators are not well suited to drive the contour out of local minima, because the value of a balloon force or a probability density threshold require careful tuning per image. Both solutions are feasible in the situation of local Gaussian distributions, as demonstrated in the previous chapter.

The basic idea of multiple contours associated with the proposed energy is to let initial contours

evolve toward the vicinity of image boundaries by assuming that features in local regions are normally distributed. Once contours are in the vicinity of an image boundary, they compete over local regions by estimating the probability densities of features with a kernel method. However, image boundaries are not known, since they are exactly the items which segmentation must determine. We thus assume that, typically, contours arrive in each other's vicinity when they are close to real boundaries. This assumption relies on the fact that, in many images, local image regions often have normal densities. Many real boundaries can thus be found using the Gaussian density model. However, different contour pieces will approach boundaries at different time steps, which means that in the same iteration, the different types of forces must act on the points of the same contour depending on the points' positions.

Fig. 5.1 provides a concrete example for the problematic addressed here. Driven by forces computed assuming Gaussian densities, contours stop at many image boundaries, while leaking at some other positions marked in Fig. 5.1(b). As multiple contours arrive in each other's vicinity in the marked areas, they start competing. One may notice in this image that pieces of a contour will reach the vicinity of other contours at different times during evolution. In the image, some contour pieces have leaked, while other contours still have to grow toward the boundaries. Thus, the contour cannot behave the same way at all contour points at a given time step.

This chapter introduces a general formulation for a new type of AC for which forces at different points can be computed in different ways, from a space-variant definition of energy terms. To the best of our knowledge, no other such AC exist. For all other ACs, energy terms have space-invariant definitions, which means that the force at every contour point is computed from the same way. The general framework is subsequently exemplified with active contours that are driven by two types of forces, specifically forces computed assuming Gaussian densities or using kernel density methods. The constructed contours can be placed well inside image boundaries, while being robust to initialization and complex images in our experiments. The result of the first experiment is shown in Fig. 5.1(c). In this figure, contours often stop at an image boundary, while region competition corrects the leakage. The marked, and apparently faulty, boundary pieces occur in local regions where an edge between the two regions does not exist, due to the way the local color gradients blend. At these positions, the contours stop at a local equilibrium.

5.2. Active Contours with Space-Variant Definitions of Energy Terms

Let R_i denote one of m image regions, and the entire image domain is $R = \cup_{i=0}^{m} R_i$. We remind that the energy of an AC is very often written as (see Section 3.1.2)

$$E(R_i, p_i, m) = -\sum_{i=1}^{m} \left(\iint_{R_i} \log p_i d\mathbf{x} - \frac{\mu}{2} \int_{\partial R_i} ds \right). \tag{5.1}$$

Let the image domain R be divided into a set of regions such that, for $a \in R$ and $b \in R$ being two regions, $a \cap b = \emptyset$. All pixels in a region a meet some given criterion. These criterion-regions entirely cover the image domain, $\cup_{a \in R} a = R$, as shown in Fig. 5.2. Consider a continuous function $g_{a,i}(\mathbf{f}(\mathbf{x}))$ that generalizes the term $\log p_i$, defined on region $a \cap R_i$. Function $g_{a,i}$ depends on (pdfs of) features $\mathbf{f}$ in the LRDs, and it differs between criterion-regions. Let $\mathbf{1}_a(\mathbf{x})$ be the characteristic function of a criterion-region a. With these notations, we define the energy of an AC with space-variant definition of energy terms:

$$E(R_i, g_{a,i}, m) = -\sum_{i=1}^{m} \left(\iint_{R_i} \sum_{a \in R} \mathbf{1}_a \, g_{a,i} \, d\mathbf{x} - \frac{\mu}{2} \int_{\partial R_i} ds \right). \tag{5.2}$$

The proposed active contours are implemented with the level-set method. If we associate a level-set function ϕ_i to each region R_i, such that the level set function is positive inside and negative outside the region, the energy (5.2) can be written as (Chan and Vese, 2001)

$$E(\phi_i, g_{a,i}, m) = -\sum_{i=1}^{m} \left(\iint_{R} H(\phi_i) \sum_{a \in R} \mathbf{1}_a \, g_{a,i} - \frac{\mu}{2} |\nabla H(\phi_i)| \right) d\mathbf{x}, \tag{5.3}$$

where $H(x)$ is the Heaviside function or its regularized version and ϕ_i is the level-set function that represents region R_I. The AC of minimal energy is found employing the gradient descent method (Strang, 2008). The energy's gradient is obtained by taking the derivative of E with respect to time, in a general direction v.

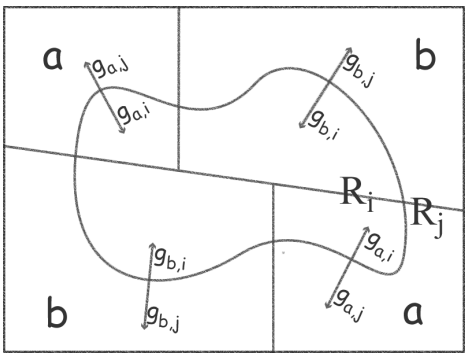


Figure 5.2.: A contour with piecewise constant behavior on a domain $R = R_i \cup R_j$. The image domain is divided into criterion-regions of two types, a and b. Each image region $R_i \cap a$ generates a statistics-force according to a function $g_{a,i}$. A function $g_{a,i}$ applied at pixels in a criterion-region a is different from a function $g_{b,i}$ applied at pixels in a criterion-region b.

The functional E is discontinuous at points on the boundary between criterion-regions, where the function $g_{a,i}$ is changed, for example to $g_{b,i}$. The energy E is thus only piecewise differentiable. Theoretically, the evolving contour thus obtained will be discontinuous. In practice, this problem

can easily be dealt with. In most level-set methods, each contour point moves with speed of one pixel, at maximum. It can thus be enforced that contour pieces stay connected, e.g. with the fast level-set method (Shi and Karl, 2005a). This is meaningful because the objects to be segmented have smooth, connected boundaries and we expect the image data to guide contours toward these boundaries. We take an expectation maximization approach for derivation, and consider that LRDs do not depend on level-set functions ϕ_i. We obtain the following flow

$$\frac{\partial \phi_i}{\partial t} = H'(\phi_i) \left(\sum_{a \in R} \mathbf{1}_a (g_{a,i} - \max_{j \neq i, H(\phi_j) > 0} g_{a,j}) + \mu \left(div \left(\frac{\nabla \phi_i}{|\nabla \phi_i|} \right) \right) \right), \tag{5.4}$$

where $H'(x)$ represents the derivative of $H(x)$, which needs to be defined. In equation (5.4), the term $(g_{a,i} - \max_{j \neq i, H(\phi_j) > 0} g_{a,j})$ is a region competition term, and R_j are regions that neighbor R_i at pixel $\mathbf{x}$. This data term can deal with junctions of multiple regions because it ensures that a pixel will be assigned to one segmentation region only. We denote the curvature term with $k_i \equiv div(\frac{\nabla \phi_i}{|\nabla \phi_i|})$; this term can cause slight overlaps or gaps between contours. Again, this is only a theoretical problem. In practice, these pixels can also be uniquely assigned to segmentation regions, for example, according to the data term.

Regions a are disjoint by definition, and thus, for a contour point $\mathbf{x}$, only one of the functions of the characteristic functions $\mathbf{1}_a$ may take the value 1 at $\mathbf{x}$, while all others will take the value 0. It follows that in the proposed equation of motion (5.4), the role of the characteristic function $\mathbf{1}_a$ is to select which of region forces $g_{a,i}$ will be applied at each contour point. The characteristic function of a criterion-region thus switches between possible definitions of energy terms. There are as many possible definitions, as there are given criteria. The terms that determine region forces vary depending on the position $\mathbf{x}$, i.e. the forces have a spatially-variant definition. In the following section we present an example where regions a depend on the number of contours that neighbor each other locally. The selector functions $\mathbf{1}_a$ select between local regions where only one contour is present, and local regions where more contours are present.

The data term in equation (5.4) will assign the point to the region that exerts the largest force on the point, while the curvature term ensures that the contour stays smooth.

5.3. Two Energy-Term Definitions Based on Different Local Density Models

This section introduces multiple contours with energy terms defined in one of two ways. More specifically, the energy terms are defined using local region descriptors with probability densities estimated assuming normal distributions or kernel density models. These active contours are thus a particular case of the general framework proposed in the previous section. They yield a segmentation method that builds on active contours driven by LRDs with Gaussian densities, but

it is more robust than these.

Active contours driven only by LRDs with Gaussian densities were proven to be a successful method for segmenting individual objects (see Chapter 4). Many local regions in many images can be well approximated with normal densities. On the other hand, local regions are often not Gaussian. For instance, in the image in Fig. 5.3 the tail of the squirrel is badly modeled using Gaussian densities. In the intermediary evolution step in Fig. 5.3(b) locations where contours evolve using Gaussian distributions are marked. At these locations, contours are not locally neighbored by other contours. The ellipse-shaped markers enclose regions where boundaries have been found correctly. The square-shaped marker points out a location where the contour has leaked because the bimodal density in the region of the tail (which contains extreme dark and light pixels) has been approximated with a Gaussian model. Changing the density estimation method when contours approach one another at this location not only corrects leakage, but also yields visually accurate results, as can be observed in Fig. 5.3(c).

(a) (b) (c)

Figure 5.3.: Exemplary segmentation on a squirrel image. (a) Initial contours. (b) Segmentation with LRDs with Gaussian densities: many boundary pieces are found, but the contour for the squirrel leaks into the grass. The circles mark some of the correctly found boundaries; the square marks the leakage. (c) Accurate segmentation with the proposed contours. (The chosen feature for the LRD is pixel color.)

The result in Fig. 5.3(c) is due to employing a kernel density estimator where the Gaussian model fails, as will be shown in the following. A KDE can be very specific in representing the underlying data. This property makes KDEs suitable for accurately determining the boundary between two distinct real regions by letting neighbored contours compete. However, KDEs lack the power to generalize and they are thus not as good at growing contours out of local minima as normal approximations of pdfs are. We thus need a contour for which some contour pieces are driven by Gaussian LRDs, while others compete by kernel estimated LRDs. We need to define the criterion-regions accordingly, and then write the corresponding energy. Even if motion by Gaussian LRDs causes a contour to both grow and shrink, they are mostly intended to grow; for

shortness we use the attribute "growing" to refer to this motion.

In order to decide which of the two models is employed at a point, we check if the there are other contours present in the vicinity of the point. We obtain two complementary criteria for defining criterion-regions. For the first types of criterion-regions there are no other contours present in the vicinity of a pixel on a contour. For the second type of criterion-regions, other contours can be present in the vicinity of a pixel on a contour.

Formally, let there be m initial contours that delimit regions $R_i, i = 1...m$. We denote with R_0 the set of pixels not yet assigned to regions R_i. In order to determine the number of contours present in the vicinity of a pixel, we employ a window $W(\mathbf{x})$ centered on the pixel. One type of criterion-region contains all pixels with only one contour present in this window. The remaining pixels belong to the other type of criterion-region.

Using set operations, the criterion for the first type of criterion-region can be written as $W(\mathbf{x}) \cap_{j=1, j\neq i}^{m} R_j = \emptyset$, which means that the local window does not intersect other regions than R_0 and R_i. Let us denote the characteristic function of this set with $\mathbf{1}_i^W$. For pixels in this region we wish to compute a growing force $G(\mathbf{x})$ based on LRDs with normal densities. Also, for pixels in this region we have $\mathbf{1}_i^W = 1$. The criterion for the second type of regions is $W(\mathbf{x}) \cap_{j=1, j\neq i}^{m} R_j = \emptyset$. For pixels in this region we wish to compute a competing force $K(\mathbf{x})$ based on KDE. Let us denote the characteristic function of this set with $\mathbf{1}_i^{C_W} = 1 - \mathbf{1}_i^W$.

Characteristic functions $\mathbf{1}_i^W$ and $1 - \mathbf{1}_i^W$ replace characteristic functions $\mathbf{1}_a$ without changing the energy (5.3). This means that $\mathbf{1}_i^W$ is now the function switching between energy terms. With this switching function, the energy of multiple ACs that grow and compete is written as a special case of (5.3):

$$E_{GK}(\phi_i, G, K, m) = -\sum_{i=1}^{m} \left(\int\int_R H(\phi_i) \left(1_i^W (G + c) + (1 - 1_i^W) K\right) - \frac{\mu}{2} |\nabla H(\phi_i)| d\mathbf{x} \right). \quad (5.5)$$

The growing force $G(\mathbf{x})$ and the competing force $K(\mathbf{x})$ are based on Gaussian pdfs $p_i^{G,W}$, and kernel density estimated pdfs $p_i^{K,W}$, respectively, and will be defined in the following. The index i denotes the local region within window W for which the pdf is approximated. The constant c will be defined when deriving the growing force $G(\mathbf{x})$.

The Growing Force. Two methods for creating a region-growing term that reduces the number minima for energy (5.5) were proposed in (Darolti et al., 2008b). The first one involves adding a balloon force λ to region competition forces generated by LRDs with normal pdfs (remember Section 4.2 and its notations):

$$E(\phi_i, p_1^{G,W}, p_0^{G,W}) = -\int_R H(\phi_i)(\log p_i^{G,W}(f(\mathbf{x})) + \lambda) + (1 - H(\phi_i)) \log p_0^{G,W}(f(\mathbf{x})) - \mu |\nabla H(\phi_i)| \, d\mathbf{x}. \quad (5.6)$$

Indirectly, the balloon force λ reflects the minimal difference between two neighbored image patches that belong to different objects. The index 0 represents the set of pixels not yet assigned to other regions.

With the second method, a contour grows as long as outside pixels are similar to inside ones. A feature f is assumed to have a Gaussian local pdf p_i^W, with mean m_i^W and standard deviation s_i^W, and pixels in an interval around the mean are considered to be similar. The size of the interval, $c_\alpha s_i^W$, is determined by fixing the probability α of the interval, i.e.

$$\int_{m_i^W - c_\alpha s_i^W}^{m_i^W + c_\alpha s_i^W} p_i^W(f(\mathbf{x}))d\mathbf{x} = \alpha. \tag{5.7}$$

We can directly derive a function g_i^W to reflect the amount of similarity as

$$g_i^W(c_\alpha, f, s_i^W, m_i^W) = c_\alpha s_i^W - |f(\mathbf{x}) - m_i^W|. \tag{5.8}$$

This definition is extended for a feature vector $\mathbf{f} \in R^d$, with mean $\mathbf{m}_i^W$ and covariance matrix Σ_i^W, assumed to be diagonal. For this purpose, we employ the χ_d^2 distribution with d degrees of freedom:

$$g_i^W(\alpha, \mathbf{f}, \mathbf{m}_i^W, \Sigma_i^W) = \chi_d^2(\alpha) \; - \; (\mathbf{f}(\mathbf{x}) - \mathbf{m}_i^W)' \, (\Sigma_i^W)^{-1} (\mathbf{f}(\mathbf{x}) - \mathbf{m}_i^W). \tag{5.9}$$

We create an energy based on g_i^W similar to (5.6). The two growing methods behave differently, as local pdfs deviate more and more from the normal pdf (Darolti et al., 2008b). For both methods, the contours may leak or stop prematurely, but they often do so in different situations. We would rather have contours leak, since leakage can be corrected by the competing force K. We thus propose to compute the region growing term with both methods and select the stronger one:

$$\begin{aligned} E(\phi_i) = -\int_R H(\phi_i) \left(\log p_i^{G,W_1}(f(\mathbf{x})) + \max(\lambda, \nu g_i^{W_2}) \right) + \\ (1 - H(\phi_i)) \log p_0^{G,W_1}(f(\mathbf{x})) - \mu |\nabla H(\phi_i)| \, d\mathbf{x}. \end{aligned} \tag{5.10}$$

The size of the local windows for computing the different terms may be different: to compare the balloon force λ with the difference between local pdfs, we may choose a window of size W_1, while for computing the similarity term g_i^W we may choose a window W_2. The role of the parameter ν is to bring $\log p$ and l in the same numerical range, and it is kept constant in our implementation. The energy (5.10) can be interpreted as follows: two methods are combined in order to test if two local regions separated by the contour belong to separate image objects: 1) a logarithmic term that compares statistics of local regions inside and outside the contour requiring them to be minimally different, and 2) a term that assimilates pixels which may be observations of the Gaussian distribution of the local inside region. The corresponding contour evolution can be written as

$$\frac{\partial \phi_i}{\partial t} = H'(\phi_i)(G(\mathbf{x}) + \mu k_i), \;\; G(\mathbf{x}) = \log p_i^{G,W_1} - \log p_0^{Ga,W_1} + \max(\lambda, \nu g_i^{W_2}). \tag{5.11}$$

At this point we define the constant c as $c = \log p_0^{G,W_1}(\mathbf{f}(\mathbf{x}))$. Because we select the stronger region force for each contour pixel, the combined strategy grows faster than its individual parts. It can still correctly find many real boundaries. Once the contour is close to object boundaries, leakage is corrected by the the competing term introduced next.

The Competing Force. Images are usually composed of multiple regions, and thus multiple contours are employed for segmentation. Multiple contours driven by the growing force may overlap. The competition term is needed in order to assign pixels to regions uniquely. In general, this term is constructed based on the maximum likelihood criterion. Using KDEs we can ensure that a pixel will be assigned to the region with the largest number of pixels similar to it. In turn, this ensures that object boundaries are detected accurately when the competing contours are in their surrounding areas.

The windows for defining LRDs with kernel density estimated pdfs can be different from windows W_1 and W_2 employed in defining the growing force G. We will denote these windows with W_3. The evolution associated with the competing term has a classical form, but is computed from pdfs $p_i^{K,W}$:

$$\frac{\partial \phi_i}{\partial t} = H'(\phi_i)K(\mathbf{x}), \;\; K(\mathbf{x}) = \left(\log p_i^{K,W_3} - \max_{j \neq i, \phi_j > 0} \log p_j^{K,W_3} \right). \tag{5.12}$$

In our implementation, kernel density estimate for a local region is obtained by building a histogram from intensity or color observations found in the local region delimited by a window W_3 and smoothing it with a computationally inexpensive constant Epanechnikov kernel.

Having defined forces $G(\mathbf{x})$ and $K(\mathbf{x})$, the gradient descent flow for energy (5.5) with respect to each level-set function ϕ_i can be written out as

$$\frac{\partial \phi_i}{\partial t} = H'(\phi_i)\left(\mathbf{1}_i^W G(\mathbf{x}) + (1-\mathbf{1}_i^W)K(\mathbf{x}) + \mu k_i\right) = H'(\phi_i)\Big(\mathbf{1}_i^W(\log p_i^{G,W_1} - \log p_0^{Ga,W_1} + \max(\lambda, \nu g_i^{W_2})) + (1-\mathbf{1}_i^W)(\log p_i^{K,W_3} - \max_{j \neq i, \phi_j > 0} \log p_j^{K,W_3}) + \mu k_i\Big). \tag{5.13}$$

In this evolution equation, one may observe the characteristic function $\mathbf{1}_i^W$, which selects only one of the two motion forces, such that the other one is always zero for any given contour pixel. The function selects the term based on the Gaussian model if the contour is not locally neighbored by other contours, and it selects the term based on kernel estimation if the contours is neighbored locally by another contours. The associated contour selects the motion force at each contour pixel.

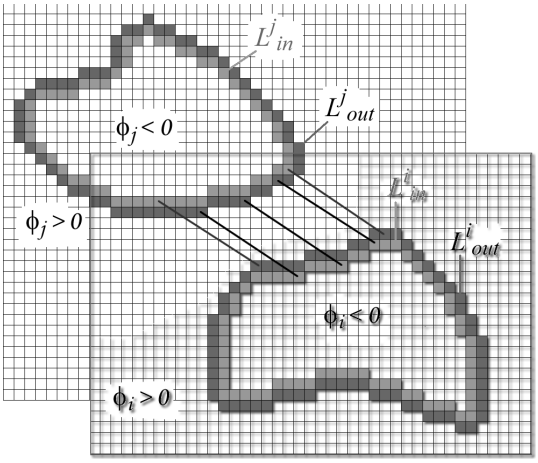


Figure 5.4.: Two level-set functions with highlighted list pixels, representing two different regions. The lines highlight the double representation of the boundary between connected regions.

5.4. Fast Implementation and Stopping Condition

Our implementation closely follows the fast level-set algorithm introduced in (Shi and Karl, 2005b). We remind that the level-set function ϕ is a matrix that takes one of the values $\{-3,-1,1,3\}$. and the positions of boundary pixels are recorded in two lists.

The original elegant algorithm (see subsection 3.6.4) has been modified in order to not to have to compute topological numbers (Shi and Karl, 2005b) for each pixels, at the cost of increased memory space due to redundant representation. The modification employs one level-set function and a set of lists per curve. The boundary between contours lies between one set of boundary pixels. The boundary pixels in the inside list of one contour are also listed in the outside list of the other, as shown in Fig. 5.4. The speed for pixel is however computed only once, using a flag matrix to memorize computed speeds.

The region indicator matrix ψ is also needed. The $switch_in(\mathbf{x})$ procedure checks in this matrix if the pixel that was switched in belonged to the background region. If the pixel belonged to another object region, a $switch_out(\mathbf{x})$ procedure must be performed for the neighboring contour to correct its level-set function. These pixels are stored in a *correct* list.

We also provide an alternative stopping condition. If forces should be positive for all pixels in L_{in}, and negative for all pixels in L_{out}, the energy must have a unique local minimum. Often, this is the case when the boundary is delimited by step-edges, but it is not the case for ramp- and roof-edges. In the experiments made to provide the results in this thesis, the first condition was not met because the contour oscillated within a narrow interval until the maximum number of

iterations was reached. Execution time was thus unnecessarily increased.

A novel stopping condition is defined in order to reduce execution time. The idea is to detect and count oscillations, and set the maximum number of allowed oscillations. Two arrays *memlin* and *memlout* are used to memorize the number of times a given pixel has been added to one of the two lists. The arrays are updated in each iteration when a pixel is added to one of the lists. For a faster implementation, we do not count the number of oscillations. Instead, every few iterations (e.g. 10) the two arrays are cleared. To stop the algorithm, we check in the arrays if all pixels in the current lists have been crossed by the respective list at least two times. The stopping condition *check_stop* can be written in pseudocode:

for each contour $i, i = 1, m$

- for all $\mathbf{x} \in L^i_{out}$, **if** ($memlout(\mathbf{x}) < 2$ and $\forall \mathbf{y} \in N_4(\mathbf{x})$ $memlin(\mathbf{y}) < 2$) **then** return false

for each contour $i, i = 1, m$

- for all $\mathbf{x} \in L^i_{in}$, **if** ($memlin(\mathbf{x}) < 2$ and $\forall \mathbf{y} \in N_4(\mathbf{x})$ $memlout(\mathbf{y}) < 2$) **then** return false

return true.

We do not prove that this stopping condition is met for any given image, but in practice, we have employed it successfully. The details of the propagating algorithm employed in this dissertation are as follows:

The algorithm is run in two cycles (Shi and Karl, 2005b): first the contour is evolved according to the sign of the speed; second, the contour is smoothed by convolving the level-set function with a Gaussian kernel. Finally, for large images, the algorithm is implemented in a multi-scale approach, such that the result of segmentation on a coarser scale is used to initialize the segmentation on the next finer scale.

5.5. Experiments with Contours Driven by Two Local Density Models

We have employed the novel active contours driven by two density models in the segmentation of gray and color images. To describe the regions locally, simple intensity or RGB color features were chosen for all images and contours were evolved according to (5.13). To create initial contours, circles of fixed radius are generated around a few user-selected pixels in each region. This semi-automatic initialization can be an advantage, since it allows for a fast input of user knowledge.

Table 5.1.: Propagate-multiple-curves algorithm

Initialize Φ_i, L^i_{in}, L^i_{out}, $i = 1,m$, and ψ.
Compute forces
Evolve contours
Initialize *memlin* and *memlout*; *count* = 0.
While *check_stop* returns false

- For each contour $i, i = 1,m$
  - Set *correct* to false.
  - For each $\mathbf{x} \in L^i_{out}$ with $F(\mathbf{x}) > 0$ set $correct(\psi(\mathbf{x})) = true$ and $switch\_out(\mathbf{x})$.
  - For each $\mathbf{x} \in L^i_{in}$ with $(\forall \mathbf{y} \in N_4(\mathbf{x}), \Phi_i(\mathbf{y}) < 0)$ delete $\mathbf{x}$ from L^i_{in}, $\Phi_i(\mathbf{x}) = -3$.
  - For each contour j, $j = 1,m$ with $correct(j) = true$
    - For each $\mathbf{x} \in L^j_{in}$ with $\Phi_j < 0$ $switch\_out(\mathbf{x})$ without changing $\psi(\mathbf{x})$.
    - For each $\mathbf{x} \in L^j_{out}$ with $(\forall \mathbf{y} \in N_4(\mathbf{x}), \Phi_j(\mathbf{y}) > 0)$ delete $\mathbf{x}$ from L^j_{out}, $\Phi_j(\mathbf{x}) = 3$.
- For each $\mathbf{x} \in L^i_{in}$, $i = 1,m$ increment $memlin(\mathbf{x})$; for each $\mathbf{x} \in L^i_{out}, i = 1,m$ increment $memlout(\mathbf{x})$;

The shape of windows of LRDs may be circular or square. We choose square windows because the implementation is then faster. All computations are reduced to integer arithmetics, also in order to obtain a faster algorithm. For LRDs with Gaussian densities, the window sides, measured in pixels (px), are $W_1 = 11px$ for the logarithmic terms and $W_2 = 21px$ for computing the term $g_i^{W_2}$, which were found to be good window sizes in Chapter 4. In the same work, good values for the balloon force λ were determined: for gray-level images $\lambda = 5$ and for color images $\lambda = 350$. In practice, the balloon force added to the logarithmic term works similar to an edge detector. It compares the similarity of pixels on the two sides of an edge within a mask of 11×11 pixels. The term based on the similarity function $g_i^{W_2}$ works like a region growing term.

The parameter ν that multiplies $g_i^{W_2}$ is set to 1. The smoothing term μk_i is implemented by smoothing the level-set function with a constant Gaussian kernel of $5 \times 5px$ and $\sigma = 1.5$. This corresponds to a small value of μ, and it preserves thin structures in the segmentation. The size of the window W for determining the number of contours present in a region and $\mathbf{1}_i^W$ was varied from $3px$ to $21px$; our method proved not to be sensitive to the value of this parameter. For all images, W was set to $7px$. The only parameter varied for a particular image was the size of windows W_3 for computing the competition term. This window should be chosen large enough as to include samples from both sides of an object boundary where the active contour leaks. On the other hand, the larger the window, the larger the computational cost. For Figures 5.5-5.12, W

was 51px, 61 px, 31px, 61px, 51px, 51px and 31px, respectively.

For comparison, we show segmentations obtained with the region-competition approach by Zhu and Yuille (1996) because this well-known method also uses local statistics for contour evolution. But local statistics do not take into account the contour's position and they are combined with global information during evolution. Both local and global information are modeled using Gaussian distributions. Also, this method unifies many other statistical approaches by accurately describing the segmentation problem as a Bayes-error minimization problem. For example, active contours without edges are a special case of contours proposed in (Zhu and Yuille, 1996) formulated using level-sets. The region-competition approach (Zhu and Yuille, 1996) was applied both using the color model proposed in (Zhu and Yuille, 1996) and by using RGB color. The size of the local window was $7px$ and the influence of the smoothing term was the same as in the proposed method.

Color images were processed in a multi-scale approach, using a Gaussian pyramid with two levels. Figures 5.7-5.12 show the initial step (at the coarsest scale), an intermediary step and the final segmentation (at the finest scale) for the respective image. Contours are shown in the original image to allow for a direct visual evaluation of the result[1], and as contour mask to allow for readability in colorless print. Each figure also presents segmentations obtained with the region competition approach Zhu and Yuille (1996). These result are also shown in color and gray scale. Since for this method gaps may be formed in the segmentation, we show filled colored regions instead of contours.

Images of intermediary evolutions steps in Figs. 5.5 - 5.10 show contours that have grown over the real boundaries of the region they represent. One can also observe in these images that many boundaries are found correctly. The mistakes were corrected by the competition term.

The semi-automatic initialization allows us to obtain segmentations with different numbers of regions for the same image, by specifying the number of objects in the image. This property is illustrated in Fig. 5.5, where the landscape is first segmented into foreground, background and trees, and then into background, grass, trees and bovine. In Fig. 5.9 we can specify that we wish to have only one background object. This background object is composed of leaves and dark patches that are very different in appearance, but its segmentation is nonetheless very good. The flexible initialization also has the advantage that segmented objects may contain very strong edges. The complex images in Figs. 5.6 and 5.10 pose a difficult segmentation problem for region-based methods that do not include shape knowledge. This is because the sail (Fig. 5.6), the dog and the person (Fig. 5.10) are composed of dark and bright regions that are very different in appearance, but similar to other image regions. With the proposed method, we only need to place initial contours such that they cross edges not representing image boundaries in order to obtain very good results.

Initial contours have little influence on the result, as long as they are reasonably distributed over

[1]Please zoom in the electronic version for details.

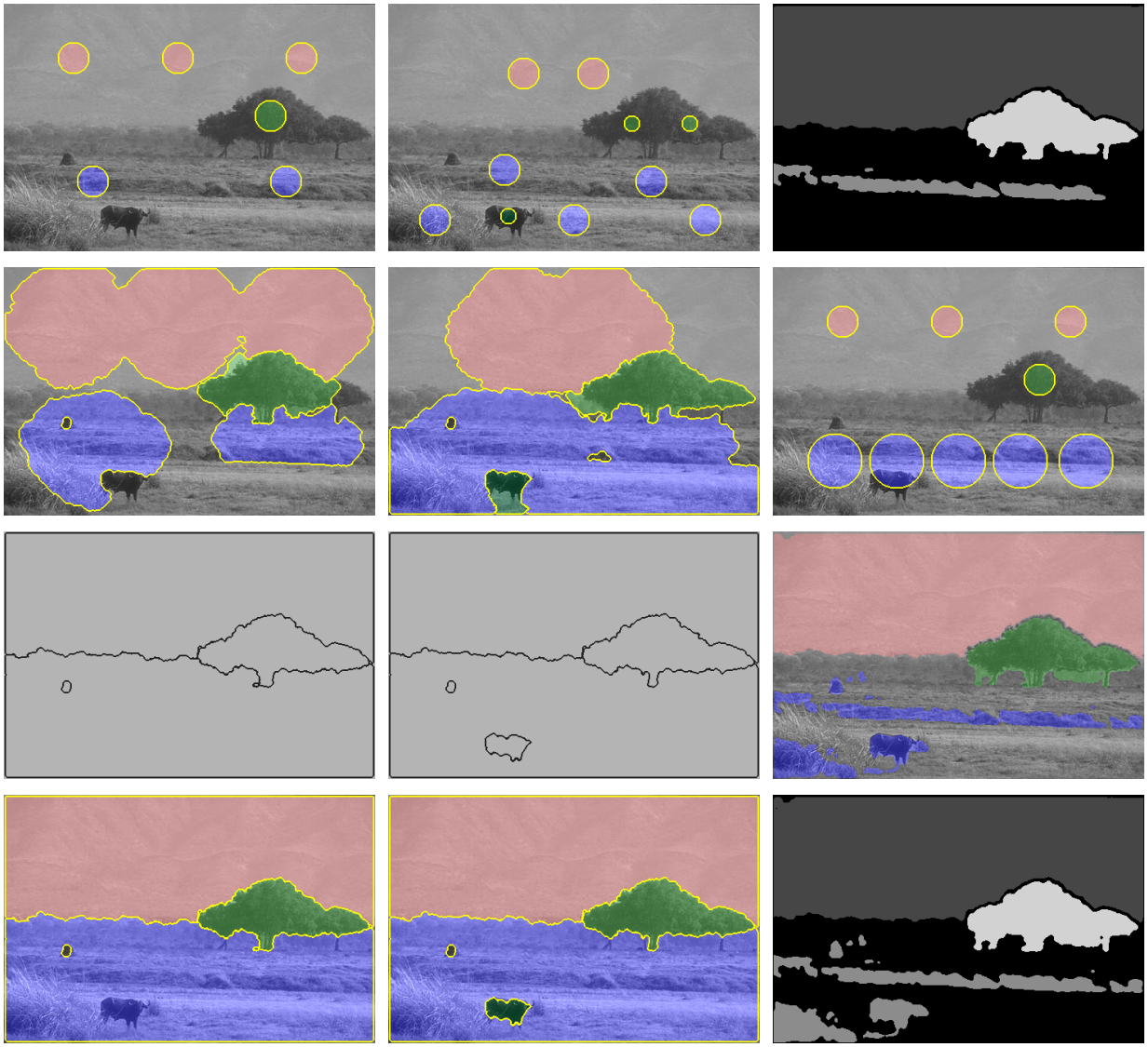


Figure 5.5.: Segmentation of a landscape. Left and middle row: Segmentation with the proposed method starting with two different initializations. Top to bottom: Initial contours; an intermediary step showing leakage; the final segmentation shown as contour mask; the final contour and regions superimposed on the image. Right: Segmentation with the region-competition approach (Zhu and Yuille, 1996). Top to bottom: Segmentation starting with the same initialization as for the top segmentation shown as region masks; a second initialization that covers a larger area of the grass region and corresponding segmentation shown superimposed on the image and as regions mask.

the object's area. For example, two different initializations for the landscape image lead to very similar segmentations of background, trees and grass, as can be seen in the top and bottom rows

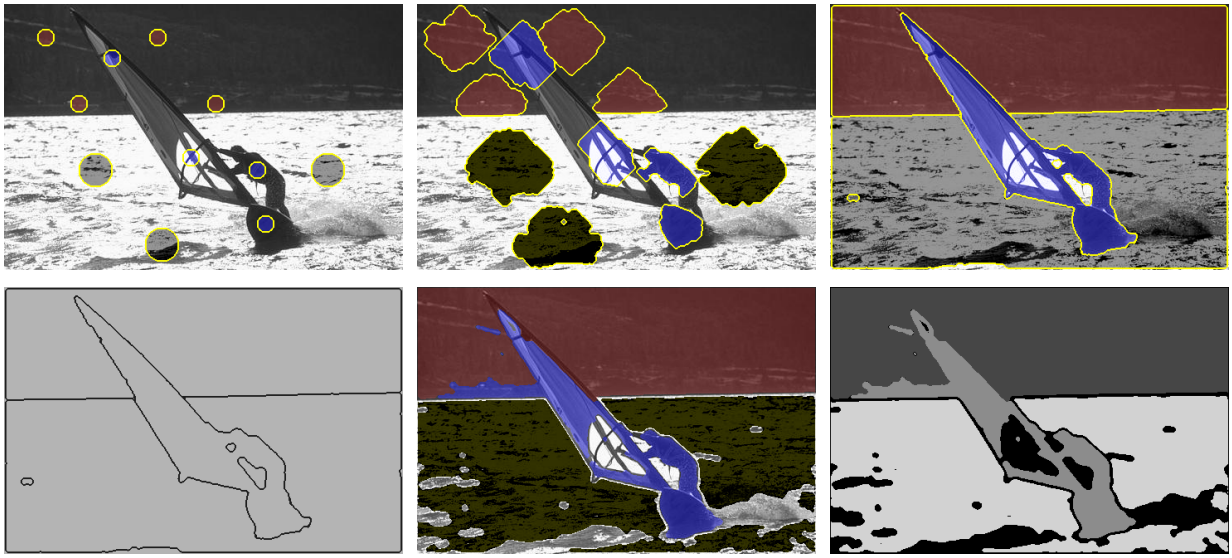


Figure 5.6.: Segmentation of an image of a surfer. Top: Segmentation with the proposed method. Left to right: Initial contours; an intermediary step showing leakage; the final segmentation superimposed on the image. Bottom, left to right: the final segmentation shown as contour mask; segmentation with the region-competition approach (Zhu and Yuille, 1996) starting with the same initialization as for the top segmentation, shown superimposed on the image, and as regions mask.

of Fig. 5.5. Very similar segmentations can also be observed for the squirrel image in Figs. 5.3 and 5.7. Even though initial contours are far away from object boundaries in all figures, the final segmentations are very accurate. Accuracy is due to LRDs with kernel estimated densities, which reliably describe regions locally.

The proposed method allows smooth transitions in an object's visual properties. For example, there is not a smooth strong edge between the cow's horns and fur in Fig. 5.8, and, as a result, the entire head can be segmented. Also, in Fig. 5.9, the koala's fur changes from dark gray to light gray.

With the region-competition approach Zhu and Yuille (1996), not all boundaries could be found due to some problems that particularly affect global distributions; they either overlap, in which situation the contour may leak over the boundary, or the distribution is multi-modal, in which situation the one region may be fragmented. In Fig. 5.6 bottom, we may observe leakage and fragmentation, while in Fig. 5.5 the grass is fragmented into dark and light patches, even though the boundaries between these patches are not very strong. In Figs. 5.8 and 5.9 not all boundaries could be correctly found, neither by using the color model in Zhu and Yuille (1996) nor by using RGB color, because global densities overlap. For example, there are many dark pixels in the koala's fur. These attract into the koala region dark pixels above the koala's head that are not as

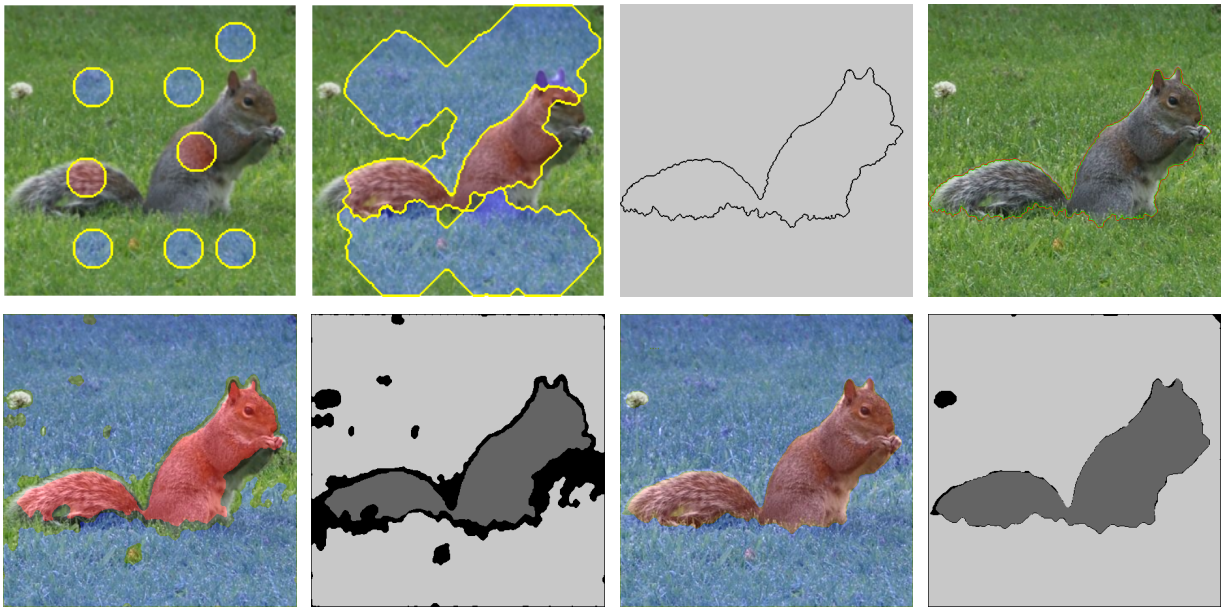


Figure 5.7.: Segmentation of an image of a squirrel. Top: Segmentation with the proposed method. Left to right: Initial contours at the coarsest of 3 scales; an intermediary step showing leakage; the final segmentation shown as contour mask; the final contour at the finest scale, superimposed on the image. Bottom: Segmentation with the region-competition approach Zhu and Yuille (1996) starting with the same initialization as for the top segmentation. Left to right: Segmentation with the color model from Zhu and Yuille (1996) superimposed on the image and shown as regions mask; segmentation with RGB color superimposed on the image and as regions mask.

dark as other pixels in the background. The region-competition method can segment the squirrel in image Figs. 5.7 when using the RGB color, and the result is similar to the one obtained with the proposed method. When using the color model in Zhu and Yuille (1996), the squirrel's lighter paw and lighter grass patches remained in the background.

The final result in Fig. 5.9 contains a slight glitch in the segmentation of the koala, but it is comparable to the segmentation in Brox and Weickert (2004a); however, in that paper the observed features are color, the entries of a structure tensor and a local scale measure for texture. The segmentation of the flying dog in Fig. 5.10 also contains glitches under the left ear, the right paws and the person's hand. These are due to specularities creating light pixels similar to those in the fur, shadows creating dark pixels similar to those in the person's hair, and a strong color gradient in the hand. These glitches may be corrected only by placing initial contours very close to these regions. The segmentation of the same image with the region competition method is shown in Fig. 5.11.

Fig. 5.12 shows details of two slices of a computer tomography (CT) image of a lung tumor

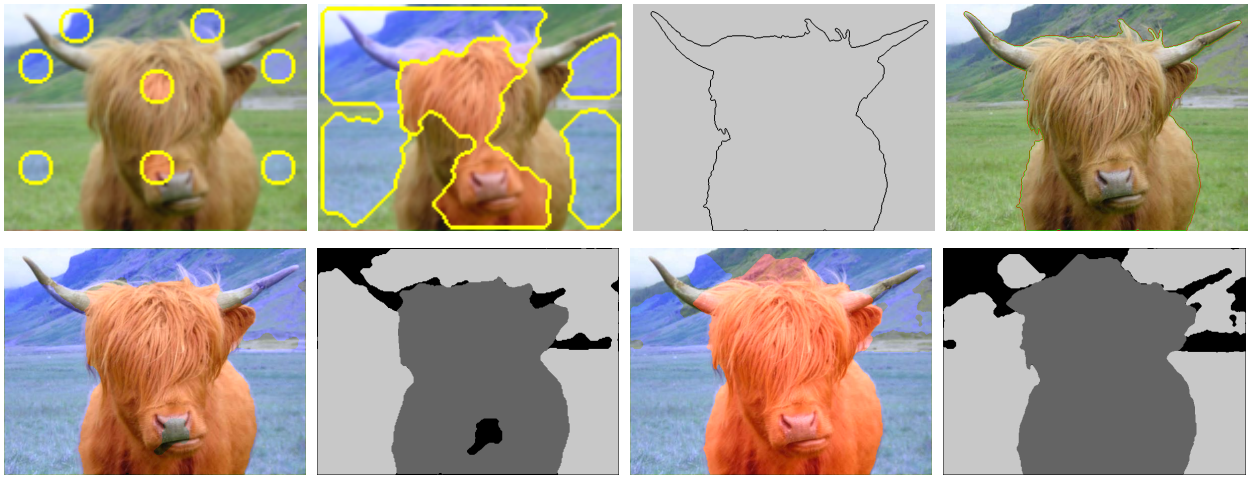


Figure 5.8.: Segmentation of an image of a cow. Top: Segmentation with the proposed method. Left to right: Initial contours at the coarsest of 3 scales; an intermediary step showing leakage; the final segmentation shown as contour mask; the final contour at the finest scale, superimposed on the image. Bottom: Segmentation with the region-competition approach Zhu and Yuille (1996) starting with the same initialization as for the top segmentation. Left to right: Segmentation with the color model from Zhu and Yuille (1996) superimposed on the image and shown as regions mask; segmentation with RGB color superimposed on the image and as regions mask.

attached to the lung wall. There is no difference in the visual characteristics of lung wall and tumor. However, the tumor is not compactly attached to the wall, and the boundary between the two is revealed by the small dark patches. This tumor thus poses a difficult medical image segmentation problem. With the proposed method, it suffices to place two initial contours around and inside the tumor in order to obtain a reliable segmentation.

Computation times for relatively large images are relatively small. The squirrel in Fig. 5.7 (450×450 pixels) has been segmented in 2.1 seconds, the cow in Fig. 5.8 (600×450 pixels) in 5.5 seconds and the koala in Fig. 5.9 (170×250) in 2.9 seconds; the radiographs in Fig. 5.12 (140×120 pixels) have all been segmented in under 0.3 secods.

In summary, in the active contours framework, definitions of energy terms are spatially-invariant, which means that at every contour point driving forces are computed the same way. In this paper, we have proposed a formulation for evolving contours with spatially-variant definition of energy terms. Theoretical issues concerning the functional's continuity and their solution in practice have been discussed.

The proposed method has been concretized as active contours that have energy terms defined in one of two ways. These contours are driven by two types of forces derived from local statistics. Multiple such contours have been employed in order to segment difficult natural images. Ac-

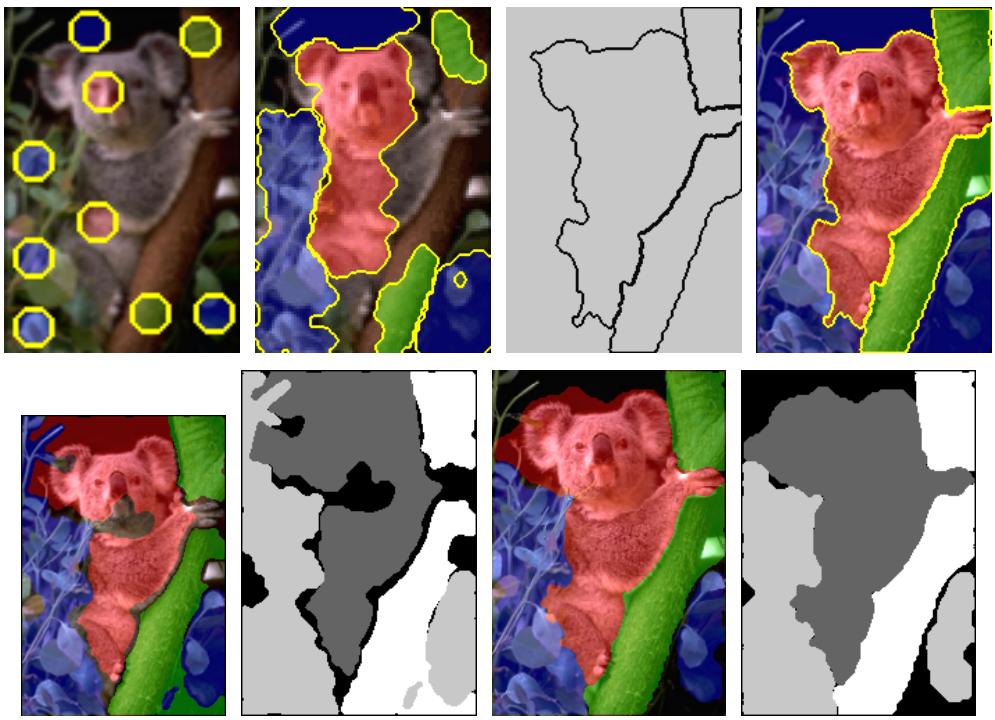
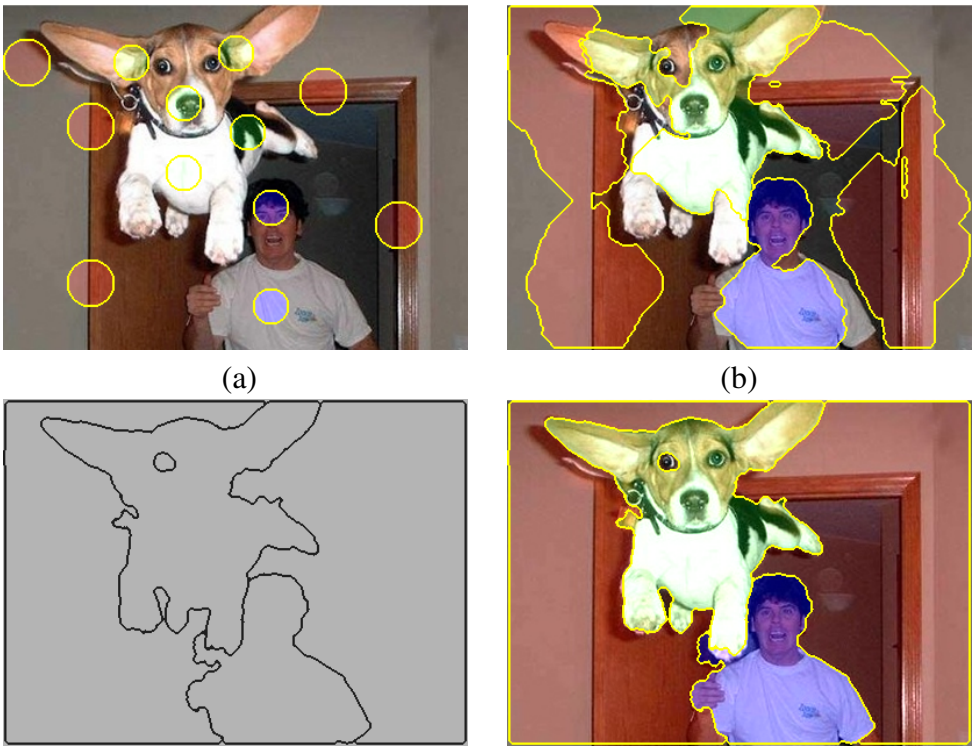


Figure 5.9.: Segmentation of an image of a koala. Top: Segmentation with the proposed method. Left to right: Initial contours at the coarsest of 3 scales; an intermediary step showing leakage; the final segmentation shown as contour mask; the final contour at the finest scale, superimposed on the image. Bottom: Segmentation with the region-competition approach Zhu and Yuille (1996) starting with the same initialization as for the top segmentation. Left to right: Segmentation with the color model from Zhu and Yuille (1996) superimposed on the image and shown as regions mask; segmentation with RGB color superimposed on the image and as regions mask.

curate segmentation results were obtained, in spite of the fact that initial contours were placed far away from object boundaries. With other methods based on local information and spatially-invariant definition of energy terms, initial contours must generally be placed in the neighborhood of the object to be segmented in order to obtain good results.

Our accurate segmentations starting from ambitious initializations are due to multiple contours with spatially-variant definition of energy terms, and thus of forces. The type of force is chosen depending on the presence or absence of other contours in the vicinity of a contour point. If there are no other contours near the point the contour piece is probably away from an object boundary, and the contour must be made to grow. For this purpose, it it more appropriate to assume that features are Gaussian distributed in local regions. If there are other contours in the local vicinity of a contour point, the contours compete. We have shown that, in this situation, it is more appropriate to assume that the data is better represented with a kernel density estimation. The



(a) (b)

(c) (d)

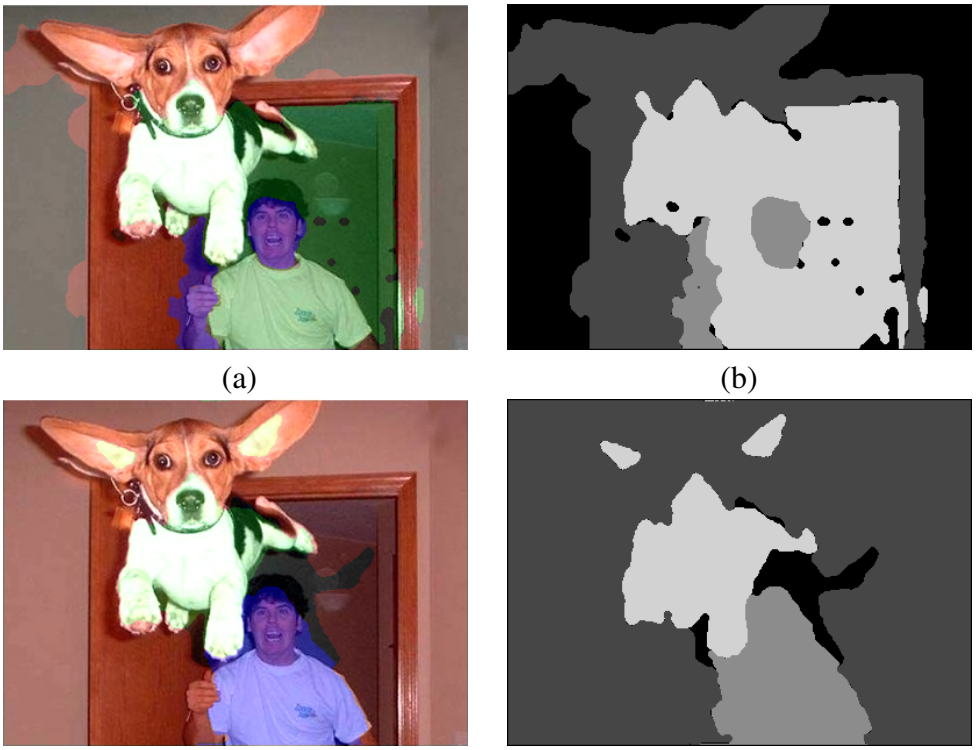
Figure 5.10.: Segmentation of an image of a flying dog using the proposed method. a) Initial contours; b) an intermediary step showing leakage. The final segmentation shown as (c) contour mask and (d) superimposed on the image.

result is a robust active contour that takes advantage of the accuracy of local region descriptors, while substantially reducing sensitivity to initialization.

Our new method involves five main parameters. For four of them, we were able to empirically determine generic values, while the window size for the region competition term still had to be selected by the user for a particular image. This is clearly a point that needs improvement. Determining window sizes for the growing and competition term that automatically adapt to the image will certainly not be a trivial task.

Energies based on local descriptors can accurately describe image data at the cost of having more local minima. Energies based on global descriptors have much less minima. Active contours with spatially-variant definition of energy terms could be used to combine the advantages of global and local approaches. For example, global descriptors could be used in criterion-regions where the contour has reached an equilibrium under the action of local descriptor forces. The stopping condition proposed in Section 5.4 could be used to determine if a contour piece has stopped.

Finally, there exist a number of state-of-the-art methods where initial contours and the number of



(a) (b)

(c) (d)

Figure 5.11.: Segmentation of an image of a flying dog using the region-competition approach Zhu and Yuille (1996) starting with the initialization in Fig. 5.10a). a) Segmentation with the color model from Zhu and Yuille (1996) superimposed on the image and b) shown as regions mask. c) Segmentation with RGB color superimposed on the image and d) as regions mask.

regions is detected automatically. Our method should also be extended to a fully automatic version. However, semi-automatic initialization has the advantage that it benefits from user knowledge, allowing a user to obtain the desired segmentation.

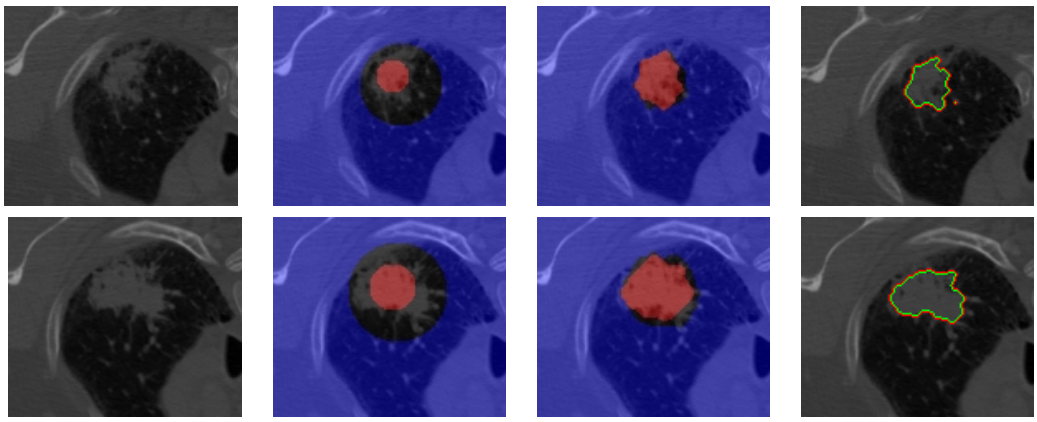


Figure 5.12.: Details of two CT slices of a lung tumor. Left to right: the original images, initial contours, an intermediary step and the final segmentation.

6. Local Region Descriptors for Texture Segmentation

> The most exciting phrase to hear in science, [...] is not "Eureka!" (I found it!) but "That's funny ..."
>
> *Isaac Asimov*

This chapter uses active contours with space-variant definitions of energy terms for the segmentation of complex natural images. The local region descriptors employed to drive the contours formalize the variation of colored texture. For this purpose, image patches are viewed as texture exemplars. We show that the frameworks proposed in the previous two chapters are flexible and allow us to tailor the segmentation solution to the nature of the problem.

The linearized image patches constitute texture feature vectors which exist as points in a high-dimensional space, where probability density estimation is problematic. We model densities with a variable kernel estimator. Finding appropriate kernel window widths for estimating the density at each data point is a difficult statistical problem. We propose a solution that involves finding the k-nearest neighbors of a feature vector in a set of vector-observations selected from the image vicinity of the pixel characterized by the feature vector. The value of the pdf at a feature vector is evaluated in the local region descriptors framework, which means that the observations needed in the evaluation are obtained from a local window centered on the pixel characterized by the texture-feature vector.

The texture-features and their pdfs evaluated locally are complex LRDs able to model textured regions in natural images. They can provide good descriptions of regions due to their local nature. The problem of numerous local minima is handled by active contours with energy term definitions based on two density models. For the term based on Gaussian model, designed to have a growing behaviour,we design a novel one-dimensional feature by computing a mean distance from other features. This mean-distance feature (md-feature) appears to have an approximately normal distribution, and it proves to yield a reliable growing term that finds many correct boundaries. For the term based on kernel density estimation, designed to have a competing behaviour, the pdf for local descriptors is approximated with a variable kernel estimator (Silverman, 1986). The sizes of kernel window widths are determined from the image data with a novel method that

involves computing k-nearest neighbors in a local image patch.

The proposed natural image segmentation method is novel, in that it combines recent ideas on density estimation of high-dimensional image patches for texture modeling with the new spatially-variant active contours driven by local information. In our experiments, this method leads to very good segmentation results of natural, large, and complex scenes. Additionally, it is two orders of magnitude faster than a state-of-the-art contour based on global probability densities of image patches as texture features, and yields comparable results.

Before presenting the proposed method, we set the stage by shortly discussing texture as visual cue, and reviewing some works concerned with texture segmentation.

6.1. Color and Texture in Image Segmentation

Color and texture provide rich visual cues that help us make sense of the world around us. For example, the texture of the picture in Fig. 6.1(a) gives away that it is an aerial picture. New objects can be created by change in texture alone, as shown in the details (b) and (c) in Fig. 6.1[1].

From a technical point of view, the question is how the color and texture of visible objects can be represented by measurable quantities, processable with computer algorithms; in the fields of image processing and computer vision these quantities are usually named features.

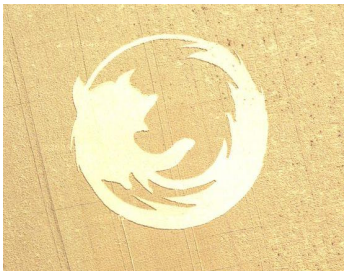
Measurable quantities for color are obtained by choosing a color space and digitizing the image in this color space. As a result, each image pixel is characterized by a 3-dimensional feature vector, since most color spaces are 3-dimensional. Images are often represented in the RGB color space since this is the one used by sensors; a color is obtained by mixing red, green and blue, each with values between 0 and 255.

A disadvantage of RGB is that to obtain a light red from a dark red one needs to increase the values in *all* RGB channels, thus ending up with a very different set of values. However, for the purpose of our algorithms, we would expect small changes in values for a small change in color. This has been achieved with the CIE Lab color space as derived by the International Commission on Illumination. In this color space, the intensity (or monochromatic) component of the image (L) is separated from its chromaticity component (ab). The L channel is nothing but the gray version of the color image. In this chapter, images will be represented in the CIE Lab color space for segmentation purposes.

Intensity and color modeled by parametric distributions are well known region descriptors in

[1]The Firefox browser logo was created in the fields of Oregon by members of the Oregon State University, who wanted to suggest the browser was preferred by extraterrestrials

(a)



(b)

(c)

Figure 6.1.: (a)A place in Oregon, USA. The textures in the image strongly suggest this is a aerial image (Firefox, 2008). (b) and (c) Details of (a); the shape of a fox can be perceived because there is a change in the texture of the field.

active-contours segmentation, e.g. (Jehan-Besson and Barlaud, 2003; Chan and Vese, 1999, 2001). Although color features with parametric distributions can yield very good segmentation results, their performance degrades rapidly as the distribution of the image data is increasingly different from a normal distribution. In fact the PDFs of feature vectors may have arbitrary shapes that can be better modeled by non-parametric methods for density estimation when appropriate parameters to match the data are chosen.

Non-parametric methods were introduced into active contours after parametric ones. Many use (smoothed) histograms (Brox and Weickert, 2004a; Kadir and Brady., 2003; Chan et al., 2007) and Parzen windows (Kim et al., 2005) since these are simple and fast to compute. Histograms can represent well feature vector with up to three components, provided that one can find the appropriate bins sizes and positions. However, it is most often not feasible to create the histogram when many more features are involved, as usually the case with texture. But before addressing the issue of statistical modeling for texture features we need to specify what the quantities are that characterize texture.

6.1.1. Texture representation

Texture is a rich source of information, but obtaining descriptive quantities for it is not as straight forward as for color, since it arises from more than one pixels. Characterizing the visual properties of texture involves computation of texture features. Texture features mean to capture the orientation and scale of structures in the texture. The features are very often obtained filtering the image with a filter bank, i.e. a set of filters, one for each pair orientation-scale. Widely used filters are Gabor filters shown in Fig. 6.2 (Sagiv et al., 2002; Dunn and Higgins, 1995; Jain and Farrokhnia, 1990; Sandberg et al., 2002) and wavelet filter banks (Unser, 1995). Filter banks have a biological motivation. Simple cortical cells of a biological visual system seem to implement Gabor filters (Marcelja, 1980; Jones and Palmer, 1987). Each cell is responsible for responding only to the presence of structures of a particular orientation at a particular scale in the visual field.

An image of two Brodatz textures, for grass and for straw, is shown in Fig.6.3(a). The orientation of textures is different: while the grass texture is not oriented, the straw texture in the middle has a unique orientation. There is also a difference in texture scale, the granularity of the straw texture being larger. The discretization of the exemplary Gabor function shown in Fig.6.2 has been employed to filter the image of Brodatz textures 6.3(a). The function was oriented at 0, 45 and 90 degrees and the respective results of filtering operations are displayed in Fig. 6.3(b)-(d). In these images we observe that the grass texture has large responses for each of the orientations, while the straw texture most strongly responds to the 45 degree-oriented filter in Fig.6.3(c).

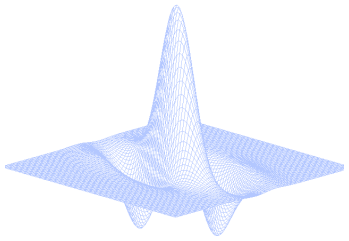
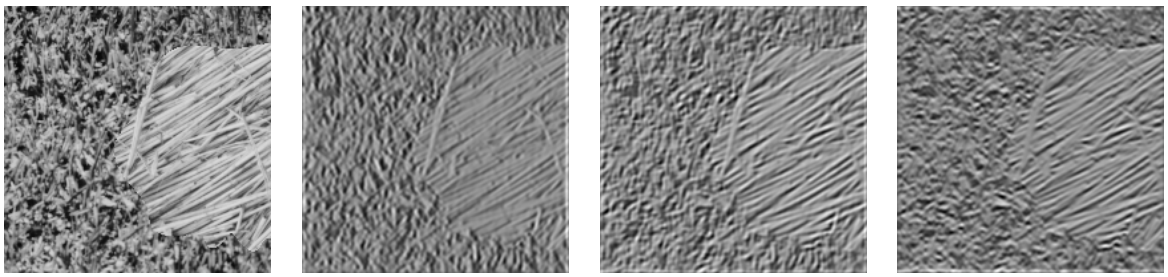


Figure 6.2.: An exemplary Gabor function.



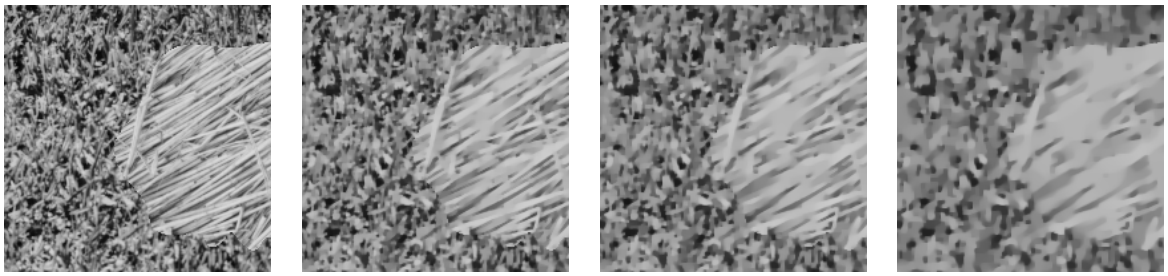
(a) (b) (c) (d)

Figure 6.3.: (a) An image showing the Brodatz textures for grass and straw. Results of filtering the image with the Gabor filter in Fig.6.2 oriented at (b) 0, (c) 45, and (d) 90 degrees.

Gabor filters can offer a very good representation of a texture if the number of employed filters is large enough, but, this representation is most often redundant. Computing an image's segmentation with the resulting features can thus be computationally very expensive.

A representation that involves a smaller number of features is the structure tensor (Bigün et al., 1991). For each scale the structure tensor yields three features, namely the image's second order derivatives (Rousson et al., 2003; Brox et al., 2003a). Before computing the structure tensor, images are often smoothed with Gaussian kernel in order to remove noise. This operation may however blur the boundaries. The problem may be avoided with the non-linear structure tensor that involves replacing Gaussian smoothing with an anisotropic diffusion process (Rousson et al., 2003). During this process texture features undergo coupled smoothing, i.e. the value of a feature is updated in each diffusion step by an amount that depends on the values of all other features. With help of the non-linear structure tensor, a colored texture can be described by a vector $(R,G,B,R_x^2+G_x^2+B_x^2,R_y^2+G_y^2+B_y^2,R_{xy}+G_{xy}+B_{xy})$, where R,G,B denote the color channels, and the subscripts denote partial derivatives (Brox et al., 2003a).

Joint smoothing of vector components by diffusion results in a version of the original image where edges are preserved while color differences in texture elements can be reduced. The requirement is that the variability of features within a region is sufficiently small compared to the variability of features between regions, since edges of the same magnitude will be either smoothed or preserved independent of their position (within a texture or at a boundary). For example, in Fig.6.1.1 many gradients in each region of the two-Brodatz textures image are progressively reduced by the diffusion operation. The figure shows the results of diffusion after 25, 50 and 100 iterations. However, many strong gradients still remain in each region, even after a large number of iterations. Another example is the color image in Fig.6.1.1, where most gradients caused by texture are strongly reduced. This however causes some parts of the image-subject and the gravel foreground to become very similar in color and texture. In fact much of the texture information useful in discriminating between the two regions is lost. This effect can be observed even better by looking at the gray version of this image in Fig.6.1.1(d).



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

Figure 6.4.: (a) An image showing the Brodatz textures for grass and straw and the results of non-linear diffusion on the image. Diffusion depends on the values of the structure tensor. Results after (b) 25, (c) 50, and (d) 100 diffusion iterations.

Quantitative features for characterizing texture obtained with a filter bank or using the structure tensor involve a not negligible amount of computation. Image patch exemplars are an alternative texture representation, which does not require the computation of any features. We dedicate the next section to discussing this alternative, since this is the texture representation used in our

(a) (b)

Figure 6.5.: (a) A color image and (b) the result of non-linear diffusion on the image. Diffusion depends on the values of the structure tensor.

method.

6.1.2. Texture from image patches

An image patch is simply a neighborhood (also window) of specified shape and size around a given pixel. Pixels in this patch are not independent and identically distributed (iid), as methods often assume, but their spatial positions influence the distributions of their values. In other words, pixels in a patch are conditioned random variables, since texture exhibits regularities.

Julesz was the first to suggest that these relations can be modeled by the joint probability distribution of N neighbored pixels, which he denoted as higher-order statistics (Julesz, 1962). In this work we will more often use the terms joint PD and multivariate PD. An image can then be modeled as a Markov Random Field. The relationships between pixels can be described by specifying the probability at a pixel conditioned on the probabilities at pixels in its neighborhood. MRFs have been long used in image processing for the purpose of image filtering (Geman and Geman, 1984; Bouman and Sauer, 1993).

Later, MRF modeling has been used in the field of texture synthesis (Efros and Leung, 1999; Liang et al., 2001; Zalesny and Gool, 2000). The model is successful because it can quickly generate textures with good visual qualities, similar to those of an input texture. For example, in (Efros and Leung, 1999) the conditional PD is approximated from the input texture by selecting similar patches from the texture and then learning the value of the center pixel with a histogram. To synthesize a patch, a 3×3 seed selected for the input texture at random is grown in layers outward by sampling from the learned conditional PD.

The works in (Popat and Picard, 1997; de Bonet and Viola, 1998) is among the first to use MRF modeling for neighborhoods up to 13 pixels large for texture classification. Both works demonstrated very good texture classification results sparking the interest in patch-based meth-

ods, which were recently proven to be superior to filter-based approaches, when combined with a good probability density estimation method (Varma and Zisserman, 2003, 2008). These latter works have also shown that conditioned probability distribution in MRFs can be replaced by joint PDs as proposed by Julesz (Julesz, 1962), without affecting the results.

A fundamental issue associated with texture represented as image patch exemplars is the estimation of their probability densities in the high-dimensional spaces they exist in. The references cited here empirically showed that this estimation is feasible such that pdfs can be used for distinguishing between classes of textures. For this reason, but only very recently, image patch modeling has been employed in texture segmentation.

In the work of Wolf et al. (2006), texture edges are detected by computing similarities between image patches around a given pixel. With the help of a non-parametric statistics test (the Wilcoxon Mann-Whitney test), it can be decided if the computed similarities are drawn from the same distribution; if they are not, the pixel is labeled as edge. Segmentation is achieved using a free-form-deformation method on the distance transform of the edge image.

The work of (Awate et al., 2006) presents an active-contours method driven by global descriptors based on image patches, and is thus related to the method presented here. We are not aware of any other related methods. The method (Awate et al., 2006) uses very large image patches of size 13×13px for segmenting monochrome images. The pdf at an image patch is estimated with Parzen windows using an isotropic Gaussian kernel with constant window width. The contour's motion is determined with the shape derivative tool thus having more terms than a motion determined in an expectation maximization style, as explained in Section 3.2.

Our active-contours method for texture segmentation is based on local region descriptors and their probability densities are approximated with variable kernel density estimators. The EM-style equation of motion is implemented with contours with spatially-variant definitions of energy terms. As a result of combining local processing with the novel type of spatially-variant contours, our active contours are two orders of magnitude faster than those presented in (Awate et al., 2006).

The problem of estimating probability densities of high-dimensional feature vectors

Methods based on image patches have to deal with high-dimensional feature vectors (their dimension is equal to the number of pixels in the patch). For this reason, all methods mentioned above resort to non-parametric density estimators. These estimators can be very powerful as shown in the seminal work of (Viola, 1995); however, the author warns that estimating PDs in high-dimensional spaces might not be feasible because sparsity introduces distortions. "Empirical evidence argues against using Parzen estimation in many more than six dimensions."

Before we continue our discussion, it is important to observe that the nomenclature commonly used in the literature does not describe exactly what is estimated. Most authors state that PD of feature vectors are approximated with non-parametric methods. If this statement is to be taken literally, we should expect to know (be able to graph) the shape of the PDF on its definition domain. However, to the best of our knowledge, no one has claimed to be able to provide such a result. At best, results provide the number of clusters in the data and some way to describe their shape and size.[2] More often, the evaluated quantity is the value of the modeled probability distribution function at a given data point and our method is no exception. For the sake of conciseness, we will denote this quantity with density at a point and, when appropriate, point density estimation.

Returning to patch-based methods, those mentioned previously work with feature vectors with a number of components between 9 and 169 which exist in high-dimensional spaces. The problem of analyzing high-dimensional data is a notoriously difficult one and it is known as the curse of dimensionality (Scott, 1992). Contrary to the statement in (Viola, 1995), there is empirical evidence that multivariate density estimation in higher-dimensional spaces is feasible and meaningful, because, in general, the underlying structure of the data has a lower dimension than the data itself (Scott, 1992). Empirical evidence (Lee et al., 2003; Georgescu et al., 2003) supports the idea that image features cluster in the high-dimensional space.

After all, the methods cited in this section do present good results, providing more evidence that point density estimation in high-dimensional spaces is possible. An explanation could originate in the nature of texture itself: pixels follow specific spatial relationships thus yielding patches that look alike and should thus be close to each other in the high-dimensional space. To have patches entirely scattered over the data space the image would have to be the results of experimenters trying to absolutely confuse the human visual system, rather than a picture of something in the world. It is also very important to keep in mind that our goal is to discriminate between image regions by using pdfs of features over regions. The bias of the estimation is thus a matter of secondary importance, as long as separability is ensured. As such, good classification results can be obtained even if density estimates are quite crude (Silverman, 1986), pp.124. Even so, density estimation involves carefully choosing the estimation method and its smoothing parameters.

Quite a number of different non-parametric methods have already been applied to estimate statistics of image features. Clusters of points were determined by k-means clustering in (Popat and Picard, 1997; Varma and Zisserman, 2003). In (Popat and Picard, 1997), the user is involved in choosing the number of clusters and after determining the points of each cluster, a multivariate Gaussian with diagonal covariance matrix is fitted to each cluster.

A widely used strategy for estimating densities from high-dimensional data points is to estimate a pilot density using a k-nearest neighbors method, and subsequently choose parameter(s) to

[2] In this work, we do not address methods that have the goal of determining surfaces that separate clusters, like Linear Discriminant Analysis or Support Vector Machine methods. In general, these are methods for supervised learning, while this dissertation deals only with unsupervised segmentation.

determine variable window widths for an adaptive kernel estimate Silverman (1986). For up to three features, it has been shown that virtually all parameters can be automatically determined from the data (Comaniciu et al., 2001; Han et al., 2008).

For much higher dimensional feature vectors, the strategy was adopted in a heuristic data partitioning algorithm for finding clusters in high-dimensional spaces (Georgescu et al., 2003). In this algorithm, modes of the pdf were found with the iterative mean-shift procedure from (Comaniciu, 2003). To evaluate the densities at points involved in the mean-shift procedure, a variable kernel density estimator was employed. The kernel window width at each point was determined by taking nearest neighbors in the same partition with the point.

The k-nearest neighbors of vectors obtained from image patches are used to estimate densities for the purpose of texture synthesis in (Efros and Leung, 1999; Liang et al., 2001). Finding the k-nearest neighbors of a data point can be a very expensive computational step, but many algorithms have been designed in order to optimize the efficiency of the procedure, e.g. the one in (Liang et al., 2001).

Parzen-windows with fixed window width were employed in (Awate et al., 2006) for texture segmentation from image patches. The fixed window width is determined from the data via entropy minimization as follows: the user selects a the number of observations to participate in kernel entropy estimation. Observations are then selected from the image for entropy evaluation. To find the optimal window width, entropy is minimized by gradient descent with respect to this parameter. This procedure is repeated every few iterations. This method works well when the evaluated entropy function depending on the window width has a unique minimum. But of course, for many images this is not the case. When graphing the entropy for some of the images segmented in this thesis, the shape of the function was often asymptotically decreasing or it was widely flat near the minimum.

The non-parametric Wilcoxon-Mann-Whitney statistical test was used to decide if two patches belong to the same textured region in (Wolf et al., 2006). The very promising results suggest that this avenue of research is to be looked at more thoroughly. In our research, we did not find other works on segmentation based on non-parametric statistical test.

One may observe that segmentation methods based on texture from image patches involve important decisions regarding patch size, the pdf estimator, its parameters and the selection of data points for evaluation. With this observation in mind, in the following we will construct novel local region descriptors that involve image patches. Unlike any of the methods presented so far, that use image patches only for gray images, the novel descriptors use color texture patches.

6.2. LRDs from Image Patches and Non-parametric Statistics for Texture Segmentation

We wish to propose a solution to the texture segmentation problem that relies on active contours with spatially-variant definitions of energy terms and local region descriptors. Remember that a local region descriptor is defined by choosing those features that best quantify region characteristics, and by choosing the pdf model to approximate the features' variation in a region. This we will do in the following. First, to describe image textures, we will introduce image patches as features for characterizing texture. Second, we will introduce a solution for the difficult problem of estimating the local probability densities with variable kernel density estimation.

6.2.1. Image Patches as Color Texture Features

Intuitively, texture can be often viewed as the repetition of an arrangement of pixels on a patch with a certain size. Every time it is repeated, the arrangement is varied to a certain degree in color and in position, but when its pixels are examined together, a common pattern is perceived. Pixels in the patch thus determine each other's intensities or colors. In statistical terms this means that the probability of observing a given texture patch is equal to probability of jointly observing the pixels in the patch.

Formally, the textured region is modeled as a random field: a random variable is associated with every image pixel $\mathbf{x}$ and a neighborhood $N_s(\mathbf{x})$. A patch is defined as the random vector containing the random variables associated with $\mathbf{x}$ and with all pixels $s \in N_s(x)$.

Most often, the neighborhood $N_s(\mathbf{x})$ is considered to be a $D \times D$ square centered on $\mathbf{x}$. For a gray scale image, the image patch centered at pixel $\mathbf{x}$ is composed of random variables f_s, $s = 1..d$, where $s \in N_s(x)$, and $d = D^2$ is the number of elements in each pixel's neighborhood. This patch can be represented as the d-dimensional random vector of features $\mathbf{f} = (f_1, ..., f_d)$. The notation $\mathbf{f}(\mathbf{x})$ will be used when we wish to emphasize the realization of $\mathbf{f}$ at pixel $\mathbf{x}$. With this model, for a 5×5pixel square image patch, $\mathbf{f}$ will have 25 components.

For color images, an image patch is defined as the joint realization of patches in each image channel. A square patch centered at pixel $\mathbf{x}$ yields a random vector
$\mathbf{f}(\mathbf{x}) = (f_{L,1}, ..., f_{L,d_L}, f_{a,1}, ..., f_{a,d_a}, f_{b,1}, ..., f_{b,d_b}) \in R^d$. This notation is motivated by the fact that we use the CIE Lab color space in our experiments. One may allow the sizes of patches to differ between channels, if one channel is to be given more importance than others. For example, in the CIE Lab color space, the intensity channel carries more information than the chromaticity channel. The size d_L of the patch in the intensity channel can thus be two times the size $d_a = d_b$ of a patch in a chromaticity channel, in order to give the same importance to intensity and chromatic information. When there is reason to assume more complex relationships between channels,

vector components from each channel can be assigned weights, or equivalently, one can set an a priori probability for each channel.

So far, a feature vector was obtained that can be employed to describe the texture of a region. A pdf estimator is further needed in order to obtain a complete region descriptor.

6.2.2. Variable Window-Widths from Local Nearest-Neighbors for KDE

We have already discussed that kernel density estimation is a good method for approximating the pdf at a feature vector $\mathbf{f}(\mathbf{x})$ and that these data points in the high-dimensional space tend to cluster in its lower-dimensional subspaces.

In practice, image patches are very likely to cluster around more modes than one for each region. Clusters will probably have different shapes and sizes depending on the size of a patch and on the nature of the texture. For example, in the image of a checker-board, some image patches will start with white pixels, others will start with black pixels, some will contain a line, others will contain a crossing. If patches are large compared to the scale of the board, patches will contain multiple crossings. The pdf of the corresponding feature vector will certainly have multiple modes. Additionally, the basins of attraction for modes are likely to have different sizes and they may overlap.

It is possible to assume that a constant kernel width models the data well, with the benefit of reduced algorithm complexity. This is indicated by the results presented in (Awate et al., 2006). It is however improbable that all natural textures should accommodate this assumption just in order to not let algorithms grow too complex. In order to accommodate a larger class of images we propose to use variable kernel density estimation for texture segmentation.

Variable kernel estimators can represent the data better than kernel estimators by adjusting bandwidths h_s to data samples used for estimation. We remind here that given n data samples $\mathbf{f}(\mathbf{x}_s) \in R^d$ from a region R_i and a symmetric kernel function K, the variable kernel estimator $\hat{p}(\mathbf{f}(\mathbf{x}))$ is (Silverman, 1986)

$$\hat{p}_i(\mathbf{f}(\mathbf{x})) = \frac{1}{n}\sum_{s=1}^{n}\frac{1}{h_s^d}K\left(\frac{\mathbf{f}(\mathbf{x})-\mathbf{f}(\mathbf{x}_s)}{h_s}\right). \tag{6.1}$$

The most common kernel functions are the Epanechnikov kernel

$$K(u) = \frac{3}{4}(1-u^2)\mathbf{1}_{(|u|<1)}, \tag{6.2}$$

and the Gaussian kernel

$$K(u) = 1/\sqrt{2\pi}\exp(-\frac{1}{2}u^2)\mathbf{1}_{(|u|<1)}. \tag{6.3}$$

The Epanechnikov kernel is more efficient to compute. In our experiments, we have tested both kernels and obtained similar results. Therefore, later on we will present result obtained with the Epanechnikov kernel.

There are numerous methods for computing the widths h_s. A widely used strategy is to partition the data into clusters and then determine the size of clusters by some means. Many algorithms start the clustering by finding the k-nearest neighbors (NN) of each point (Georgescu et al., 2003; Popat and Picard, 1997). For large data sets, the resulting procedure is computationally expensive, in spite important advances concerning kNN search (Georgescu et al., 2003; Liang et al., 2001).

In our method, we also use nearest neighbors. Observations are selected from local regions for evaluating a point density, as well as for determining the kernel window width for each point, for reasons that will be described shortly. We concentrate for a moment on the vicinity of a pixel $\mathbf{x}_s$, denoted by V. The k nearest neighbors of $\mathbf{f}(\mathbf{x}_s)$, shortly $\mathbf{f}_s$, are sought for in the set of feature vectors observed at pixels that lie in the vicinity V. Let f_{kNN} be the k^{th}NN of the data point f_s in the set of observations generated from the vicinity V. The size of the kernel window for the data point $\mathbf{f}_s$), h_s, is the L_1 distance between the data point and its k^{th} nearest neighbor

$$h_s = \|\mathbf{f}_s - \mathbf{f}_{kNN}\|_{L_1}. \tag{6.4}$$

The L_1 distance was chosen because it is fast to compute and the difference between two d-dimensional vectors in (B.4) needs to be computed very often.

The question now is how to determine k. The number of pixels in the vicinity of pixel $\mathbf{x}_s$ is determined by the shape and size of V. We can thus consider that the number of modes in the local density is limited. To find a good value for k three additional assumptions are made. First, we suppose that clusters around these modes are relatively compact and separable to a sufficient degree. Then we can state that there is a minimal number of feature vectors k_{min} in each cluster. Second, we presume that there is a maximal limit to the number of feature vectors needed to estimate the bandwidth for the current patch, and denote it as k_{max}. Third, we require that the distance between two neighbored feature vectors be no larger than a maximum distance, which is to say that we set a maximum bound on the kernel window width, denoted as h_{max}.

A good value k for the k^{th} nearest neighbor is found between k_{min} and k_{max} by taking the largest k, such that the window width h_s satisfies $h_s \leq h_{max}$. This means that we take the maximum value k such that $k \in [k_{min}, k_{max}]$ and the L_1 distance between current feature vector and its k^{th}NN is smaller than h_{max}, $\left\|\mathbf{f}_j - \mathbf{f}_{kNN}\right\|_{L_1} \leq h_{max}$. Finally, the limits k_{min} and k_{max} are fractions of the total number of observation in the vicinity V.

In our implementation, we use a square vicinity and set the limit k_{min} to 1/10 and the limit k_{max}

is 1/2 of the number of pixels in the local window. The interval $[k_{min}, k_{max}]$ is discretized to a set by taking fractions {1/10, 1/9, .. 1/2} of the number of pixels in the window.

Local sampling strategy. A widely used assumption is that textured image regions are stationary random fields. However, like in the case of intensity and color, textures often do not appear constant over the surface of the object, but they undergo smooth changes. We are thus motivated to adopt a local sampling strategy in order to determine the kernel window width as just described.

Let us again concentrate on the vicinity V of a pixel $\mathbf{x}_s$; an exemplary window $V(\mathbf{x}_s)$ is depicted in Fig. 6.6 in light gray. According to the previous paragraph, patches most similar to the one centered on pixel $\mathbf{x}_s$ are those patches centered on pixels that lie in the vicinity $V(\mathbf{x}_s)$. This suggests that the kernel window width h_s for the feature vector $\mathbf{f}(\mathbf{x}_s)$, centered at $\mathbf{x}_s$, can be found by searching for nearest neighbors in the set of feature vectors observed at pixels within window $V(\mathbf{x}_s)$. By the same reasoning, the pdf at a point $\mathbf{f}(\mathbf{x})$ is also estimated from local observations in the vicinity of a contour pixel $\mathbf{x}$, denoted as $W_2(\mathbf{x})$, rather then from the whole image; an exemplary window $W_2(\mathbf{x})$ is drawn in black in Fig. 6.6.

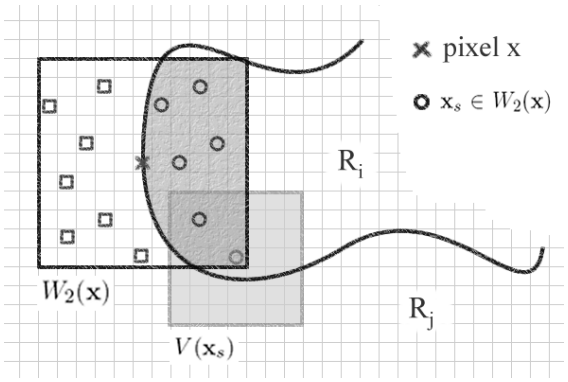


Figure 6.6.: Windows for local sampling strategy for selecting observation to compute variable kernel window width and estimate the pdf at a feature vector centered on pixel $\mathbf{x}$. Window V is for computing variable kernel window width. Window W_2 is for computing the pdf estimate and the competition forces.

The local sampling strategy reduces computation time considerably since distances between feature vectors are computed in small local windows, in stead of the whole image. For each window, we compute and memorize in an initial step the distance between the center patch $\mathbf{f}(\mathbf{x})$ and every other patch $\mathbf{f}(\mathbf{y})$ centered on a pixel in the window $\mathbf{y} \in W_2(\mathbf{x})$. These distances are used initially to compute kernel window widths, and they are reused in every iteration of the level-set evolution to estimate densities of local descriptors. The procedure for computing kernel window widths and local sampling strategy can be summarized as follows

- Select shape and size of V and W_2; compute the number of pixels N_V and N_{W_2}

- Initialize the distances structure $D(\mathbf{x},\mathbf{y})$: for each $\mathbf{x}$ *inR*
  - For each pixel $\mathbf{y} \in W_2(\mathbf{x})$ compute $D(\mathbf{x},\mathbf{y}) = \|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})\|_{L_1}$.
- Compute kernel window widths.
  - Set vector $k = \left\{ \frac{1}{10}N_V, \frac{1}{9}N_V, ..., \frac{1}{2}N_V \right\}$ and h_{max}.
  - For each $\mathbf{x} \in R$
    - Sort distances $D(\mathbf{x},\mathbf{y})$, $\mathbf{y} \in V(\mathbf{x})$ into D_{temp}. Set number of neighbors for $\mathbf{f}(\mathbf{x})$ $nn = 0$.
    - Count nn and find largest $k(i)$ such that $nn > k(i)$, and $D_{temp}(\mathbf{x},\mathbf{y}_{k(i)}) < h_{max})$. Set $h(\mathbf{x}) = D_{temp}(\mathbf{x},\mathbf{y}_{k(i)})$.
- { When computing the competition term for level-set evolution... }
- Estimate the pdf for each $\mathbf{x}$ on the zero level set, according to equation 6.1 by using all $\mathbf{y} \in W_2$.

The descriptors thus obtained are local texture descriptors. They will be used to drive active contours with piecewise constant behavior in order to segment textured images.

6.3. Contours with Spatially-Variant Definition Based on LRDs from Image Patches

Region descriptors allow us to estimate region probability densities, and based on that generate forces to evolve initial contours. Conventional active contours based on local descriptors often get trapped in local minima, because the local nature of the information causes the descriptor to be "short-sighted". For the same reason, segmentations obtained with these contours are dependent on the initial positions of contours. In chapter 5, we have proposed contours with piecewise constant behavior that are driven by local region descriptors and can be initialized far away from real boundaries.

We remind the reader of the multiple contours driven by two types of forces, one of which is a growing force that assumed locally Gaussian regions, and the other was a competing force for which pdfs of competing regions were kernel density estimated. The growing force was applied at a contour point if the contour was not neighbored in a local window W by other contours, which meant that the characteristic function $\mathbf{1}_i^W$ of the set $W \cap_{j=1,j\neq i}^{m} R_j = \emptyset$ had value 1 at $\mathbf{x}$. If at least another contour was present in the local region around the contour point, the competing

force was applied.

The growing force requires that the feature in the region descriptor can be approximated by a normal distribution. However, feature vectors representing image patches exist in high-dimensional spaces, and, in practice, it is very unlikely to find high-dimensional data distributed according to a multivariate normal law. More likely, these densities are very complex and, mostly, they are only crudely estimated, as discussed in section 6.1.2. The analysis of high-dimensional data is often approached with a dimension reduction method. In the spirit of this idea, we design a novel one-dimensional feature for the growing term by averaging over local image patches and name it mean-distance feature.

Mean-distance features (Md-features). The novel one-dimensional md-feature, denoted as $fd(\mathbf{x})$, is computed for every pixel in a local window by manipulating distances between patches.

Fig. 6.7 shows local windows and pixels involved in computing an md-feature. Consider a pixel $\mathbf{x}$ on an evolving contour and a local window $W_1(\mathbf{x})$ centered on pixel $\mathbf{x}$. An md-feature is computed for each pixel $\mathbf{x}_s \in W_1(\mathbf{x}) \cap R_i$. This is done using another local window $W_1(\mathbf{x}_s)$ centered on the pixel $\mathbf{x}_s$ in order to select some other pixels $\mathbf{x}_t$ that belong to the same region as pixel $\mathbf{x}_s$, i.e. $\mathbf{x}_t \in W_1(\mathbf{x}_s) \cap R_i$. The L_1 distances between the feature vector $\mathbf{f}(\mathbf{x}_s)$ centered on pixel $\mathbf{x}_s$ and each feature vector $\mathbf{f}(\mathbf{x}_t)$ centered on a pixel $\mathbf{x}_t$ are computed and denoted with $D_t(\mathbf{f}(\mathbf{x}_s), \mathbf{f}(\mathbf{x}_t))$. The md-feature for pixel $\mathbf{x}_s$, denoted as $md(\mathbf{x}_s)$, is the mean value of distances $D_t(\mathbf{f}(\mathbf{x}_s), \mathbf{f}(\mathbf{x}_t))$

$$md(\mathbf{x}_s) = \frac{\sum_{\mathbf{x}_t \in W_1(\mathbf{x}_s) \cap R_i} D_t(\mathbf{f}(\mathbf{x}_s), \mathbf{f}(\mathbf{x}_t))}{|W_1(\mathbf{x}_s) \cap R_i|}. \tag{6.5}$$

The computation can be made more efficient if not all pixels $W_1(\mathbf{x}_s) \cap R_i$ participate in determining the mean; for instance, one can randomly select a fixed number of pixels from this region, or take every second pixel.

We assume that the md-feature is a random variable that is normally distributed in each local window $W_1(\mathbf{x})$. At any pixel $\mathbf{x}$ we can compute the mean m and standard deviation s of this random variable in the local window $W_1(\mathbf{x})$ from md-features of pixels $\mathbf{x}_s$

$$m = \frac{\sum_{\mathbf{x}_s \in W_1(\mathbf{x}) \cap R_i} md(\mathbf{x}_s)}{|W_1(\mathbf{x}) \cap R_i|.} \tag{6.6}$$

The growing term. The growing force in equations (5.5) and (5.13) was the maximum between a balloon force term and a similarity term based on properties of the normal distribution. When using md-features in our experiments, we have found that many boundary pieces are correctly detected even when computing the similarity term only.

The growing force for a pixel $\mathbf{x}$ is thus written using the similarity measure g^{W_1} that depends on

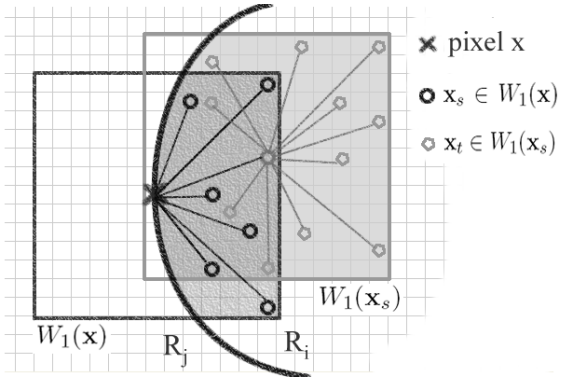


Figure 6.7.: Local windows and pixels involved in computing md-features. $\mathbf{x}$ is the pixel for which the growing force is to be computed. The local window $W_1(\mathbf{x})$ is centered on $\mathbf{x}$. $\mathbf{x}_s$ are pixels in $W_1(\mathbf{x}) \cap R_i$. For each pixel $\mathbf{x}_s$, an md-feature is computed by averaging the distances from the feature vector centered at $\mathbf{x}_s$ to each feature vector centered at a pixel $\mathbf{x}_t \in W_1(\mathbf{x}_s) \cap R_i$

the md-feature of the pixel $\mathbf{x}$ and the mean and variance of this feature in the local window

$$g^{W_1}(\mathbf{x}) = 2s(\mathbf{x}) - |md(\mathbf{x}) - m|. \tag{6.7}$$

This similarity measure is analogous to the one given for intensity features in Section 4.2.3: the larger the difference between the local mean of the md-feature and the md-feature of $\mathbf{x}$, compared to the standard deviation of the md-feature, the less similar the patch $\mathbf{f}(\mathbf{x})$ centered on pixel $\mathbf{x}$ is to patches in R_i. When this difference is larger than $2s$ pixel $\mathbf{x}$ most likely does not belong to R_i.

The competing term is fundamental in the framework of region-based active contours. This term selects the region for which the a posteriori probability of a contour pixel $\mathbf{x}$ is highest. The competition force is computed for a contour pixel $\mathbf{x}$ by considering a local window $W_2(\mathbf{x})$ centered at on pixel $\mathbf{x}$. The observations in the local window already have kernel window widths computed as proposed in subsection 6.2.2. The pdf at feature vector $\hat{p}_i(\mathbf{f}(\mathbf{x}))$ is determined from these observations according to the variable kernel density estimator in equation 6.1. The pdf is estimated for each region R_i in the vicinity $W(\mathbf{x})$ of pixel $\mathbf{x}$. The growing term helps escape the local minima points where the competition force is very small. Alternatively, the size of the local window for competition can be varied. If a minimum is not optimal, the contour will start evolving anew. However, the time to pre-compute distances between feature vectors in each window increases with the window size exponentially. In our implementation, we choose to double the size of the local window W_2 for computing $\hat{p}_i$ every tenth iteration; these iterations then require more time, since distances necessary for this operation are not pre-computed.

The level-set energy and evolution equation. Having modified the growing term and consider-

ing the estimate $\hat{p}_i(\mathbf{f}(\mathbf{x}))$ for the competing term, the energy equation (5.5) is written as follows

$$E = -\sum_{i=1}^{m} \int\int_R H(\Phi_i) \left(\mathbf{1}_i^W(\mathbf{x}) g^{W_1}(\mathbf{x}) + (1 - \mathbf{1}_i^W(\mathbf{x})) \log \hat{p}_i(\mathbf{f}(\mathbf{x})) \right) - \frac{\mu}{2} |\nabla \Phi_i| d\mathbf{x}. \tag{6.8}$$

The gradient descent equation that minimizes the energy is similar to the one in (**??**):

$$\frac{\partial \Phi_i}{\partial t} = \mathbf{1}_i^W g^{W_1} + (1 - \mathbf{1}_i^W)(\log \hat{p}_i - \max_{j \neq i, \Phi_j < 0} \log \hat{p}_j) - \frac{\mu}{2} div \left(\frac{\nabla \Phi_i}{|\nabla \Phi_i|} \right). \tag{6.9}$$

As previously, we consider that region descriptors are constant during gradient descent; otherwise, we would have to use the shape gradient tool Jehan-Besson and Barlaud (2003) that introduces additional terms in the equations. These, in turn, substantially increase computation time. According to the gradient descent equation, each contour moves to assimilate pixels similar to the ones in its interior, and similarity is computed via md-features. Simultaneously, for contour pieces that are in the vicinity of another contour, the level-sets compete for pixels in between.

The computational advantage of the semiautomatic initialization. The initialization of active contours with piecewise constant behavior is semi-automatic. The user selects a few pixels in each region and circles are automatically generated around them, (e.g., see Fig. 6.10,d). In chapter 5, we have discussed that allows flexible segmentation of the same image. Semi-automatic initialization considerably contributes to the efficiency of the proposed texture segmentation algorithm, making it even faster when compared to global region descriptor methods.

With global region descriptors, automatic initialization is possible if many small initial contours are spread over the entire image, and, for each, the region index is specified. This means that competition forces must be computed for a large number of contour pixels. However, kernel density estimation of the pdf at each contour pixel is computationally very expensive, since every time a sufficient number of observations must be selected from the global region, and distances between observed feature vectors and the feature vector of the contour pixel must be calculated.

With the proposed semi-automatic initialization, the initial contours consist of a comparatively small number of pixels. The number of pixels on the contours becomes largest as contours grow close to image boundaries. This number is among the smallest in global active contours. The number of computations in the proposed method is thus substantially reduced and iterations can be executed much faster. Computation times will be discussed in more detail in the next section.

6.3.1. Implementation Details

We implement the level-set evolution with the fast two-cycle algorithm Shi and Karl (2008). As far as possible, computations for forces in (6.9) are approximated with integers. An important

implementation detail refers to the minimum number of samples necessary for computing the growing and competing terms. The formula of the variable KDE (6.1) requires that the feature vector where the pdf is evaluated is within the computed kernel window width for the observation, from any observation that contributes to the estimation. The number of observations that contribute to the estimation may thus be very small. For instance, this happens when evaluating the pdf at a feature vector of a pixel close to an edge. The estimated pdf is very unreliable in this situation.

A more robust evaluation is obtained if we require there be a minimum number of patches $\mathbf{f}(\mathbf{x}_s)$ around a pixel $\mathbf{x}$ such that $\mathbf{f}(\mathbf{x})$ is within one kernel window width $h_s(\mathbf{x}_s)$ from the patch at $\mathbf{x}_s$, i.e. $|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{x}_s)| < h_s(\mathbf{x}_s)$. Otherwise, the term is set to zero. In our implementation, $\hat{p}_i(\mathbf{f}(\mathbf{x}))$ and the md-feature of $\mathbf{x}$ are evaluated with a minimum of 5, respectively 20 such samples. The smaller the value for the minimum number of samples for the competing term, the higher the chances that a pixel gets assigned to one of two competing regions, even if in reality is should belong to a third region that did not yet arrive at the competing site. This often causes incorrect detection of some small boundary pieces. The higher the value of this number, the more holes form between regions, at an image boundary, since pixels close to edges have a decreased similarity to their respective region. The smaller the value for the minimum number of samples for estimating the md-feature, the more randomly the growing term behaves because the information contained in the md-feature is not reliable. The larger this value, the growing term will be slowed down in its evolution as the degree in which image patches vary gets larger.

An image is processed in a multi-scale setting where the image pyramid has two level. Level-set evolution in the higher level mostly corrects the details of the segmentation on the smaller image.

6.4. Segmentation of Natural Images

The segmentation method proposed in Section 6.3 based on texture descriptors presented in Section 6.2 was tested on natural color images. Images are represented in the CIE Lab color space. The last two images in Fig.6.10 were taken from the Berkley image database (Berkley, 2008) and the first three were selected because of the imaged subjects. All local windows are square-shaped and they have the same size for all segmented images. The size of the side of the window W_2 for determining the region-competition force is largest and is set to 31px ($W_2 =$ 31px). The size of the window W_1 for computing the md-feature is set to 21px, like in chapters 4 and 5, ($W_1 = 21$px). The size of the local window V for determining kernel window widths was set to 21px ($V = 21$px). Distance between pairs of patches were pre-computed in local windows of size 31px; the distance structure for each image can be thus initialized in a few seconds.

At the image border, all local windows are cropped to fit into the image. Feature vectors for pixels near the border have fewer components than within the image because they are generated from patches because they are also cropped.

Local descriptors were created from patches of size 5×5px in the intensity channel and 3×3px in each color channel. For the leopard image patches of size 3×3px in each channel proved to be a better descriptor for regions in the image. The parameter h_{max} was set to 600 for the first two images, 400 for the third and 300 for the fourth in Fig.6.10. Image d) of each row shows the intensity-coded size of window width $h_j(\mathbf{x})$ for the respective image; widths were scaled for presentation purposes, with dark pixels representing small widths. Pixels with large $h_j(\mathbf{x})$ can be observed at boundaries between objects, but also within regions, when the patch size does not capture texture properties best, e.g. the ground in the second image where there are large variations in texture scale due to stones. On one hand, this suggests that variable window widths can capture data properties well; on the other, image size and object textures influence widths computation and this is reflected in having to somewhat adjust h_{max}. However, the goal of future research is to compute h_{max} from the data, similar to approaches for computing a fixed window width Awate et al. (2006).

Previously, we have imposed a condition that energy terms are evaluated from a minimum number of patches and this has two effects. First, some image elements remain in region R_0 because their visual appearance is not similar to any of the regions within contours, e.g. the stone under the prairie dog in the second image. Second, the growing term reliably stops when there is a sudden change in appearance even in the absence of a competing contour. This is the case for the region above the cat (first row), the one above the polar bears (third row) and the "pirate eye" of the mouse. What seems to be a glitch in the segmentation, contains the subject's tail and some other objects, which on a closer look are not identifiable with certainty. Note that these errors are very few, although the segmented images are difficult. Because local descriptors were used, the two bears were accurately segmented into two different regions; this would have not been possible with global descriptors.

For processing, two levels for an image pyramid are computed. The sizes of images at the lowest level are 300×217px, 284×420px, 212×180px. 284×420px and 284×420px. For these images, distances for the local sampling strategy are computed on average in 5 seconds and one iteration for evolving level-sets takes on average 0.5 seconds. For global level-sets, 3 minutes per iteration were reported for a 256×256px image Awate et al. (2006) (we measured 2 minutes per iteration on our machine for a similar but simpler method). Commonly, at least tens of iterations are needed. Our method is thus two orders of magnitude faster than a global one and it requires minimal user interaction. When $\hat{p}_i$ is evaluated from samples in $2W_2$ where distances are not pre-computed computation time increases to 35 seconds, on average.

To summarize, we have presented and demonstrated very good segmentation results on natural images using level-sets driven by region descriptors from image patches. Their probability distributions in a local image region is estimated with a variable kernel estimator. For this estimator, we compute the window width h_s at a pixel $\mathbf{x}_s$ by finding a good value k for selecting the k^{th} local nearest neighbor of the patch at $\mathbf{x}_s$. To achieve short computation times, a patch's probability is also evaluated locally, from samples taken from a window around the patch. For the same purpose, initialization is semi-automatic. As a result, our local method is two orders of

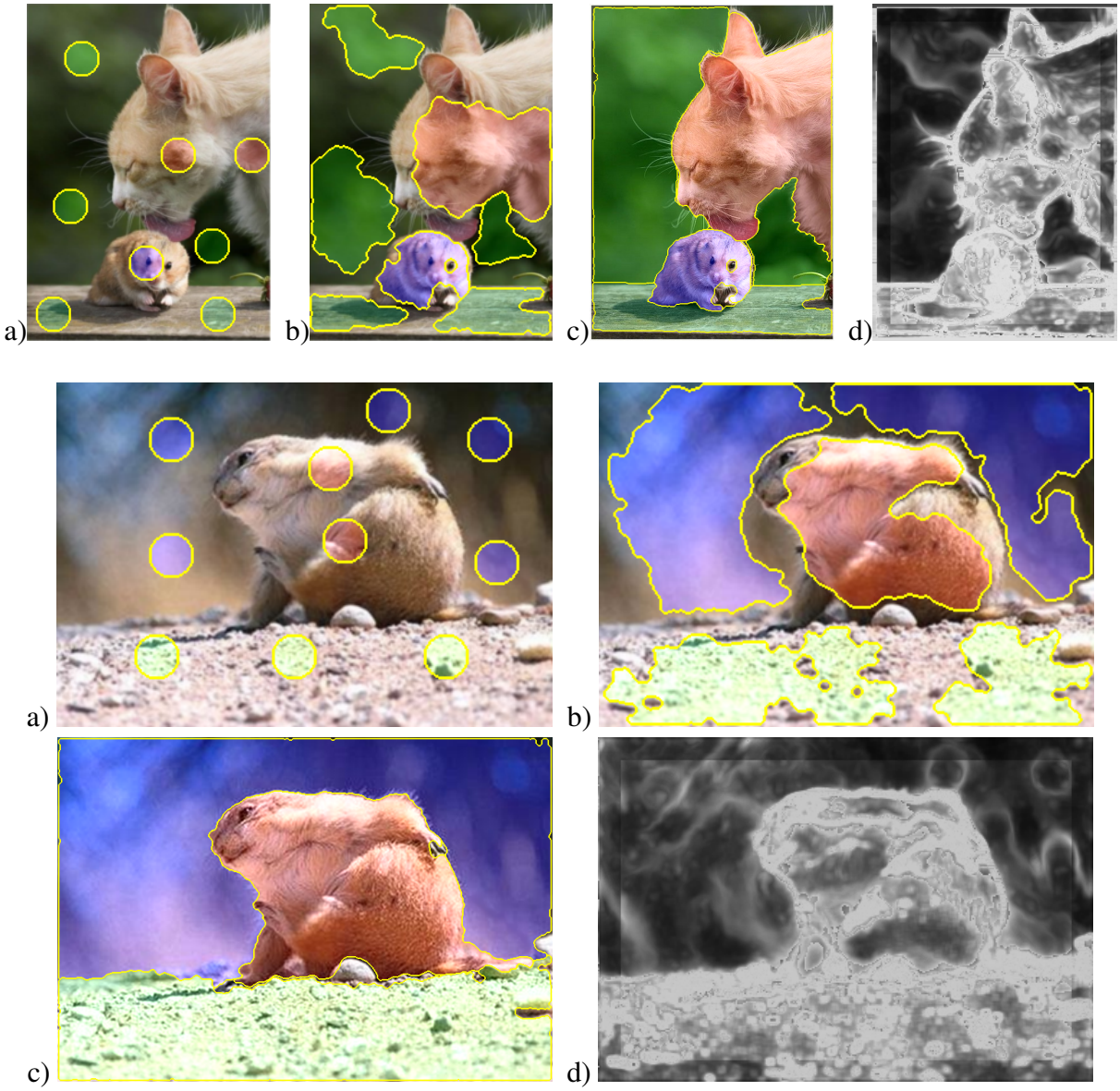


Figure 6.8.: Results of the proposed segmentation method on natural images selected due to the photographed subjects: a) level-set initialization; b) level-sets at an intermediary evolution step; c) final segmentation; d) image with the intensity-coded size of h_j. Images a) to d) for the same picture are referred to in the text as "a row". (Please zoom in in the electronic version for more details.)

magnitude faster than a global one involving global initialization and sampling. The drawback of local methods compared to global ones, is that their associate energies have more local minima where level-sets may get trapped. To deal with this issue, we use active contours with a special property: different contour pieces may behave in different ways, one piece may grow while an-

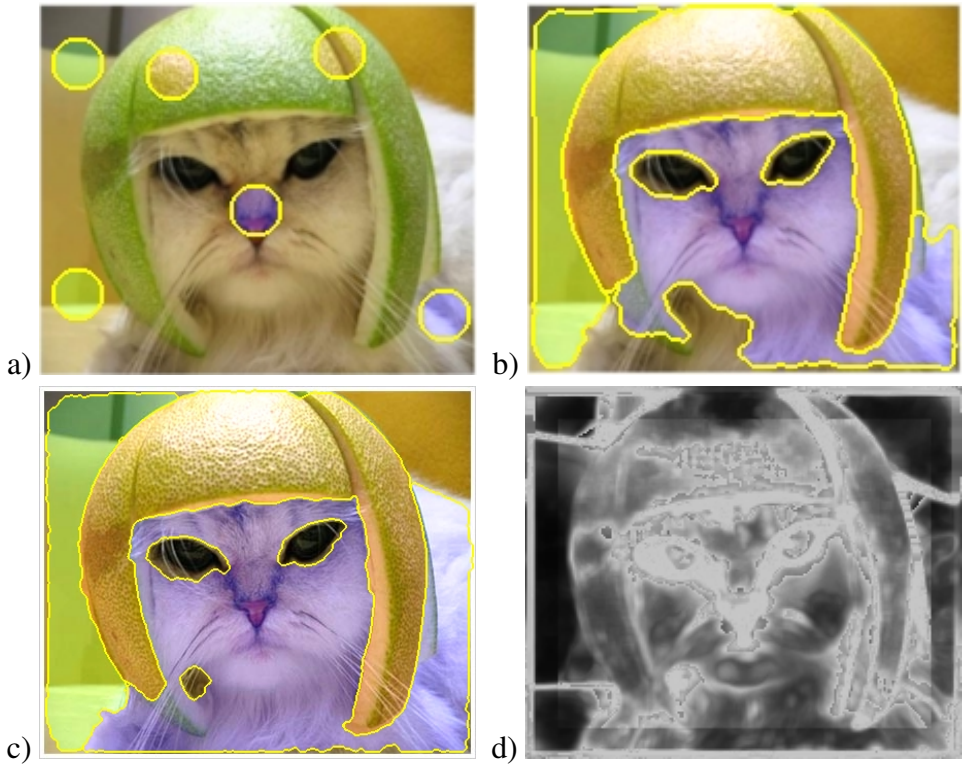


Figure 6.9.: Results of the proposed segmentation method on natural images selected due to the photographed subjects: a) level-set initialization; b) level-sets at an intermediary evolution step; c) final segmentation; d) image with the intensity-coded size of h_j. Images a) to d) for the same picture are referred to in the text as "a row". (Please zoom in in the electronic version for more details.)

other contour piece is competing with neighboring level-sets. To create the growing term, we have designed a one-dimensional mean-distance feature that is approximately locally Gaussian and that correctly finds most of the boundaries. However, a formal justification missing, a better understanding of md-features is needed. Patch size strongly influences the shape of the PD over a region. Finding the best patch size for each region and finding the best width h_s for each patch are related issues that still need a solution.

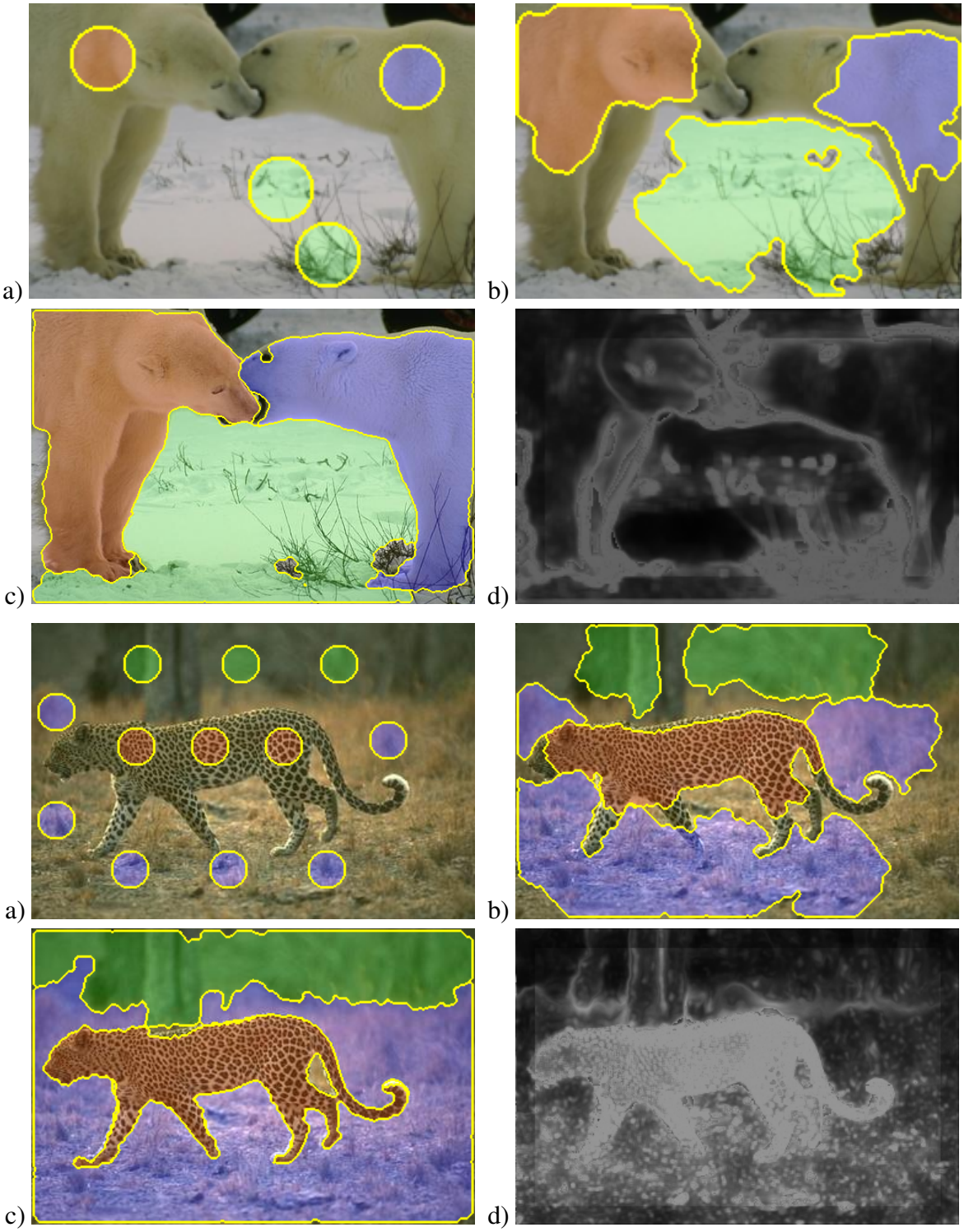


Figure 6.10.: Results of the proposed segmentation method on natural images from the Berkley image database: a) level-set initialization; b) level-sets at an intermediary evolution step; c) final segmentation; d) image with the intensity-coded size of h_j. Images a) to d) for the same picture are referred to in the text as "a row". (Please zoom in in the electronic version for more details.)

7. Marker free Hand Motion Measurement with a Multicamera Vision System

> There is something fascinating about science. One gets such wholesale returns of conjecture out of such a trifling investment of fact.
>
> *Mark Twain*

This chapter presents a marker free computer vision system for tracking the articulated movement of hands in order to measure surgical skill. In this system, the hands and arms of a person are tracked with the help of local region descriptors and fast level sets. The accurate hand and arm contours are processed to obtain so-called anatomical features, which are the middle of the arm, the wrist, and the palm. Knowing 2D feature positions in individual cameras, the 3D feature positions result from employing principles of stereo vision. In our experiments we show that the anatomical features are sufficient for tracking the trajectory and motion of arms and hands with the proposed system.

7.1. Hand Tracking: Motivation, Technologies and Applications

At the sight of a messy room one may start to daydream about possessing a genie. She could clean the room at the wave of the hand and do the dishes at the blink of an eye (like Major Nelson's Jeannie from the image (Jeannie, 2008)). An ideal cleaning machine is the scientific version of a genie without the communication skills. One question is, how do we rapidly get our wishes into the machine? Hand gestures constitute an important means of communication, and they are indispensable to object manipulation. Seeing the hands of a person, most people would have no difficulties identifying what these sentences mean: "Put that there", "The dog was about this size", "Hold the onion like this and the knife like this". Hands can be used to grasp, hold, manipulate, assemble, move, point, show, write, paint, explain, express, caress. They

can manipulate things, but they can also convey ideas or feelings. [1] Consequently, a machine that could understand hand gestures would be very valuable. A gesture interface can look very impressive, as Steven Spielberg's "Minority Report" illustrates, but its value lies mainly in the fact that hand motion comes natural and easy.

Already in the early ages of home computers, researchers set out to create machines that accept hand motion as input. For example, a "Put-that-there" interface was presented in 1980 (Bolt, 1980). This and other pioneering works have shown the difficulty of creating such machines. The machines first encounter all the difficulties present in computer vision. The machine must be capable of identifying if and where hands are present in an image(-stream). Additionally, the application may require that changes in both hand position and posture are identified and tracked. *Hand posture* denotes the configuration of the hand at a given moment in time, i.e. the bending and position of the fingers and of the palm.

The difficulty of tracking hands with computer vision systems depends on the application's context, and on what exactly is being tracked. For some applications, it is sufficient to isolate the region occupied by the hand, while for others it is important to determine if the hand is in one of few known postures. The most ambitious applications require that the posture of the highly flexible hand be constantly recognized. Concerning the context, tracking the positions of hands is quite uncomplicated if only a uniform, static background and the hands are visible in the picture. However, this is rarely the case. Hands are, in general, attached to bodies, which usually exist in cluttered environments with varying illumination. In such cases, robustly solving the tracking problem suddenly becomes very difficult. The application designer needs to choose features or models that best describe the hand in the context and for the type of tracking required by the application. In the following we give a short overview of modeling possibilities, while we mostly reference recent relevant work.

A feature that naturally describes hands is skin color (Kurata et al., 2001; Wu and Huang, 2002; Kölsch and Turk, 2004b). This feature is modeled as a random variable with a probability density that needs to be determined. At the same time, one also estimates the probability density of colors in the background, typically employing the maximum a-posteriori principle. Properties of image acquisition systems, changes in illumination, and changes in environment cause probability densities of colors in the background and skin to vary between frames. To make tracking algorithms more robust, density models for background and skin colors are updated in every video frame. Algorithms also have to address the problem of learning initial distributions, since different people have different skin colors. There are various approaches to update color models and they often incorporate spatial information regarding pixels in the same frame and pixels in consecutive frames (Kurata et al., 2001). Spatial information between frames may be obtained by tracking features (Kölsch and Turk, 2004b) or by assuming the hand follows a motion model (Chang et al., 2005b).

[1] In fact, some peoples depend on their hands for communication: tie an Italian's hands and chances are good that this will interfere with his communication ability.

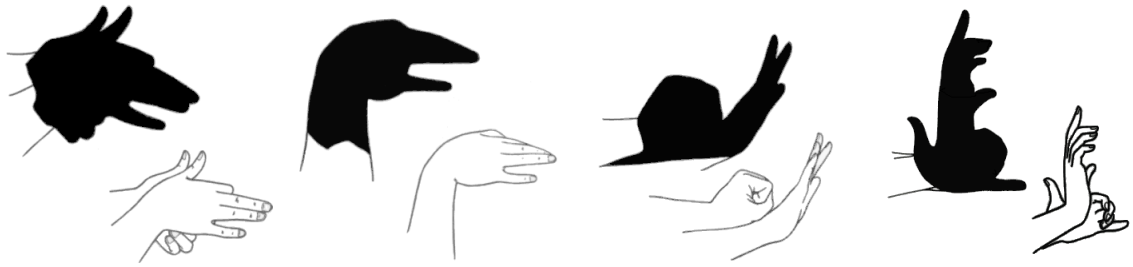


Figure 7.1.: Different hand configurations project into very different 2D shapes (Hands, 2008).

When a stereo-camera system is available, depth information may take the role of color, if we assume that hands are the objects closest to the cameras (Wilson and Oliver, 2003; de la Hamette and Troester, 2008). In user interaction scenarios, this is a reasonable assumption. Depth information is gained from depth-map algorithms. The robustness of these algorithms is also increased by incorporating additional sources of information, like motion detected by block matching (Wilson and Oliver, 2003) or color (de la Hamette and Troester, 2008).

Both color models and depth-maps algorithms rely on assumption about the world that are often not met. We are familiar with issues of statistical modeling from previous chapters, and depth-map computation constitutes an entire field in computer vision. Many researches have thus opted to integrate a priori shape information into their algorithms.

The human hand is a system of connected rods with 21 degrees of freedom. It is thus a highly articulated object. As a result, the shapes it projects on a 2D surface may be very different. Four such situations are exemplified in Fig. 7.1. In order to model an object of such complexity, one either restricts the number of postures that can be tracked by the system, or uses very complex, computationally expensive models that might not be robust.

A simple idea is to learn what a limited number of hand postures looks like in different environments. Amazingly, this idea has a simple and very fast solution in the ingenious method by Viola and Jones. (2001b,a). They propose to simultaneously learn the appearance and shape of an object using very large sets of very simple features, called rectangle features. Rectangle features are obtained by summing the values of pixels in rectangular windows and subsequently subtract or add the sums given some configuration. For example, in Fig.7.2, the sums in black rectangles are subtracted from sums in white rectangles. In this manner, features representative for a hand posture are learned from examples like the ones in Fig. 7.3. It has been shown that algorithms based on the rectangle features can yield good real-time tracking results, but that they are also extremely sensitive to rotation and deforming transformations (Kölsch and Turk, 2004a;

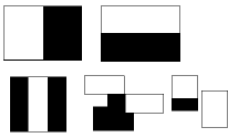


Figure 7.2.: Rectangle features.

Figure 7.3.: Examples for learning rectangle features for different hand postures in different environments (Kölsch and Turk, 2004a).

Barczak et al., 2005).

Another simple idea is to design the application in such a way that it only has to recognize one hand posture (T. and Weissman, 1995) or one template (O'Hagan and Zelinsky, 2000). For example, the television control application presented in (T. and Weissman, 1995) compensates the use of a single hand posture, the opened hand, by providing rich visual interface and rich visual feedback. However, in many applications one either needs a richer interaction vocabulary, or one is interested in the exact configuration of the hand joints. In these situations, researchers construct hand models to help them (Licsar and Sziranyi, 2002).

Simplified models are designed for the needs of the application, and Fig. 7.4 shows four examples. The model in Fig.7.4(a) uses windows to specify the possible positions of palm and fingers and their spatial relationship (Abe et al., 2000). The point of reference for this model is the center of the palm, which is the point farthest away from all detected hand contours. This point is detected using the distance transform, an idea also encountered in (Oka et al., 2002), and it has the advantage that it is invariant to hand opening and closing. The model is employed in interfaces for 3D drawing applications (Abe et al., 2000; Oka et al., 2002). Fig.7.4(b) shows a hand model for television control in cluttered environments (Bretzner et al., 2002). It is based on the view that palms, fingers, and fingertips look like round or elongated blobs that occur in given spatial configurations when one, two, three, four or five fingers are shown to the camera simultaneously. The image shows the configuration for a hand showing three fingers. The *cardboard* model in Fig.7.4(c) can track a still hand that simultaneously flexes multiple fingers (Wu et al., 2001a), while the model in Fig.7.4(d) can track a pointing hand that is moving very fast. This last model is constructed using B-spline functions and control points (Isard and Blake, 1998b).

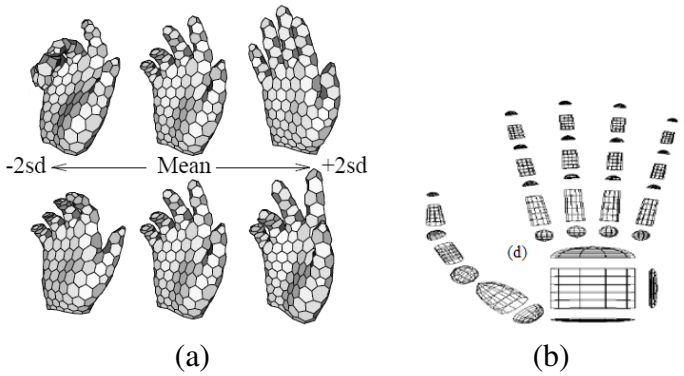
One problem of these simplified models is that they cannot deal with hand rotations and simultaneous motion of multiple fingers. Furthermore, the models may identify hand postures when the posture is in fact not present in the image. To solve these problems, researchers have used 3D hand models. 3D models are not given the full number of degrees of freedom, either by not constructing a full model, or by fixing some of the parameters. The number of model parameters, which specify the degrees of freedom, are constrained to a few, while other degrees of freedom are ignored or considered constant. Otherwise, the problem of determining the values of parameters to match the image data would be intractable. For example, the user rests their hand in a hand-box thus eliminating three degrees of freedom (Noelker and Ritter, 1999). The

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

Figure 7.4.: Simplified hand models. (a) Windows specify the possible positions of fingers relative to each other (Abe et al., 2000). (b) The hand is composed of circular and elongated blobs in the shown configuration (Bretzner et al., 2002). (c) The cardboard model approximates each finger by a rectangle (Wu et al., 2001a). (d) The shape of a pointing hand is estimated using B-splines (Isard and Blake, 1998b).

number of parameters is reduced to 10 by making simplifying assumptions about finger joints [2]. The appearance of finger tips is learned and detected using a neural network, and the model parameters are also determined by neural networks. The authors propose to use the system to analyze the finger motion of a music conductor.

Natural hand motions do not fully exploit all degrees of freedom. An experimental statistical analysis of natural hand configuration has shown that most of the feasible hand configurations can be reached from a set of basis configurations, by linearly combining them (Wu et al., 2001b). As a result, a 3D-mesh model can be employed to learn various hand configurations and their possible "deformations" with active shape models (Heap and Hogg, 1996). Fig. 7.5(a) shows such a hand model and its deformation along the first and second modes of variation. Starting from an initial configuration, that must be relatively close to the position and posture of the hand in the image, the model is deformed to match data from video images. Tracking can be achieved if hand motion between frames is small, since for both model deformation and image data processing, complex algorithms are needed (Heap and Hogg, 1996; Wu et al., 2001b). For example, transition probabilities between different hand configurations are modeled with an optimized Bayesian tree (Stenger et al., 2006), and chamfer matching is used to match the model's 2D projection to the image edge map (Thayananthan et al., 2003). The hand model, shown in Fig. 7.5(b), was created from 3D geometric primitives. Although the model includes all degrees of freedom, in practice only six of its parameters were allowed to vary simultaneously, since otherwise the algorithm's



(a) (b)

Figure 7.5.: 3D hand models. (a) A deformable mesh representation showing the hand mean-shape and its deformations along the principal variation axes in learning examples (Heap and Hogg, 1996). (b) A full 3D model created from geometrical primitives (Stenger, 2004).

complexity grows too large (Stenger et al., 2006).

7.2. Hand Tracking for Surgical Motion Measurement

Our hand tracking application is designed specifically for measuring hand motion during simple surgical procedures. This section presents first the rationale of this application. Second, it motivates the choice of a different hand tracking approach than the ones overviewed in the previous section.

A surgeon's manual skill is vital for the patient and the doctor. As such, training surgeons is an important, time consuming activity, that involves the assessment of acquired skills. Recently, computer systems have been introduced to support the search for objective measures of surgical skill. Two types of systems are widely used to measure hand motion: virtual reality simulators like MIST-VR (Inc, 2008) or daVinci (Surgical, 2008), or electromagnetic based tracker systems, like the Imperial College Surgical Assessment Device (ICSAD). Virtual reality systems typically permit the practice of laparoscopic procedures. The ICSAD system is employed in tracking conventional procedures, but it requires the trainees to wear a significant amount of gear which may interfere with their skills.

Our work explores the use of computer vision based hand tracking systems as an alternative to electromagnetic trackers. Computers eyed with video cameras can monitor a surgeon's hand movements while the surgeon is performing simple surgical tasks. The advantages of a computer vision system refer to comfort and time. The trainee is not required to wear any hardware gear so

[2]It is know for example that certain hand configurations are not possible, like bending some finger without bending another.

that his movements remain natural and unimpeded. Simultaneously, the trainees need not waste time by attaching gear to themselves. Saving time is especially important when the users of the system are surgeons, since these people are notoriously always in a hurry.

Measuring the surgical skill of practicing surgeons using machines is just as important as evaluating the skills of trainees. First, this helps create objective measures for surgical skill evaluation. Up to present, objective quantities for surgical skill assessment are the number of movements, the distance traveled by each hand during the procedure and the speed at which movements are executed (Datta et al., 2001). The surgeon that executes fewer, less ample movements is considered rated better than the surgeon that executes more movements.

The example we concentrate on in this work is the simple suturing procedure in conventional surgery. Motion patterns involved in this procedure include reaching for the needle, positioning/holding the needle and inserting/pushing the needle through tissue (Lin et al., 2005). Second, the analysis of hand motion during surgical procedures provides important information for the design of laparoscopic instruments.

The motion of hands can be measured by selecting or fixing landmarks on the hand and determining their 3D position in a sequence of images. In an ideal situation, landmarks would coincide with the joints of the hand, and their positions would be determined by matching the image data to a 3D hand model. As discussed in the previous section, a tractable algorithm can be designed only if the number of parameters in the model is constrained. For determining the constrained parameter subspace, one needs to acquire a set of learning examples, segment them and establish the correspondence with the model, while designing a method for dimensionality reduction. The task proves to be complicated because the hands hold instruments, such that the instruments occlude parts of the hand, while other hand parts are occluded by the hand itself. Additionally, different surgeons grasp and hold instruments in different ways, such that their hand configurations may differ considerably. Also, during the procedure, hands rotate such that visible parts become occluded and occluded parts become visible, leaving the researcher with the problem of dealing with partial visibility of landmarks.

In order to avoid these problems, we opt for an accurate segmentation of hand contours based on their appearance described with local region descriptors. Accurate contours are necessary in order to obtain reliable landmarks. We define these reliable landmarks and name them *anatomical landmarks or features*. To compute the positions of landmarks in space we use a stereo vision system. The fact that our system has multiple cameras can be employed to make measurements more exact. The system functions optimally when hands and arms are in contrast with their immediate background. This requirement can be easily met in the setup we propose if the trainee or surgeon wears a dark sweater with tucked sleeves. There are no other restrictions to the system. Its software and the hardware setup, as well as the necessary technical background in stereo vision are presented in the following.

7.3. Multicamera Setup and Principles of Stereo Vision

The setup and methods for marker free motion measurement have been introduced in (Darolti et al., 2007), and this chapter details both. Surgeons and trainees perform surgical procedures in a workspace designed for this purpose. The workspace consists of an aluminum rack (120 x 200 x 60 cm) shown in Fig. 7.6. A transparent acrylic glass plate is hinged in the rack and can be adjusted to reside 90 to 140 cm above ground to yield an optimal working height for the operator. An object serving as dummy for the surgical procedure is fixed on the plate. Two photo umbrellas are used to illuminate the scene.

The rack is equipped with six digital monochrome progressive scan BaumerLink iX60-s cameras (780 x 582 px, 50 frames per second) wearing Pentax H612A-TH lenses. The amount of data from six color cameras would have exceeded the capacity of the data bus at the time the hardware for the project was aquired. The cameras are accessed through two Baumer PCI-A25 PCI interface boards (FireWire based) connected to the same desktop computer. Each board hosts four camera connectors and offers the possibility to synchronize the cameras. The six cameras can slide along the rack-bars and can be flexibly positioned. The surgical motion can thus be viewed from multiple different angles - including the "from under the table" view. An example of viewing the same scene from six different positions is given in Fig.7.6. The figure also shows the definition of the coordinate system in this setup. In our final setup, the cameras build three narrow-base stereo-pairs: a pair looks at the scene from the top, one from the left side an one from “under the table”. Stereo-pairs need to have narrow bases to ensure that the same landmarks are detected in each camera, as will be explained later in this chapter. First, we define stereo vision and related terms.

7.3.1. Stereo vision

This chapter gives a concise overview of the principles of stereo vision involved in the proposed system for the purpose of 3D measurements from 2D points. For an introduction on *camera models* and *stereopsis* see Trucco and Verri (1998). For a detailed treatment of *multiple view geometry* and algorithms for computer vision see Hartley and Zisserman (2004). The concepts used or defined in this section can be found in both references.

The video camera of the hardware setup are modeled as *perspective pinhole* cameras. Let $\mathbf{X}$ be a point in a 3D *world reference frame* and let $\mathbf{x}$ be the projection of this point in the *image reference frame* of the image recorded by a camera. The positions of both points is expressed in *homogeneous coordinates*; for a textbook introduction on homogeneous coordinates see (Hartley and Zisserman, 2004). For a 3D point $\mathbf{X} = (X_1, X_2, X_3, X_4)$ expressed in homogeneous coordinates, the following relation holds: $X_1/X_4 = X_2/X_4 = X_3/X_4$, and these fractions are the coordinates of the point in the 3D coordinate system. Analogously, for a 2D point $\mathbf{x} = (x, y, w)$ expressed in homogeneous coordinates, the coordinates in the camera coordinate system are $x/w, y/w$. Given the

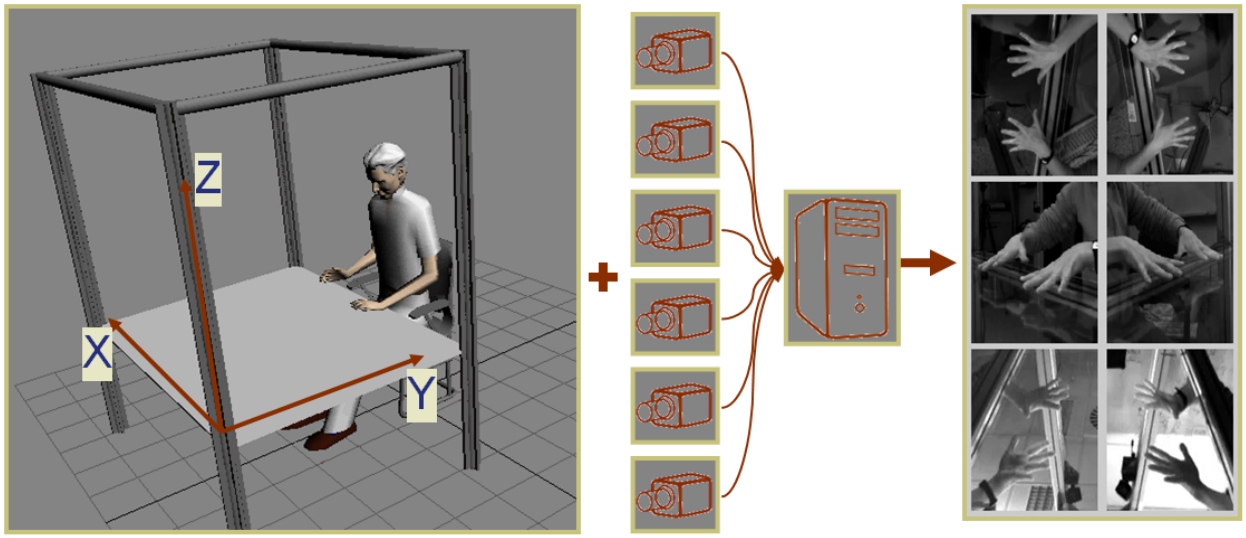


Figure 7.6.: The hardware setup of the multicamera vision system composed of a rack, six monochrome cameras and a desktop computer. An example for viewing the same scenes through the flexibly positionable cameras is also shown.

position of the point in the world reference frame, $\mathbf{X} \in R^4$, the position of point $\mathbf{x} \in R^3$ projected in the camera reference frame can be determined with the help of the 4×3 *camera projection matrix* $\mathbf{P}$ according to the following system of equations in matrix form

$$\mathbf{x} = \mathbf{P}\mathbf{X}. \tag{7.1}$$

The entries in the camera projection matrix $\mathbf{P}$ can be determined from n known correspondences $\mathbf{X}_i \leftrightarrow \mathbf{x}_i$, $i = 1..n$ by writing the equations resulting from the equalities $\mathbf{x}_i = P\mathbf{X}_i$.

In general, the image positions $\mathbf{x}$ are the ones that are known. Geometrically, all 3D-points on a line project to the 2D-point. This line, denoted as $\mathbf{X}(\lambda)$, passes through the *camera center*, denoted as $\mathbf{C}$, and the position of $\mathbf{x}$ in 3D space, $\mathbf{P}^+\mathbf{x}$. $\mathbf{P}^+$ is the *pseudo-inverse* of matrix $\mathbf{P}$ (Hartley and Zisserman, 2004). The equation for this line, known as *back-projection line* is written as

$$\mathbf{X}(\lambda) = \mathbf{P}^+\mathbf{x} + \lambda\mathbf{C}, \tag{7.2}$$

where λ is the parameter that describes the line.

To determine the position of a particular point on this line, addition information must be gained from a second camera. A system of two or more cameras constructed for this purpose is a *stereo system*. The ambiguity of seeing a point through a single camera is illustrated in Fig.7.7(b).

Both images in this figure illustrate the geometry of a stereo system. The cameras are indicated by their centers $\mathbf{C}$ and $\mathbf{C}'$ and by their image planes in Fig.7.7. A space point $\mathbf{X}$ and its images $\mathbf{x}$

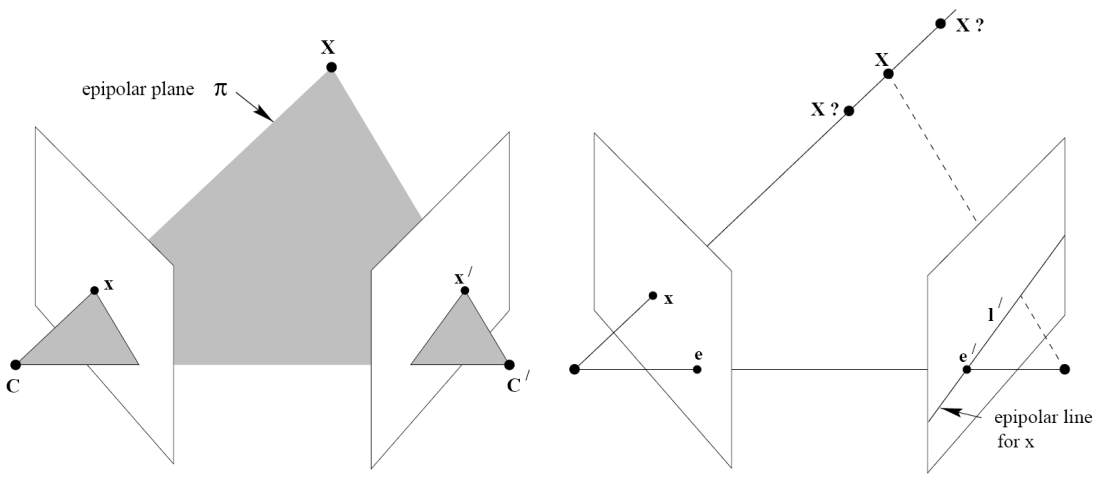


Figure 7.7.: Stereo geometry: point correspondences and epipolar lines (Hartley and Zisserman, 2004) p.220. See text for explanation.

and $\mathbf{x}'$ define a plane named *epipolar plane*, denoted in the figure as π. This plane is alternatively defined by two lines that lie in this plane (Fig. 7.7(b)), and that can be determined from a single projection point $\mathbf{x}$. One of the lines is the back-projection line for point $\mathbf{x}$. The other one is the image of the back-projection line in camera $\mathbf{C}'$, which is known as the *epipolar line* for point $\mathbf{x}$, and is denoted in the figure with $\mathbf{l}'$. The corresponding point for point $\mathbf{x}$, denoted as $\mathbf{x}'$, must thus lie on this line. Knowing $\mathbf{x}'$, and the projection matrices of the two cameras $\mathbf{P}$ and $\mathbf{P}'$ completely determines the position of the 3D-point $\mathbf{X}$, as shown by the dotted line in Fig.7.7(b).

Triangulation is an ancient, well-known method for determining the 3D-point $\mathbf{X}$ knowing its images $\mathbf{x}$ and $\mathbf{x}'$ in the two cameras. The method relies on the straight forward observation that the back-projection lines through $\mathbf{x}$ and $\mathbf{x}'$ intersect at point $\mathbf{X}$. In practice, these lines do not intersect due to measurement errors, and triangulation algorithms must compensate for this issue.

The needed camera projection matrices are determined by *camera calibration* methods. Cameras may be calibrated independently, however calibrating the stereo pair simultaneously has the advantage of determining the orientation of the cameras in the same world reference frame. Camera orientation is one of the components of a camera's projection matrix, and it is given by camera rotation and translation with respect to the origin of the reference frame. The other component is a transformation that maps image points to 3D-points in the *camera reference frame* which has its origin at the camera center $\mathbf{C}$. Camera lenses typically distort the recorded image. In order to correct the image, it is necessary to determine the parameters of the distortion assuming it can be described by a radial and a tangential model (Trucco and Verri, 1998).

A *calibration object* with known geometry (e.g. a checker board) is imaged by the stereo pair to start the calibration process. The calibration object should enable a fast automatic detection

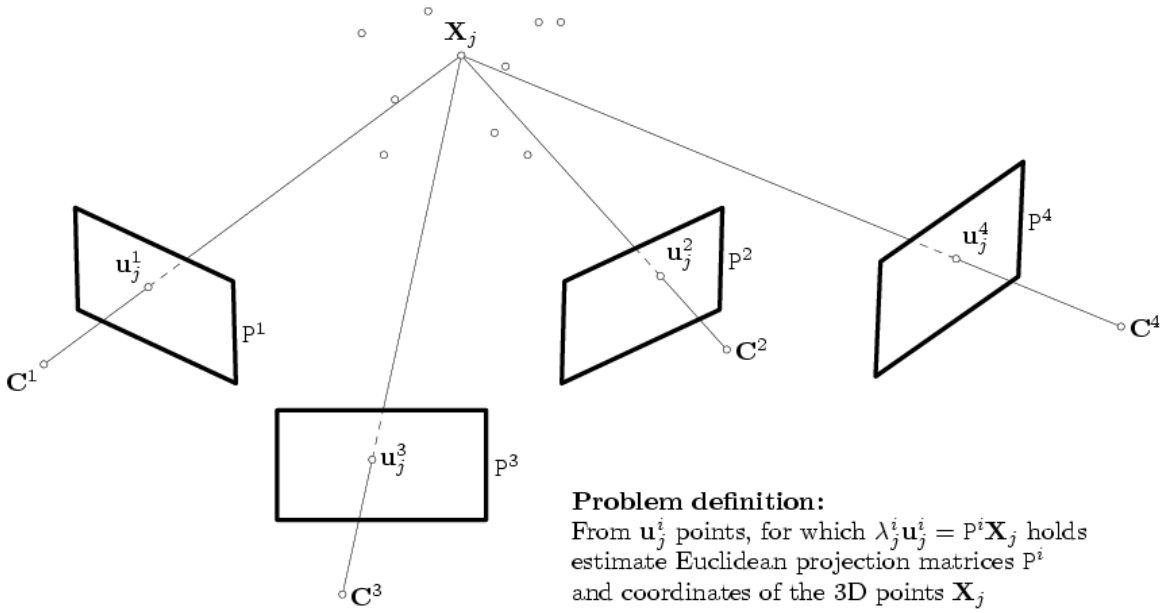


Figure 7.8.: The formulation of the multiple camera calibration problem (Svoboda et al., 2005a).

of corresponding images of the same 3D point. Once correspondences are known, the system of equations can be solved for the parameters of the projection matrices. One must ensure sufficient correspondences have been found to create an over-determined system. The system must be over-determined so that one can compensate for measurement errors, as explained for example in the textbook (Hartley and Zisserman, 2004).

7.3.2. Multiple camera calibration

Calibrating the cameras in the multiple camera setup two by two would involve a large amount of hand work. This is not practicable, since cameras can be easily displaced, invalidating the calibration matrices. Svoboda et al. (2005a) proposed a method for simultaneous calibration of multiple cameras, accompanied by a piece of free software available at (Svoboda et al., 2005b)). The calibration object is a punctual light source constructed from a laser pointer. The points necessary for calibration are obtained by simply waving the calibration point through the volume viewed by the cameras. The set of equations projecting the point in each camera can be written for each calibration point, as illustrated in Fig. 7.8.

To solve the set of equations according to theoretical results from the field of self-calibration methods, the software robustly tackles some practical problems. The software starts by solving the image processing problem of finding the images of the calibration point in the recorded images. In an ideal setting, the calibration point is the only bright object in a dark room. Its projection will be larger than one pixel, thus being an measurement error source which disturbs the

calibration process. The calibration routine is thus iterative. It starts by estimating the projection matrices, the distortion parameters and the 3D calibration points from all detected 2D points. It then iteratively discards 2D points that do not fit the model by RANSAC analysis (Hartley and Zisserman, 2004) and recomputes calibration parameters.

The software also provides a routine for reconstructing the geometry of imaged objects based on a routine presented in (Hartley and Zisserman, 2004). Fig. 7.9 shows the geometry of the reconstructed waved motions executed in a calibration experiment with the setup proposed in this chapter. One may observe that the results of the calibration are not aligned with the world reference systems: the cameras denoted in the figure by 0 and 1 are the top cameras in the setup, but in the figure they appear on the side. This is because the software uses the relative positions of recorded points, not their absolute one. The software can align the result of the calibration with the setup's world reference frame, if the user provides the positions of three cameras in this frame. For measuring motion, the absolute position of the reference frame for the calibration is irrelevant. However, the results can be controlled better if they are displayed in a meaningful world reference frame.

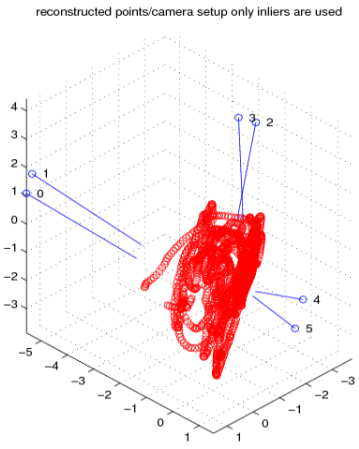


Figure 7.9.:

7.3.3. Image rectification

Given a stereo camera pair, and a 2D-point $\mathbf{x}$ in a camera image, epipolar geometry ensures that its corresponding 2D-point $\mathbf{x}'$ in the other camera lies on the epipolar line of point $\mathbf{x}$. As a practical matter, one wants a fast and simple method for determining the image pixels which lie on this line. When camera image planes in a stereo pair are parallel, the image of a camera's center in the other camera is at infinity, and hence, the epipolar lines are parallel, as shown in Fig. 7.10. This means that the epipolar lines are image rows, if the epipolar lines are horizontal, or image columns, if the lines are vertical (Hartley and Zisserman, 2004).

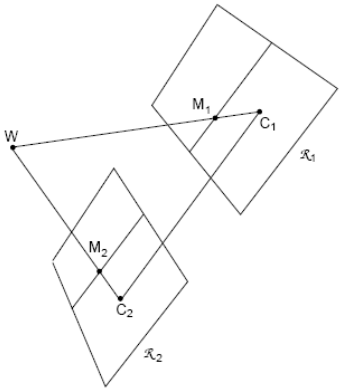


Figure 7.10.: Rectified cameras(Fusiello et al., 2000).

In general, two cameras will not have parallel image planes. The idea behind *rectification* is to construct two new projection matrices from the old ones such that the cameras defined by the new projection matrices have parallel, horizontal (or vertical) epipolar lines. The recorded images are *rectified* to create the image that would have been ob-

tained if the cameras had indeed have parallel image planes. The rectifying image transformation is computed from the new projection matrices. In this work, the new projection matrices and the rectifying transformation were computed with the compact algorithm proposed by Fusiello et al. (2000).

7.4. Marker Free Hand Motion Measurement

The measurement of hand motion with the designed computer vision system does not occur in real time due to computational requirements. As a result, the suturing procedure executed by a trainee or surgeon is recorded from the six perspectives offered by the six cameras. The frames in the video streams are compressed using the *ffmpeg* library (ffmpeg, 2005) before they are saved. Images can thus be saved at a rate of approximatively 18 frames per second (fps) with image quality loss that makes hand contours slightly unsharp. Short videos of the background are recorded for each camera, additional to the recorded suturing procedure. Image processing for motion measurement starts with these saved video streams. The steps and components of the measurement methods can be seen in Fig. 7.11.

Videos recorded by a camera pair are rectified frame by frame according to Section 7.3.3. As a result, the epipolar lines of the rectified images are parallel and horizontal. A background subtracted image is computed for each frame to provide additional information for contour initialization and tracking. Accurate contours of hands are segmented and tracked in each video independently using active contours based on local region descriptors. The segmented frames are processed to detect the 2D positions of anatomical features in each frame. Since the calibration matrix of the multiple camera system is known, the 2D positions are corrected to account for the lens distortion, and are then used to triangulate the 3D positions of the landmarks in each frame using the reconstruction routine according to the methods described in Section 7.3.2.

7.4.1. Accurate Tracking of Hand and Arm Contours

Due to different viewing conditions, methods for segmenting images from the side cameras need to be more robust than those for segmenting images from the top cameras.

In the top cameras the contrast between hands and their immediate background is large, since the trainee or surgeon wears a dark sweater. We thus assume that the skin surface will appear as patches of light intensity in the recorded images. Other light patches viewed by the cameras may be almost entirely eliminated by background subtraction. For this purpose an average background image is computed from the background recording. Exemplary background and original images are shown in Fig. 7.12 (a) and (b).

The background image is subtracted from each processed frame, and pixels in the result are set

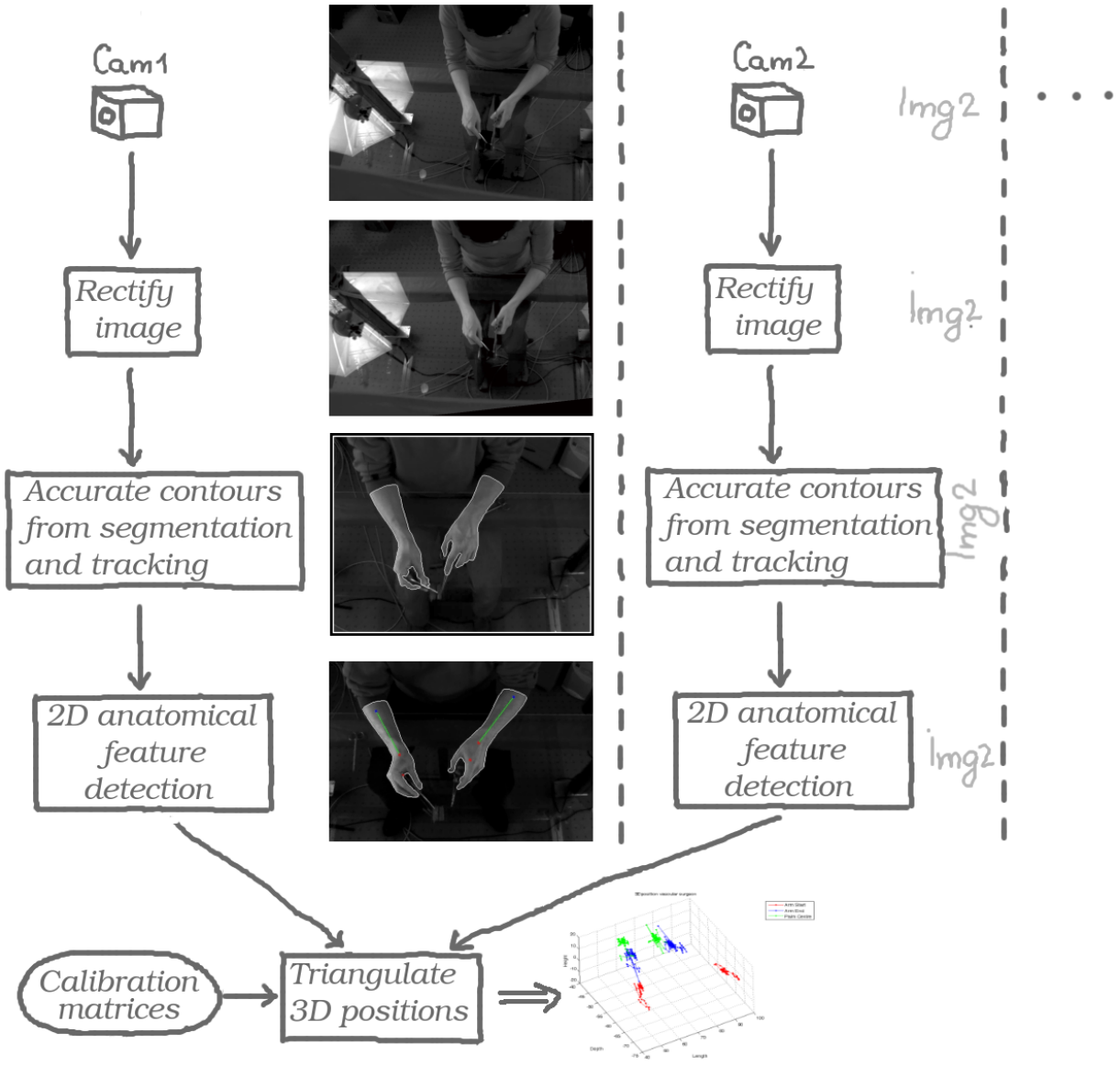
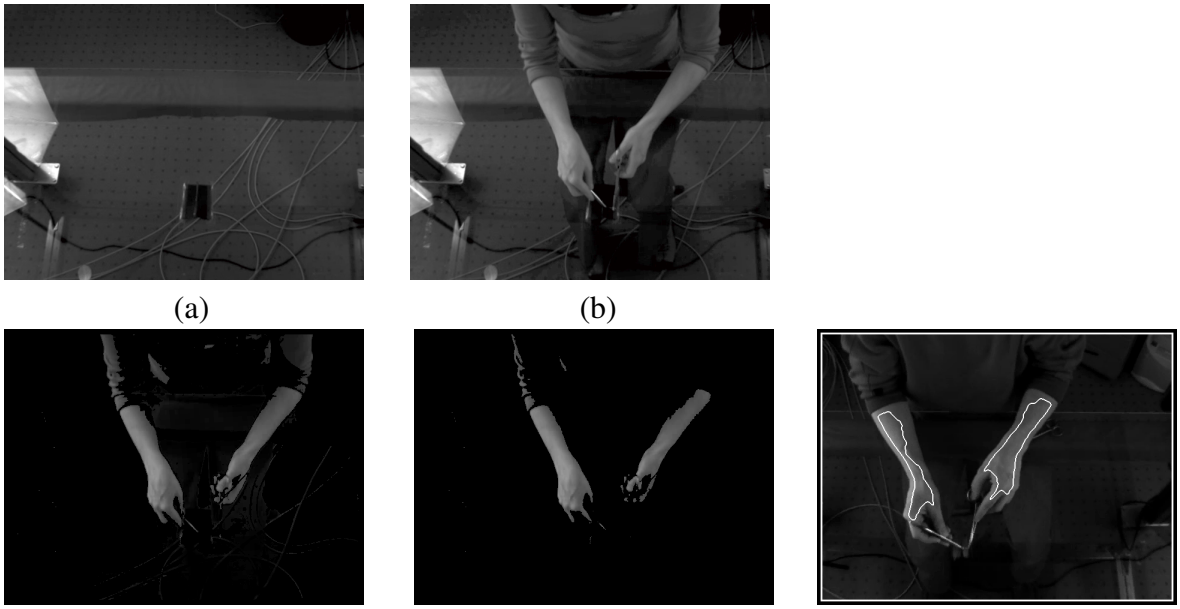


Figure 7.11.: The steps of the marker free hand motion measurement method (see text for details).

to 0 if the absolute value of the difference is smaller than 10. Fig. 7.12(c) shows a typical result for this operation. We see that most of the hands' surfaces are preserved in the background-subtracted image, but so are parts of the sweater. To retain only the lightest patches which occur on the hand, the image is segmented with an adaptive threshold. To compute the threshold, we assume the existence of two classes of pixels in the image and and determine the value that separates the two classes such that the intraclass variance is minimal.



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

Figure 7.12.: Background subtraction and threshold segmentation as means of obtaining initial contours: (a) background image, (b) original image, (c) background subtraction result or difference image, (d) adaptive threshold segmentation of the difference image. (e) Typical initial contours for segmenting hands in a video frame.

The result of adaptive threshold segmentation on image 7.12(c) is shown in Fig. 7.12(d). Only the largest two connected components are retained from the binarized thresholded image. Additional iterated erosion ensures that initial contours are well inside the hand surface and no other objects have been wrongfully enclosed in the contours.

In the first frame, an initial rectangular contour slightly smaller than the image is evolved in the binary image toward the two retained patches (the speed at a contour pixel is zero only if the pixel is white). Exemplary start rectangular contour and found initial contours are shown in Fig. 7.12(e). The final hand contours in this frame are evolved according to the threshold segmented difference between the subsequent frame and the background to initialize the segmentation in the subsequent frame.

In each frame, initial contours are evolved toward the boundaries of the hands using the motion driven by local region descriptors plus balloon force, as described in Section 4.2.2 by the equation of motion (4.3). The regions occupied by palms holding instruments are difficult to segment accurately in all frames. Segmentations can be improved by subsequently evolving the curve according to the maximum likelihood equation (4.2), since the contours are already in the vicinity of hand boundaries. The implementation is the one described in Sections 4.3 and 4.4.2. Each of the videos recorded by the top cameras is segmented independently. Fig. 7.13 shows segmented frames from the top stereo camera pair at two different time steps (one row shows one time

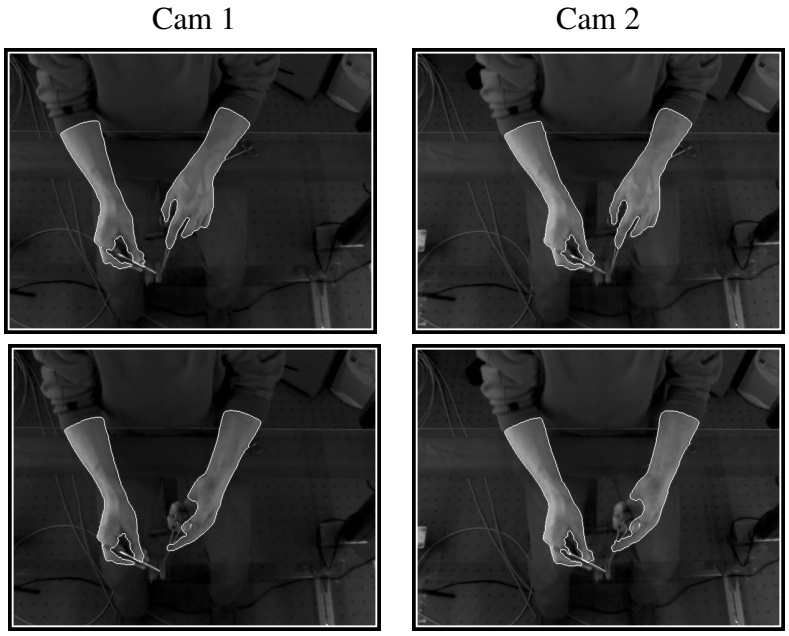


Figure 7.13.: Accurately tracked hand contours in the top cameras: images from two different time steps (one row shows one time step).

step). The frames are taken from a video recorded while a suturing procedure was executed by a visceral surgeon.

7.4.2. Anatomical features for markerless tracking

Hand motion can be measured if we track the position of landmarks on the hand. We define *anatomical features* as follows: the middle of the forearm near the elbow, the middle of the wrist and the center of gravity of the palm. These features have the important advantage that they stay visible in all cameras as hands rotate during the procedure. However, to ensure that both cameras in a stereo pair detect approximately the same anatomical features, the distance between cameras must be rather small in order to provide very similar views of the scene. Since images are rectified to have parallel horizontal epipolar lines, corresponding anatomical features should be found on the same image line.

Looking at a hand and forearm we observe that the forearm near the elbow is widest and the wrist is narrowest in the direction of the bones (radius and ulna). The orientation of hand and forearm is determined by computing the ellipse that has the same second moments as the segmented hand, as shown in Fig. 7.14. We consider the orientation of this ellipse to be the orientation of the hand. The pixel coordinates of the forearm's middle are then at the middle of the perpendicular to this orientation, where the hand is widest. The middle of the wrist is at the middle of the perpendicular where the hand is at its narrowest. The palm center is computed as the center of

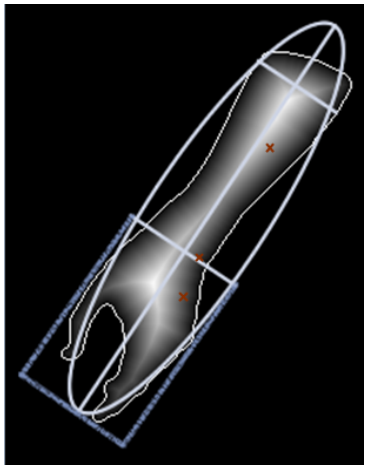


Figure 7.14.: Detection of anatomical features.

gravity of the palm and fingers.

To approximate the width of the segmented hand, the distance transform of the segmented contour is computed as illustrated in Fig. 7.14. The pixels with highest and the lowest values in the distance transform will be anatomical feature candidates. Starting from the forearm we traverse the distance transform in the computed orientation while pixel values increase. The middle of the forearm is found when values start decreasing. Continuing along this direction, the middle of the wrist is found when values start increasing again. In some particular frames, this may be past the actual wrist, and we thus limit the maximum length of the forearm to 180px.

One must note that the positions of anatomical features can be reliably detected only if the segmented hand contours are accurately tracked, since in many frames the difference in width between forearm and wrist are very small. Anatomical features detected at two time steps on the hands of a vascular surgeon during the suturing procedure are shown in Fig. 7.15 (one row shows one time step). The position of the detected anatomical features in space is finally triangulated knowing the system's calibration matrices.

7.5. Experiments and results

A visceral surgeon, a vascular surgeon, and two trainees have executed suturing procedures while being recorded with the computer vision system presented in this chapter.

Parameters for the right and left hand were set slightly differently to obtain optimal tracking accuracy. For both hands the smoothing Gaussian kernel was 5×5 pixels (standard deviation of 1.5) (as in 4.4.2). For the right hand the window size of the local region descriptor was 11px and

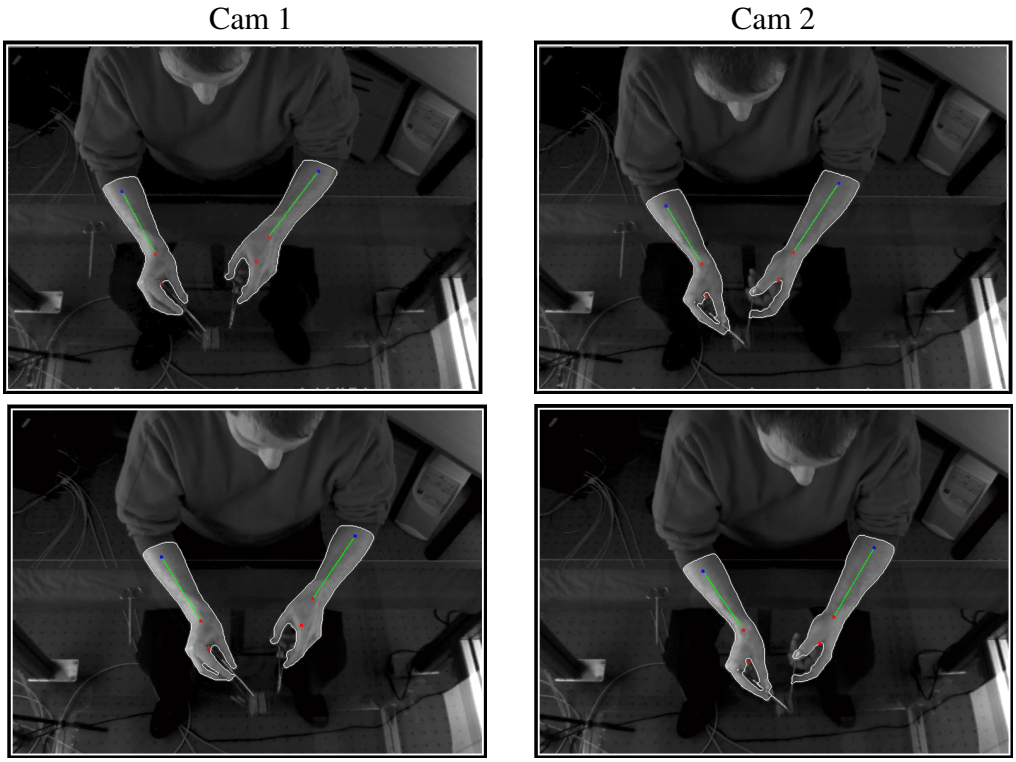
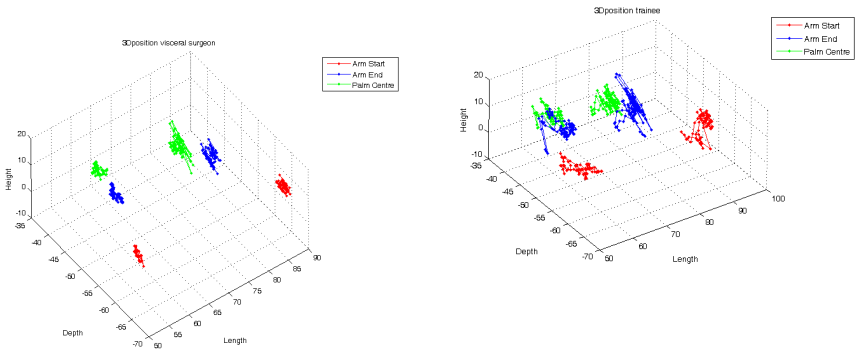


Figure 7.15.: Detected anatomical features at two time steps of top video streams (one row shows one time step).

the balloon force was $\lambda = 5$ (as in 4.4.2). Although acceptable results were obtained for the left hand using these parameters for most frames, for some other the contours leaked. This problem could be solved while improving the accuracy of the segmentation by setting the balloon force to $\lambda = 7$ and a window size of 15px. The need for different parameters is caused by different illumination of the two hands, due to the fact that the rack was stationed perpendicularly to a window and the photo umbrellas could not compensate for this.

Multiple trials were recorded for each system user. The first 100 frames of each video stream were processed to detect the contours of each hand. Only a few frames per video showed partly incorrect arm boundaries. The accurate detection of fingers is more problematic due to strongly shadowed regions, making the center of gravity of the palm a little less reliable than the other two anatomical features. Hand contours and anatomical features were displayed in the original video frame as illustrated in Fig. 7.15 to enable the experimenter to visually verify the results. This experimenter mostly agreed with the results.

The triangulated 3D positions of anatomical features have been plotted for each subject. The results for the two surgeons and one trainee are presented in a comparative fashion. Fig. 7.16 shows in 3D plots the positions of anatomical features for the visceral surgeon on the left and the



Vascular surgeon Trainee

Figure 7.16.: Plots showing the 3D positions of anatomical features in the working space.

trainee on the right. The middle of the forearm is shown in red, the middle of the wrist in blue and the center of gravity of the palm in green. Unsurprisingly, the plot shows that the trainee has moved his hands more than the surgeon - the points in the plot on the left are much more scattered.

The 3D scatter plot gives a general impression about the distance traveled by the hands, but it does not allow us to analyze the details of the motion. For this purpose, one can plot the XYZ coordinates of each measured point independently as in Fig. 7.17. The figure shows one of the recorded procedures executed by the visceral surgeon on the left and the similar data for the vascular surgeon on the right. Plotting each spatial coordinate against time yields different curves for the two surgeons. Comparing the X- and Y-coordinates, it can be observed that the vascular surgeon has more ample motions at the beginning and end of the procedure, but very steady motions for both hands in between. The core of the procedure is performed by the visceral surgeon with more ample motions than those of the vascular surgeon. One may be tempted to assume that the amplitude of motion skilled surgeons is proportional to their object of work. Since our sample consists of only two observations, we should refrain from drawing such conclusions. We may observe that the Z-coordinate plots show many jumps. This is due to the geometry of the setup in which the top cameras are oriented almost parallel to the define XY-plane.

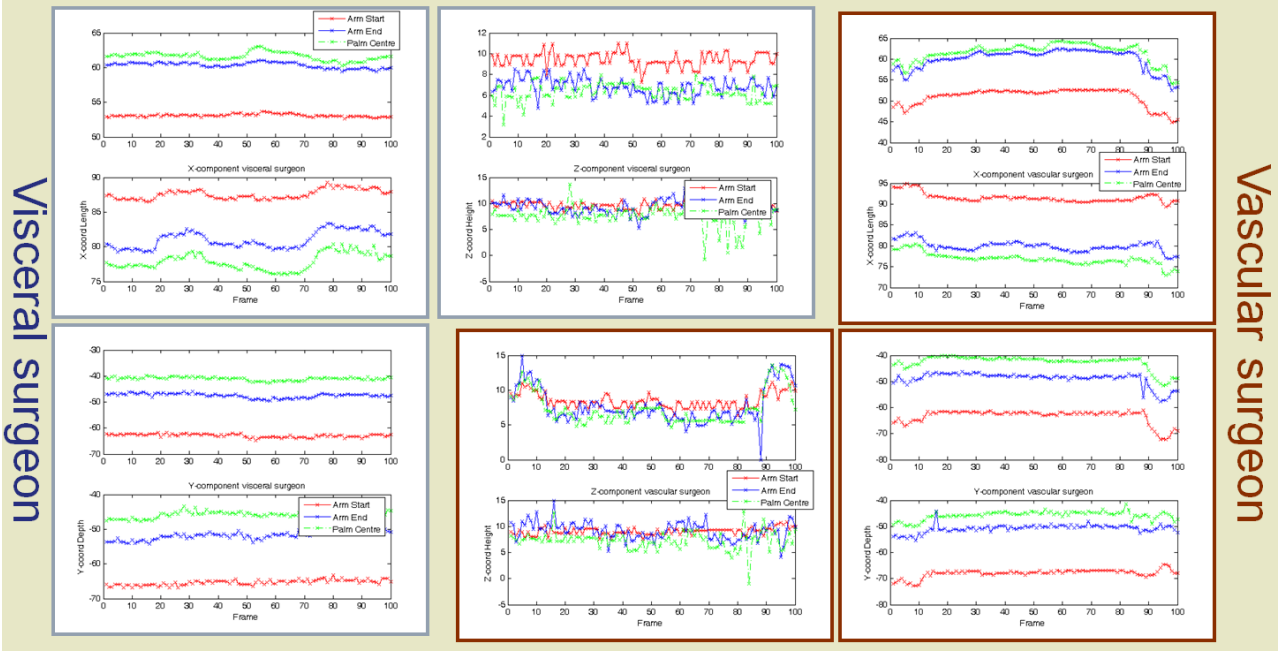


Figure 7.17.: Plots showing the individual coordinates of anatomical features for the visceral surgeon (on the left) and vascular surgeon (on the right). From left to right and top to bottom: X-coordinate visceral, Z-coordinate visceral, X-coordinate vascular, Y-coordinate visceral, Z-coordinate vascular, Y-coordinate vascular against frame number. Each plot shows the coordinates for both left and right hands.

8. Conclusion

I was born not knowing, and I have had very little time to change that here and there.

Richard Feynman

Computer vision is a very challenging research field with applications in industry, surveillance, entertainment, medical care, research in biology or archeology, and many others [1]. The goal of researchers is to create methods that accomplish vision tasks as robustly as human vision, but the latter generally outperform the proposed vision algorithms. Take for example the basic edge detection task: human vision is still the best edge detector there is. There is thus much room for improvement in computer vision methods.

The contributions of this thesis are in the field of image segmentation using region-based active contours. We have introduced two novel ideas in this field. First, we have proposed that contour evolution should be driven by local statistics instead of global ones. Specifically, local statistics have been incorporated into the concept of *local region descriptors* employed to construct energies associated with active contours. Since these energies have many local minima, along with local region descriptors we have introduced *segmenter functions* to reduce the number of local minima by assuming that image features are locally normally distributed. The properties of the proposed concept have been determined and analyzed in numerous image segmentation examples.

Second, we have proposed to construct energies from *spatially-variant definitions of energy terms*. This is equivalent to a mechanism for selecting the type of force that acts upon the contour at each point. The idea was exemplified for multiple contours where forces were defined assuming that features are locally normally distributed, or that their local distribution is best modeled with kernel density estimators.

The multiple contours are initialized requiring minimal user input - a few clicks on the objects to be segmented. Important user knowledge about image regions can thus be input into the method effortlessly and very fast. The proposed force selection algorithm was robust in various gray and color image segmentation experiments. Combining the two proposed ideas with the semi-automatic initialization we obtain a powerful, flexible framework for image segmentation which

[1] It might be shorter to list all the domains where computer vision cannot be applied, than to list all domains where it can

has been demonstrated in two applications.

First, we have segmented complex natural images using texture information. Texture was described by estimating the nonparametric statistics of image patches modeled as Markov Random Fields. This type of modeling has only recently been applied to texture segmentation and was proven to be extremely time intensive (Awate et al., 2006; Wolf et al., 2006). Due to the local nature of the proposed framework, segmentation time can be considerably shortened. In our experiments, the results were obtained two orders of magnitude faster than with global methods.

Second, the ideas proposed in Section 4 have been employed to track the hands of surgeons in stereo system setup. The advantage of the proposed system is that hand motion can be measured without using markers. Instead, anatomical landmarks, like the middle of the arm and wrist, are tracked. This helps saving time and eliminating the problem of visual markers that go out of the sight of a camera.

Both definitions of local region descriptors and of spatially-variant forces are dependent on manually selected parameters which are the sizes of local windows. In our experiments we have found good values for these parameters. This proves that our approach can be successful in many situations, however it does not guarantee it. In our experiments we slightly varied some of the parameters (the smoothness force, the balloon force and the sizes of windows) to improve the results of segmentation for different images. The empirically determined parameter values we have experimented with will probably not be appropriate for all possible images. A very important direction for future work is thus to determine the window sizes from the image data. This will raise difficult issues concerning the criteria employed for determining parameters, as well as computation time.

The proposed methods yield the same results for various initializations, but not all initializations will lead to correct segmentation results. One must also consider that our initializations contain user knowledge. However, computer vision methods should ultimately be fully automatic. Active contours based on global region descriptors have the advantage that they can be initialized fully automatically. It seems logical to combine local and global region descriptors to evolve the active contour. Global information could also be used in validating the values of computed parameters. Since initialization and parameter estimation are very difficult issues, the method will probably have to be more than a linear combination of global and local energy terms.

Another direction of research is the implementation of the proposed methods in 3D. Local feature statistics are computed from values, irrespective of the dimension - 2D or 3D - of the object described by the features. The fast level-set implementation is easily extended to 3D (as stated by the author of the fast level-set method in our personal correspondence). This direction of research is thus fairly straight forward to explore.

Considering the directions for future exploration, and the results presented in this thesis, one may conclude that the introduced local methods are a novel, promising approach in the difficult field on image segmentation, with state-of-the-art results in concrete applications.

A. Appendix I

In this appendix, we give a solution for finding the minimum of the following energy:

$$E(\Phi, p) = -\iint_R \Big(H(\Phi) g(f, \theta^W) - \mu |\nabla H(\Phi)| \Big) dxdy. \tag{A.1}$$

To find the optimum Φ, the Euler-Lagrange equations for the level set function must be derived. For this purpose, it is common to assume that the parameters of g do not depend on Φ. To derive the variation of Φ, consider replacing Φ by $\Phi + \varepsilon\psi$ where ε is a very small number. Since E is minimized by Φ, $\frac{\partial E(\Phi+\varepsilon\psi)}{\partial \varepsilon} = 0$ for $\varepsilon = 0$. Because of $H'(x) = \delta(x)$, we have $|\nabla H(\Phi)| = \delta(\Phi)|\nabla\Phi|$. According to the chain rule, this partial derivative can be written by simultaneously substituting $\varepsilon = 0$ (and thus obtaining $\frac{\partial E}{\partial \Phi}$):

$$\Big(\frac{\partial E}{\partial \Phi}, \psi \Big) = -\iint_R g\delta(\Phi)\psi - \mu \Big(\delta'(\Phi)|\nabla\Phi|\psi + \delta(\Phi) \frac{\partial |\nabla(\Phi + \varepsilon\psi)|}{\partial \varepsilon} \Big) dxdy \tag{A.2}$$

The last partial derivative is written by substituting $\varepsilon = 0$:

$$\begin{aligned} \frac{\partial |\nabla(\Phi + \varepsilon\psi)|}{\partial \varepsilon} &= \frac{1}{2|\nabla(\Phi + \varepsilon\psi)|} + \frac{\partial}{\partial \varepsilon} \Big((\Phi_x + \varepsilon\psi_x)^2 + (\Phi_y + \varepsilon\psi_y)^2 \Big) \\ &= \frac{1}{2|\nabla\Phi|} 2 \left(\Phi_x \psi_x + \Phi_y \psi_y \right) = \frac{\nabla\Phi\nabla\psi}{|\nabla\Phi|}. \end{aligned} \tag{A.3}$$

By plugging this term into (A.2) and integrating it by parts we obtain:

$$\begin{aligned} \Big(\frac{\partial E}{\partial \Phi}, \psi \Big) &= -\iint_R g\delta(\Phi)\psi - \mu \Big(\delta'(\Phi)|\nabla\Phi|\psi + \delta(\Phi) \frac{\nabla\Phi\nabla\psi}{|\nabla\Phi|} \Big) dxdy \\ = -\iint_R g\delta(\Phi)\psi - \mu \Big(\delta'(\Phi)|\nabla\Phi|\psi - div(\delta(\Phi) \frac{\nabla\Phi}{|\nabla\Phi|}) \psi \Big) + \int_{\partial R} \frac{\delta(\Phi)}{|\nabla\Phi|} \nabla\Phi n \psi ds \end{aligned} \tag{A.4}$$

where n is the outward normal vector to ∂R. We can write

$$div(\delta(\Phi)\frac{\nabla\phi}{|\nabla\phi|}) = \frac{\partial}{\partial x}\left(\delta(\Phi)\frac{\Phi_x}{|\nabla\Phi|}\right) + \frac{\partial}{\partial y}\left(\delta(\Phi)\frac{\Phi_y}{|\nabla\Phi|}\right) \tag{A.5}$$

$$= \delta'(\Phi)\frac{\Phi_x^2}{|\nabla\Phi|} + \delta(\Phi)\frac{\partial}{\partial x}\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right) + \delta'(\Phi)\frac{\Phi_y^2}{|\nabla\Phi|} + \delta(\Phi)\frac{\partial}{\partial y}\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right) \tag{A.6}$$

$$= \delta'(\Phi)|\nabla\Phi| + \delta(\Phi)div\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right). \tag{A.7}$$

By substituting (A.7) in (A.5)

$$\left(\frac{\partial E}{\partial\Phi}, \psi\right) = -\iint_R \left(g\delta(\Phi) + \mu\delta(\Phi)div\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)\right)\psi dxdy + \mu\int_{\partial R}\frac{\delta(\Phi)}{|\nabla\Phi|}\frac{\partial\Phi}{\partial n}\psi ds. \tag{A.8}$$

This partial derivative must be zero for all ψ, and thus we must have $\delta(\Phi)(g+\mu k) = 0$, where we denote by k the term $div\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)$. The gradient descent in time for Φ can then be written:

$$\Phi_t = H'(\Phi)(g+\mu k), \tag{A.9}$$

with $\Phi(x,y,0) = \Phi_0(x,y)$, $(x,y) \in R$ and boundary conditions $\frac{\delta(\Phi)}{\nabla\Phi}\frac{\partial\Phi}{\partial n} = 0$ on ∂R.

B. Appendix II

The normal and curvature of a curve $C(x,y)$ can be expressed in term of the level-set function Φ. In this appendix we demonstrate the equivalences.

The zero level-set of Φ is the implicit representation of $C(x(s),y(s))$, and thus $\Phi(C(s),t) = 0$. Taking this expression's derivative with respect to s, we have

$$\frac{\partial \Phi}{\partial s} = 0 \text{ on C } \Leftrightarrow \Phi_{\mathrm{x}} \cdot \mathrm{x_s} + \Phi_{\mathrm{y}} \cdot \mathrm{y_s} = 0. \tag{B.1}$$

We have used the chain rule for the equivalence. The equation above means that $\nabla\Phi$ is perpendicular to (x_s,y_s), which is the tangent to C. Taking the level-set function to be positive inside the curve and negative outside, the outward normal is $-\frac{\nabla\Phi}{|\nabla\Phi|}$.

The outward normal $\vec{\mathbf{n}}$ is also written as $-(-y_s,x_s)$, thus

$$-(-y_s,x_s) = -\left(\frac{\Phi_x}{|\nabla\Phi|},\frac{\Phi_y}{|\nabla\Phi|}\right). \tag{B.2}$$

The formula for the curvature is derived starting from the signed curvature's definition using the derivative of the tangent

$$C_{ss} = (x_{ss},y_{ss}) = k \cdot \vec{\mathbf{n}}. \tag{B.3}$$

Since $\frac{\partial\Phi}{\partial s}$ is zero on C, $\frac{\partial^2\Phi}{\partial s^2} = 0$. Let us denote the dot product of two vectors $< a,b >$. The equality is written out using the chain rule

$$\frac{\partial}{\partial s}(\Phi_x x_s + \Phi_y y_s) = \Phi_{xx} \cdot x_s^2 + \Phi_{xy} \cdot x_s y_s + \Phi_{yy} \cdot y_s^2 + < \nabla\Phi, C_{ss} > . \tag{B.4}$$

One can further write by replacing (B.2) and (B.3) in (B.4)

$$(\Phi_{xx} \cdot x_s + \Phi_{xy} \cdot y_s)x_s + (\Phi_{xy} \cdot x_s + \Phi_{yy} \cdot y_s)y_s = k < \nabla\Phi, -\frac{\nabla\Phi}{|\nabla\Phi|} > \tag{B.5}$$

$$\left(\Phi_{xx}\frac{\Phi_y}{|\nabla\Phi|} - \Phi_{xy}\frac{\Phi_x}{|\nabla\Phi|}\right)\frac{\Phi_y}{|\nabla\Phi|} + \left(-\Phi_{xy}\frac{\Phi_y}{|\nabla\Phi|} + \Phi_{yy}\frac{\Phi_x}{|\nabla\Phi|}\right)\frac{\Phi_x}{|\nabla\Phi|} = -k \cdot |\nabla\Phi|. \tag{B.6}$$

The term $|\nabla\Phi|$ can be factored out on the left, such that finally

$$\frac{\Phi_{xx}\Phi_y^2 - 2\Phi_{xy}\Phi_x\Phi_y + \Phi_{yy}\Phi_x^2}{(\Phi_x^2+\Phi_y^2)^{3/2}}|\nabla\Phi| = -k\cdot|\nabla\Phi|, \tag{B.7}$$

and thus

$$k = -\frac{\Phi_{xx}\Phi_y^2 - 2\Phi_{xy}\Phi_x\Phi_y + \Phi_{yy}\Phi_x^2}{(\Phi_x^2+\Phi_y^2)^{3/2}} = -div(\frac{\nabla\Phi}{|\nabla\Phi|}). \tag{B.8}$$

C. From Formulas to Implementation

This appendix lists the steps from the curve evolution formula to its implementation.

Take the following curve evolution for the curve represented by the level-set function ϕ_i:

$$\frac{\partial \phi_i}{\partial t} = H'(\phi_i) \left(\underbrace{\log p_i - \max_{j \neq i, H(\phi_j)>0} (\log p_j)}_{Statistics forces} + \underbrace{\frac{\mu}{2} \left(div \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \right)}_{Smoothness force} \right). \tag{C.1}$$

To implement this equation, one must first initialize the contour. The initial position of the contour is used to initialize the level-set function ϕ_i, usually computing the distance transform of the binary image of the contour. The level-set function ϕ_i is negative on the inside, positive on the outside of the contour, and zero on the contour.

To modify the position of the contour, one updates the level-set function, according to the force that acts upon it, composed of statistics and smoothness force. When the force is positive, pos-

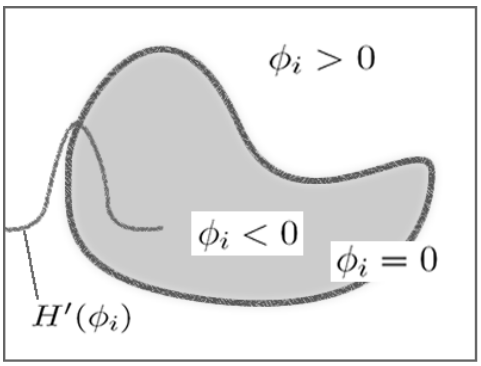


Figure C.1.: Level-set function and the derivative of the Heaviside function. The level-set function is negative inside, positive outside on the outside of the contour and zero on the contour. The Heaviside function decreases to zero, thus the level-set function is modified only around the zero level-set ($\phi_i = 0$).

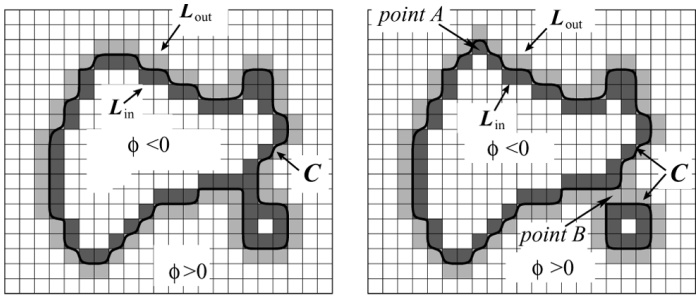


Figure C.2.: Level-set function and the derivative of the Heaviside function. The level-set function is negative inside, positive outside on the outside of the contour, and zero on the contour. The Heaviside function decreases to zero, thus the level-set function is modified only around the zero level-set ($\phi_i = 0$).

itive values are added to ϕ_i. This happens at positions where $H(\phi_i)$ is strictly positive, which is around the zero level-set. The curve is thus moved inwards.

In the fast level-set implementation, the level-set function ϕ_i is a piecewise constant function. This function is updated by computing only the sign of the data and smoothness forces that act upon the contour. The contour is also represented by the two lists, the list of inner points in dark gray and the list of outer points in light gray. The sign of the forces is computed for each point in these lists. By convention, if the speed is negative at an inner list point, the curve is moved inward, as is shown, for example, at point B. At point A in the outer list the speed is positive and the curve is moved outward.

D. Formula Glossary

I	The image to be segmented
$(x,y) = \mathbf{x}$	The position of an image pixel
R	The image domain (the domain of indexes of pixels x and y)
R_i	A region in the image, delimited by an active contour
N	The total number of image regions, thus $i = 1..N$
C, C_i	An active contours
L	The length of the active contour
C_{ij}	A piece of an active contour lying between regions R_i and R_j
$\vec{n}$, $\vec{n}_i$	The normal to the contour
k, k_i	The curvature of the contour
$u(\mathbf{x})$	A smooth approximation of the image I
$\int_C ds$	Smoothness term of the curve C, parametrized by arc-length s
$\mathbf{f}(\mathbf{x})$	Value of (computed) feature at pixel $\mathbf{x}$
p_i	The pdf that describes the variability of a feature (vector) in region R_i
θ_i	Parameters of the pdf p_i
$\iint_{R_i} \log p_i d\mathbf{x}$	Data term
t	Artificial time variable for curve evolutions
ϕ_i	Level-set function representing curve C_i
$H(x)$	The Heaviside function
$\lvert\nabla\phi\rvert$	The normal of the zero level-set (the curve) in terms of the level-set function
$div\left(\frac{\nabla\phi}{\lvert\nabla\phi\rvert}\right)$	The curvature in terms of the level-set function

Bibliography

Abe, K., Saito, H., and Ozawa, S. (2000). 3d drawing system via hand motion recognition from two cameras. In *Proc. IEEE International Conference on Systems, Man and Cybernetics*.

Adalsteinsson, D. and Sethian, J. (1995). A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118:269–277.

Aubert, G., Barlaud, M., Faugeras, O., and Jehan-Besson, S. (2003). Image segmentation using active contours: calculus of variations or shape gradients. *SIAM J. of Applied Mathematics*, 63(6):2128Ű2154.

Awate, S. P., Tasdizen, T., and Whitaker, R. T. (2006). Unsupervised texture segmentation with nonparametric neighborhood statistics. In *European Conf. Comp. Vis.*

Barczak, A. L. C., Dadgostar, F., and Johnson, M. (2005). Real-time hand tracking using the viola and jones method. In *Proc. International IEEE Conference on Image and Signal Processing*.

Barth, E., Beard, B. L., and Jr., A. J. A. (1999). Nonlinear features in vernier acuity. In *Proc. of SPIE, Human Vision and Electronic Imaging IV*, volume 3644.

Bear (2008). http://www.eecs.berkeley.edu/research/projects/cs/vision/grouping/segbench.

Berkley (2008). http://www.eecs.berkeley.edu/research/projects/cs/vision/ grouping/segbench/.

Bertalmio, M., Vese, L., Sapiro, G., and Osher., S. (2003). Simultaneous structure and texture image inpainting. *IEEE Trans. Image Proc.*, 12:882–889.

Besag, J. (1974). Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 36(2):192–236.

Besson, S. J., Barlaud, M., and Aubert, G. (2000). Detection and tracking of moving objects using a new level set based method. In *Proc. Int. Conf. Patt. Recog.*

Bigün, J., Granlund, G. H., and Wiklund, J. (1991). Multidimensional orientation estimation with applications to texture analysis and optical flow. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(8):775–790.

Blake, A. and Isard, M. (1998). *Active Contours: The Application of Techniques from Graphics, Vision, Control Theory and Statistics to Visual Tracking of Shapes in Motion*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Bolt, R. A. (1980). Put-that-there: Voice and gesture at the graphics interface. In *Proc. of Internat. Conference on Computer Graphics and Interactive Techniques*.

Bouman, C. and Sauer, K. (1993). A generalized gaussian image model for edge-preserving map estimation. *IEEE Trans. on Image Processing*, 2(3):296 – 310.

Boykov, Y., Veksler, O., and Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern An. Machine Intell.*, 23(11):1222Ű1239.

Bresson, X., Esedoglu, S., Vandergheynst, P., Thiran, J.-P., and Osher, S. (2007). Fast global minimization of the active contour/snake model. *Journal of Mathematical Imaging and Vision*, 28(2):99–190.

Bretzner, L., Laptev, I., and Lindeberg, T. (2002). Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering. In *Face and Gesture*.

Brigger, P., Hoeg, J., and Unser, M. (2000). B-spline snakes: A flexible tool for parametric contour detection. *IEEE Trans. on Image Processing*, 9(9):1484–1496.

Brox, T. and Cremers, D. (2007). On the statistical interpretation of the piecewise smooth mumford-shah functional. In *Proc. of Internat. Conf. on Scale Space and Variational Methods in Computer Vision*.

Brox, T., Rousson, M., Deriche, R., and Weickert, J. (2003a). Unsupervised segmentation incorporating colour, texture and motion. In *Computer An. of Images and Patterns.*

Brox, T., Rousson, M., Deriche, R., and Weickert, J. (2003b). Unsupervised segmentation incorporating colour, texture, and motion. Technical report, INRIA, Sophia Antipolis.

Brox, T. and Weickert, J. (2004a). Level set based image segmentation with multiple regions. In Rasmussen, C.-E., Bulthoff, H., Giese, M., and Scholkopf, B., editors, *Pattern Recognition, Springer LNCS 3175*.

Brox, T. and Weickert, J. (2004b). A tv flow based local scale measure for texture discrimination. In *Proc. European Conf. Comp. Vis.*

Caselles, V., Catté, F., Coll, T., and Dibos, F. (1993). A geometric model for active contours in image processing. *Numerische Mathematik*, 66(1):1–31.

Caselles, V., Kimmel, R., and Sapiro, G. (1997). Geodesic active contours. *Int. J. Comput. Vision*, 22(1):61–79.

Chakraborty, A., Staib, L. H., and Duncan, J. S. (1996). Deformable boundary finding in medical images by integrating gradient and region information. *IEEE Trans. Medical Imaging*, 15(6):859–870.

Chan, T., Esedogluy, S., and Nikolova, M. (2004). Algorithms for finding global minimizers of image segmentation and denoising models. Technical report, UCLA.

Chan, T. and Vese, L. (1999). An active contour model without edges. In *Scale-Space Theories in Computer Vision*, pages 141–151.

Chan, T. and Vese, L. (2001). Active contours without edges. *IEEE Trans. Image Processing*, 10:266–277.

Chan, T. F., Esedoglu, S., and Ni, K. (2007). Histogram based segmentation using wasserstein distances. In *Scale Space and Variational Methods in Comp. Vis., LNCS 4485*.

Chang, J. S., Kim, E. Y., Jung, K., and Kim, H. J. (2005a). Object tracking using mean shift and active contours. In *Innovations in Applied Artificial Intelligence*, volume 3533/2005 of *LNCS*, pages 26–35.

Chang, W.-Y., Chen, C.-S., and Hung, Y.-P. (2005b). Appearance-guided particle filtering for articulated hand tracking. In *Proc. IEEE Conf Computer Vision Pattern Recognition*.

Chen, Y., Tagare, H., Thiruvenkadam, S., Huang, F., Wilson, D., Gopinath, K. S., Briggs, R. W., and Geiser, E. (2002). Using shape priors in geometric active contours in a variational framework. *Int. J. of Computer Vision*, 50(3):315Ű328.

Chesnaud, C., Refregier, P., and Boulet, V. (1999). Statistical region snake-based segmentation adapted to differentphysical noise models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1145–1157.

Clark, A. A. and Thomas, B. T. (2001). Evolving image segmentations for the analysis of video sequences. In *Proc. Computer Vision and Pattern Recognition*, pages 290–295.

Cohen, L., Bardinet, E., and Ayache, N. (1993). Surface reconstruction using active contour models. In *In Proceedings SPIE 93 Conference on Geometric Methods in Computer Vision*.

Cohen, L. and Cohen, I. (1993). Finite-element methods for active contour models and balloons for 2-d and 3-d images. *IEEE Tranactions on pattern analysis and Machine Intelligence*, 15(11):1131–1147.

Cohen, L. D. (1991). On active contour models and balloons. *CVGIP: Image Underst.*, 53(2):211–218.

Comaniciu, D. (2003). An algorithm for data-driven bandwidth selection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(5):281–288.

Comaniciu, D., Ramesh, V., and Meer, P. (2001). The variable bandwidth mean shift and data-driven scale selection. In *IEEE Int. Conf. Comp. Vis.*

Cootes, T. and Taylor, C. (1992). Active shape models- smart snakes. In *Proc. British Machine Vision Conference, 1992, p266, Springer Verlag,1992.*

Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J. (1995). Active shape models – their training and application. *Comput. Vis. Image Underst.*, 61(1):38–59.

Cows (2008). url=http://boogieland.no.

Cremers, D. (2008). Nonlinear dynamical shape priors for level set segmentation. *Journal of Scientific Computing*.

Cremers, D. and Rousson, M. (2007). Efficient kernel density estimation of shape and intensity priors for level set segmentation. In Suri, J. S. and Farag, A., editors, *Parametric and Geometric Deformable Models: An application in Biomaterials and Medical Imagery*. Springer.

Cremers, D., Rousson, M., and Deriche, R. (2007). A review of statistical approaches to level set segmentation: Integrating color, texture, motion and shape. *Int. J. Comput. Vision*, 72(2):195–215.

Cremers, D., Tischhäuser, F., Weickert, J., and Schnörr, C. (2002). Diffusion snakes: Introducing statistical shape knowledge into the mumford-shah functional. *International Journal of Computer Vision*, 50(3):295 – 313.

Darolti, C., Bodensteiner, C., Barth, E., and Hofmann, U. (2008a). Active contours that grow and compete driven by local region descriptors. *Proc. of SPIE Electronic Imaging*.

Darolti, C., Bodensteiner, C., Mertins, A., and Hofmann, U. (2008b). Local region descriptors for active contours. *IEEE Trans. Image Processing*, 17(12).

Darolti, C., Bouchard, R., Farke, S., Condurache, A., and Hofmann, U. (2007). Measurement of surgical motion with a marker free computer vision based system. *International Journal of Computer Assisted Radiology and Surgery*, 2, S1:S213.

Datta, V., Mackay, S., Darzi, A., and Gillies, D. (2001). Motion analysis in the assessment of surgical skill. *Computer Methods in Biomechanics and Biomedical engineering*, 4:515–523.

de Bonet, J. S. and Viola, P. (1998). Texture recognition using a non-parametric multi-scale statistical model. In *IEEE Conf. Comp. Vis. and Pattern Recog.*

de la Hamette, P. and Troester, G. (2008). Architecture and applications of the fingermouse: a smart stereo camera for wearable computing hci. *Personal and Ubiquitous Computing*, 12:97–110.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39:1Ű38.

Dorettoy, G., Cremersy, D., Favarozy, P., and Soatto, S. (2003). Dynamic texture segmentation. In *IEEE Int. Conf. Comp. Vis.*

Duda, R., Hart, P., and Stork, D. (2001). *Pattern Classification*. Wiley.

Dunn, D. and Higgins, W. (1995). Optimal gabor filters for texture segmentation. *IEEE Trans. on Image Processing*, 4:947 – 964.

Ecabert and Thiran, O. (2002). Variational image segmentation by unifying region and boundary information. In *16th International Conference on Pattern Recognition*.

Efros, A. A. and Leung, T. K. (1999). Texture synthesis by non-parametric sampling. In *Int. Conf. on Comp. Vision*.

Esedoglu, S., Ruuth, S., and Tsai., R. (2005). Threshold dynamics for shape reconstruction and disocclusion. In *Proc. Int. Conf. Image Processing*, page 502Ű505.

Esedoglu, S. and Tsai, Y.-H. (2006). Threshold dynamics for the piecewise constant mumford-shah functional. *Journal of Computational Physics*, 211:367–384.

Fedkiw (2008). http://physbam.stanford.edu/ fedkiw/.

ffmpeg (2005). http://ffmpeg.mplayerhq.hu/.

Firefox (2008). url=http://googlesightseeing.com/maps?p=&c=&t=k&hl=en&ll=45.123478,-123.113635&z=16).

Flickr (2008). http://www.flickr.com.

Forsyth, D. and Ponce, J. (2003). *Computer Vision: A Modern Approach*. Pearson Studium Prentice Hall.

Freedman, D. and Zhang, T. (April 2004). Active contours for tracking distributions. *Image Processing, IEEE Transactions on*, 13(4):518– 526.

Freeman, W. T. and Adelson, E. H. (1991). The design and use of steerable filters. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(9):891–906.

Fusiello, A., Trucco, E., and Verri, A. (2000). A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12(1):16–22.

Gardner, W. A. (1990). *Introduction to Random Processes with Application to Signals and Systems*. McGraw-Hill.

Gastaud, M., Barlaud, M., and Aubert, G. (2004). Combining shape prior and statistical features for active contour segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(5):726 – 734.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6:721–741.

Geman, S. and Graffigne, C. (1986). Markov random field image models and their applications to computer vision. In *In Proc. Int. Congress of Mathematicians*, pages 1496–1517.

Georgescu, B., Shimshoni, I., and Meer, P. (2003). Mean shift based clustering in high dimensions: A texture classification example. In *IEEE Int. Conf. Comp. Vis.*

Gibou, F. and Fedkiw, R. (2005). A fast hybrid k-means level set algorithm for segmentation. In *Proceedings of the 4th Annual Hawaii Int'l Conf. Statistics and Mathematics*, pages 281–291.

Gillbert, C. A. (1929). All is vanity.

Goldenberg, R., Kimmel, R., Rivlin, E., and Rudzsky, M. (2001). Fast geodesic active contours. *IEEE Trans. on Image Processing*, 10:1467–1475.

Gonzalez, R. C. and Woods, R. E. (2002a). *Digital Image Processing*. Prentice-Hall.

Gonzalez, R. C. and Woods, R. E. (2002b). *Digital Image Processing*. Prentice Hall.

Ha, J., Alvino, C., Pryor, G., Niethammer, M., Johnson, E., and Tannenbaum, A. (2004). Active contours and optical flow for automatic tracking of flying vehicles. In *Proc. American Control Conference*, pages 3441 – 3446.

Han, B., Comaniciu, D., Zhu, Y., and Davis, L. S. (2008). Sequential kernel density approximation and its application to real-time visual tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(7):1186–1197.

Han, X., Xu, C., and Prince, J. (2003a). *Geometric Level Set Methods in Imaging, Vision, and Graphics*, chapter A topology preserving geometric deformable model and its application in brain cortical surface reconstruction. Springer Verlalg.

Han, X., Xu, C., and Prince, J. (2003b). A topology preserving level set method for gemetric deformable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6):755–768.

Hands (2008). Unknown source. please inform the author of this dissertation if you are aware of the source of these images.

Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition.

Heap, T. and Hogg, D. (1996). Towards 3d hand tracking using a deformable model. In *Proc. of IEEE Internat. Conf. Automatic Face and Gesture Recognition*, pages 140–145.

Heiler, M. and Schnoerr, C. (2003). Natural image statistics for natural image segmentation. In *IEEE Int. Conf. on Comp. Vis.*, page 1259Ű1266.

Herbulot, A., Jehan-Besson, S., Barlaud, M., and Aubert, G. (2004). Shape gradient for multimodal image segmentation using mutual information. In *Proc. Int. Conf. on Image Processing*.

Horn, B. and Schunck, B. (1981). Determining optical flow. *Artificial Intelligence*, 17:185–203.

Horvath, P. (2007). A multispectral data model for higher-order active contours and its application to tree crown extraction. In *Proc Advanced Concepts for Intelligent Vision Systems, LNCS 4678*, pages 1611–3349.

Inc, M. (2008). http://www.mentice.com/default.asp.

Isard, M. and Blake, A. (1998a). Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28.

Isard, M. and Blake, A. (1998b). Icondensation: Unifying low-level and high-level tracking in a stochastic framework. In *ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume I*, pages 893–908, London, UK. Springer-Verlag.

Jaehne, B. (1997). *Digital Image Processing. Concepts, Algorithms, and Scientific Applications*. Springer.

Jain, A. and Farrokhnia, F. (1990). Unsupervised texture segmentation using gabor filters. In *IEEE Int. Conf. on Systems, Man and Cybernetics*.

Jain, A. K. (1989). *Fundamentals of Digital Image Processing*. Prentice Hall.

Jain, R., Kasturi, R., and Schunck, B. G. (1995). *Machine Vision*. McGraw-Hill.

Jalba, A., Wilkinson, M., and Roerdink, J. (2004). Cpm: A deformable model for shape recovery and segmentation based on charged particles. *IEEE Trans. Pattern Analysis Machine Intell*, 26(10):1320Ű1335.

Jeannie (2008). http://www.kecl.ntt.co.jp/icl/signal/sawada/demo/bss2to4/index.html.

Jehan-Besson, S. and Barlaud, M. (2003). Dream2s: Deformable regions driven by an eulerian accurate minimization method for image and video segmentation. *Int. J. of Computer Vision*, 53(1):45 – –70.

Jermyn, I. H. and Ishikawa, H. (1999). Globally optimal regions and boundaries. In *IEEE International Conference on Computer Vision*.

Jones, J. and Palmer, L. (1987). An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex. *J. Neurophysiol.*, 58(6):1233–1258.

Juan, O., Keriven, R., and Postelnicu, G. (2006). Stochastic motion and the level set method in computer vision: Stochastic active contours. *Internat. J. of Computer Vision*, 69(1):7Ű25.

Julesz, B. (1962). Visual pattern discrimination. *IRE Trans. Information Theory*, page 84Ű92.

Kadir, T. and Brady., M. (2003). Unsupervised non-parametric region segmentation using level sets. In *IEEE Int. Conf. Comp. Vis*.

Kass, M., Witkin, A., and Terzopoulos, D. (January, 1988). Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331.

Katkovnik, V. and Spokoiny, V. (2008). Spatially adaptive estimation via fitted local likelihood techniques. *IEEE Transactions on Signal Processing*, 56(3):873–886.

Kichenassamy, S., Kumar, A., Olver, P. J., Tannenbaum, A., and Yezzi, A. J. (1995). Gradient flows and geometric active contour models. In *International Conference on Computer Vision*, pages 810–815.

Kim, J., Fisher, J. W., Yezzi, A. J., Cetin, M., and Willsky., A. S. (2005). A nonparametric statistical method for image segmentation using information theory and curve evolution. *IEEE Trans. Image Processing*, 14:1486 – 1502.

Kim, J.-S. and Hong, K.-S. (2007). A new graph cut-based multiple active contour algorithm without initial contours and seed points. *Machine Vision and Applications*, online first.

Kölsch, M. and Turk, M. (2004a). Analysis of rotational robustness of hand detection with viola & jones' method. In *Proc. Internat Conf. Pattern Recognition*.

Kölsch, M. and Turk, M. (2004b). Fast 2d hand tracking with flocks of features and multi-cue integration. In *IEEE Workshop on Real-Time Vision for Human-Computer Interaction*.

Kurata, T., Okuma, T., Kourogi, M., and Sakaue, K. (2001). The hand mouse: Gmm hand-color classication and mean shift tracking. In *IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*.

Lankton, S., Nain, D., Yezzi, A., and Tannenbaum, A. (2007). Hybrid geodesic region-based curve evolutions for image segmentation. In *Proc. of SPIE Medical Imaging*.

Lee, A., Pedersen, K., and Mumford, D. (2003). The nonlinear statistics of high-contrast patches in natural images. *Int. J. Computer Vision*, 54:83–103.

Li, C., Kao, C.-Y., Gore, J. C., , and Ding, Z. (2007). Implicit active contours driven by local binary fitting energy. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*.

Li, C., Liu, J., and Fox, M. (2005a). Segmentation of edge preserving gradient vector flow: an approach toward automatically initializing and splitting of snakes. In *Proc. Computer Vision and Pattern Recognition*, page 162Ű167.

Li, C., Xu, C., Gui, C., and Fox, M. D. (2005b). Level set evolution without reinitialization: A new variational formulation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*.

Liang, L., Liu, C., Xu, Y., Guo, B., , and Shum, H. (2001). Real-time texture synthesis by patch-based sampling. Technical report, MSR-TR-2001-40, Microsoft.

Licsar, A. and Sziranyi, T. (2002). Supervised training based hand gesture recognition system. In *ICPR '02: Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 3*, page 30999, Washington, DC, USA. IEEE Computer Society.

Lin, H., Shafran, I., Murphy, T. E., Okamura, A. M., Yuh, D. D., and Hager, G. D. (2005). Automatic detection and segmentation of robot-assisted surgical motions. In *8th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), vol. I, pp. 802-810*.

Lindeberg, T. (1998a). Edge detection and ridge detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):117–154.

Lindeberg, T. (1998b). Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):77–116.

Liu, F., Song, X., Luo, Y., and Hu, D. (2002). Unsupervised mumford-shah energy based hybrid of texture and nontexture image segmentation. In *Proc. Internat. Conf. Image Processing*, pages 753–756.

Lu, T., Neittaanmaki, P., and Tai, X.-C. (1991). A parallel splitting up method and its application to navierŰstokes equations. *Appl. Math. Letters*, 4(2):25Ű29.

Lucas, B. and Kanade, T. (1981). An iterative image registration technique with application to stereo vision. In *Proc. DARPA Image Understanding Workshop*, pages 121–130.

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

Mallat, S. (1999). *A Wavelet Tour of Signal Processing*. Academic Press.

Mansouri, A. and Mitiche, A. (2002). Region tracking via local statistics and level set pdes. In *Proc. Int. Conf. Image Processing*.

Mansouri, A.-R. (2002). Region tracking via level set pdes without motion computation. *IEEE Trans. Pattern Anal. Machine Intell.*, 24(7):947–961.

Marcelja, S. (1980). Mathematical description of the responses of simple cortical cells. *J.l of the Optical Society of America*, 70:1297–1300.

Martin, D. R., Fowlkes, C. C., and Jitendra Malik, M. (2004). Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(5):530–549.

McInerney, T. and Terzopoulos, D. (1996). Deformable models in medical image analysis: A survey. *Medical Image Analysis*, 1(2):91–108.

Merriman, B., Bence, J., and Osher, S. (1994). Motion of multiple junctions: a level set approach. *Journal of Computational Physics*, 112(2):334–363.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller., E. (1953). Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092.

Moeller, S., Freiwald, W. A., and Tsao, D. Y. (2008). Patches with links: a unified system for processing faces in the macaque temporal lobe. *Science*, 320(5881):1355–1359.

Mortensen, E. N. and Barrett, W. A. (1995). Intelligent scissors for image composition in computer graphics. In *Proc. SIGGRAPH*, pages 191–198.

Mumford, D. and Shah, J. (1985). Boundary detection by minimizing functionals. In *IEEE Conf. on Computer Vision and Pattern Recognition*.

Nain, D., Yezzi, A., and Turk, G. (2003). Vessel segmentation using a shape driven flow. In *Proc. of Medical Image Computing and Computer-Assisted Intervention*, page 51Ű59.

Ngan, K., Meier, T., and Chai, D. (1999). *Advanced Video Coding: Principles and Techniques, 7*. Elsevier.

Ning Xu, R. B. and Ahuja, N. (2003). Object segmentation using graph cuts based active contours. In *EEE International Conference on Computer Vision and Pattern Recognition*.

Noelker, C. and Ritter, H. (1999). Grefit: Visual recognition of hand postures. In *Proc. International Gesture Workshop, LNAI 1739*, page 61Ű72.

O'Hagan, R. and Zelinsky, A. (2000). Vision-based gesture interfaces to virtual environments. In *Proc. Australasian User Interfaces Conference*.

Oka, K., Sato, Y., and Koike, H. (2002). Real-time tracking of multiple fingertips and gesture recognition for augmented desk interface systems. In *Proc. IEEE International Conference on Automatic Face and Gesture Recognition*.

Osher, S. and Sethian, J. (1988). Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *J. of Computational Physics*, 79:12–49.

Papoulis, A. (1990). *Probability and Statistics*. Prentice Hall.

Pappas, T. (1992). An adaptive clustering algorithm for image segmentation. *IEEE Transactions on Signal Processing*, 40(4):901–914.

Paragios, N. (2000.). *Geodesic Active Regions and Level Set Methods: Contributions and Applications in Artificial Vision.* PhD thesis, PhD Thesis, Ecole Superiore en Science Informatique, Universite de Nice-Sophia Antipolis/I.N.R.I.A.,.

Paragios, N. and Deriche, R. (1999a). Geodesic active contours for supervised texture segmentation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)*, volume 2, page 2422, Los Alamitos, CA, USA. IEEE Computer Society.

Paragios, N. and Deriche, R. (1999b). Unifying boundary and region-based information for geodesic active tracking. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Paragios, N. and Deriche, R. (2000). Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280.

Paragios, N. and Deriche, R. (2002a). Geodesic active regions: A new paradigm to deal with frame partition problems in computer vision. *Journal of Visual Communication and Image Representation*, 13(1/2):249Ű268.

Paragios, N. and Deriche, R. (2002b). Geodesic active regions and level set methods for supervised texture segmentation. *Int. J. Computer Vision*, 46(3):223Ű247.

Paragios, N. and Deriche, R. (2005). Geodesic active regions and level set methods for motion estimation and tracking. *J. Computer Vision and Image Understanding*, 97(3):259–282.

Paragios, N., Mellina-Gottardo, O., and Ramesh, V. (2004). Gradient vector flow fast geometric active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):402–407.

Paragios, N., Rousson, M., and Ramesh, V. (2002). Matching distance functions: A shape to area variational approach for global to local registration. In *Proc. of European Conference in Computer Vision*.

Peng, D., Merriman, B., Zhao, H., Osher, S., and Kang, M. (1999). A pde based fast local level set method. *Journal of Computational Physics*, 155:410–438.

Pinker, S. (1999). *How The Mind Works*. W. W. Norton & Company.

Piovano, J., Rousson, M., and Papadopoulo, T. (2007). Efficient segmentation of piecewise smooth images. In *Scale Space and Variational Methods in Computer Vision*, volume LNCS 4485, pages 709–720.

Pohl, K., Fisher, J., Shenton, M., McCarley, R. W., Grimson, W., Kikinis, R., and Wells, W. (2006). Logarithm odds maps for shape representation. In *Proc. of Medical Image Computing and Computer-Assisted Intervention*, volume 4191 of *Lecture Notes in Computer Science*, pages 955–963. Springer-Verlag.

Pohl, K., Kikinis, R., and Wells, W. (2007). Active mean fields: Solving the mean field approximation in the level set framework. In *Information Processing in Medical Imaging*, volume 4584 of *Lecture Notes in Computer Science*, pages 26–37. Springer-Verlag.

Polzehl, J. and Spokoiny, V. (2003). Image denoising: pointwise adaptive approach. *Ann. Statist.*, 31:30–57.

Popat, K. and Picard, R. W. (1997). Cluster based probability model and its application to image and texture processing. *IEEE Trans. Image Processing*, 6(2):268–284.

Rabbit (2008). www.dw-world.de/dw/article/0,2144,2326455,00.html.

Regan, D. (2000). *Human Perception of Objects: Early Visual Processing of Spatial Form Defined by Luminance, Color, Texture, Motion, and Binocular Disparity*. Sinauer Associates.

Rochery, M., Jermyn, I. H., and Zerubia, J. (2003). Higher order active contours and their application to the detection of line networks in satellite imagery. In *2nd IEEE Workshop on VLSM*.

Rochery, M., Jermyn, I. H., and Zerubia, J. (2005). Phase field models and higher-order active contours. In *IEEE International Conference on Computer Vision*.

Rochery, M., Jermyn, I. H., and Zerubia, J. (2006). Higher order active contours. *Int. J. Comput. Vision*, 69(1):27–42.

Ronfard, R. (1994). Region-based strategies for active contour models. *Int. J. Computer Vision*, 13(2):229–251.

Rousson, M., Brox, T., and Deriche, R. (2003). Active unsupervised texture segmentation on a diffusion based feature space. In *IEEE Conf. Comp. Vis. and Pattern Recog.*

Rousson, M. and Deriche, R. (2002). A variational framework for active and adaptative segmentation of vector valued images. In *Proc. IEEE Workshop on Motion and Video Computing*, page 56Ű62.

Rousson, M. and Paragios, N. (2002). Shape priors for level set representations. In *Proc. of European Conference in Computer Vision*, pages 78–93.

Rousson, M. and Paragios, N. (2008). Prior knowledge, level set representations and visual grouping. *Int. J. of Computer Vision*, 76(3):231–243.

Roy, T., Barlaud, M., Debreuve, É., and Aubert, G. (2003). Vector field segmentation using active contours: Regions of vectors with the same direction. In *Workshop on Variational, Geometric, and Level Set Methods in Computer Vision (VLSM)*.

Roy, T., Debreuve, É., Barlaud, M., and Aubert, G. (2006). Segmentation of a vector field: dominant parameter and shape optimization. *Journal of Mathematical Imaging and Vision*, 24(2):259–276.

Rudin, L., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D Nonlinear Phenomena*, 60:259–268.

Sagiv, C., Sochen, N. A., and Zeevi, Y. Y. (2002). Texture segmentation via a diffusion segmentation scheme in the gabor feature space. In *Workshop on Texture Analysis and Synthesis*.

Samson, C., Blanc-Feraud, L., Aubert, G., and Zerubia, J. (1999). A level set model for image classification. In *Scale-Space Theories in Computer Vision*, pages 306–317.

Samson, C., Blanc-Féraud, L., Aubert, G., and Zerubia, J. (2000). A level set model for image classification. *International Journal of Computer Vision*, 40(3):187–197.

Sandberg, B., Chan, T., and Vese, L. (2002). A level-set and gabor-based active contour algorithm for segmenting textured images. Technical report, UCLA Department of Mathematics CAM report 02-39.

Scott, D. W. (1992). *Multivariate Density Estimation, ISBN:0471547700*. Willey InterScience, first edition.

Sethian, J. (1999). *Level Set Methods and Fast Marching Methods. Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

Shi, Y. and Karl, W. C. (2005a). A fast level set method without solving pdes. In *IEEE International Conference on Acoustics, Speech, and Signal Processing.*

Shi, Y. and Karl, W. C. (2005b). Real-time tracking using level sets. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pages 34–41, Washington, DC, USA. IEEE Computer Society.

Shi, Y. and Karl, W. C. (2008). A real-time algorithm for the approximation of level-set-based curve evolution. *IEEE Trans. on Image Processing*, 17(5):645–656.

Silverman, B. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London.

Snyder, W. E. and Qi, H. (2004). *Machine Vision*. Cambridge University Press.

Stenger, B. (2004). *Model-Based Hand Tracking Using A Hierarchical Bayesian Filter*. PhD thesis, Department of Engineering, University of Cambridge.

Stenger, B., Thayananthan, A., Torr, P. H., and Cipolla, R. (2006). Model-based hand tracking using a hierarchical bayesian filter. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28:1372–1384.

Strang (2008). http://ocw.mit.edu/ocwweb/mathematics/18-085fall-2007/videolectures/index.htm.

Surgical, I. (2008). http://www.intuitivesurgical.com/index.aspx.

Svoboda, T., Martinec, D., and Pajdla, T. (2005a). A convenient multi-camera self-calibration for virtual environments. *PRESENCE: Teleoperators and Virtual Environments*, 14(4):407–422.

Svoboda, T., Martinec, D., and Pajdla, T. (2005b). Multi-camera self-calibration toolbox (http://cmp.felk.cvut.cz/ svoboda/selfcal).

T., F. W. and Weissman, C. (1995). Television control by hand gestures. In *IEEE International Work-shop on Automatic Face- and Gesture- Recognition*, pages 179–183.

Tabelow, K., Polzehl, J., Spokoiny, V., and Voss, H. (2006). Analysing fmri experiments with structural adaptive smoothing procedures. *Neuroimage*, 33:55–62.

Thayananthan, A., Stenger, B., Torr, P. H. S., and Cipolla, R. (2003). Shape context and chamfer matching in cluttered scenes. In *Proc. Conference on Computer Vision and Pattern Recognition*.

Tönnies, K. D. (2005). *Grundlagen der Bildverarbeitung*. Pearson Studium.

Trucco, E. and Verri, A. (1998). *Introductory Techniques for 3-D Computer Vision*. Prentice-Hall, ISBN-10: 0132611082.

Tsai, A., Yezzi, A., and Willsky, A. (2001). Curve evolution implementation of the mumford-shah functional for image segmentation, denoising, interpolation, and magnification. *IEEE Trans. Image Processing*, 10(8):1169–1186.

Tsao, D. Y. and Livingstone, M. S. (2008). Mechanisms of face perception. *Annual Reviews - Neuroscience*, 31:411–437.

Tziritas, G. and Labit, C. (1994). *Motion Analysis for Image Sequence Coding*. Elsevier Science Inc., New York, NY, USA.

Unser, M. (1995). Texture classification and segmentation using wavelet frames. *IEEE Transactions on Image Processing*, 4(11):1549 – 1560.

Varma, M. and Zisserman, A. (2003). Texture classification: Are filter banks necessary? In *IEEE Conf. Comp. Vis. and Pattern Recog.*

Varma, M. and Zisserman, A. (2008). A statistical approach to material classification using image patch exemplars. *IEEE Trans. Pattern Anal. Mach. Intell.*, under submission.

Vese, L. A. and Chan, T. F. (2002). A multiphase level set framework for image segmentation using the mumford and shah model. *International Journal of Computer Vision*, 50(3):271–293.

Viola, P. (1995). *Alignment by Maximization of Mutual Information*. PhD thesis, MIT, Artificial Intelligence Laboratory.

Viola, P. and Jones., M. (2001a). Rapid object detection using a boosted cascade of simple features. In *Proc. Computer Vision and Pattern Recognition*.

Viola, P. and Jones., M. (2001b). Robust real-time object detection. In *Internat. Workshop on Statistical and Computational Theories of Vision*.

Viola, P. and Wells, W. M. (1997). Alignment by maximization of mutual information. *International Journal of Computer Vision*, 24(2):1997.

Wade, C. and Tavris, C. (2005). *Psychology. (7th ed.)*. Upper Saddle River, NJ: Pearson Education, Inc.

Wang, Z. and Vemuri, B. (2004). Tensor field segmentation using region based active contour model. In *Proc. of European Conf. Comp. Vis.*, page 304Ű315.

Weickert, J., Romeny, B., and Viergever, M. (1998). Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Transactions on Image Processing*, 7(3):398–410.

Whitaker, R. T. (1998). A level-set approach to 3d reconstruction from range data. *International Journal of Computer Vision*, 29(3):203–231.

Wilson, A. and Oliver, N. (2003). Gwindows: Robust stereo vision for gesture-based control of windows. In *Proc. International Conference on Multimodal Interfaces, .*

Wolf, L., Huang, X., Martin, I., and Metaxas, D. (2006). Patch based texture edges and segmentation. In *European Conf. Comp. Vis.*, volume 2, page 481Ű493.

Wu, Y. and Huang, T. S. (2002). Non-stationary color tracking for vision-based human computer interaction. *IEEE Trans. on Neural Networks*, 13(4):948–960.

Wu, Y., Lin, J., and Huang, T. S. (2001a). Capturing natural hand articulation. In *Proc. IEEE Int'l Conf. on Computer Vision*, pages 426–432.

Wu, Y., Lin, J., and Huang, T. S. (2001b). Capturing natural hand articulation. In *Proc. IEEE Internat. Conf. on Computer Vision*.

Xiang, Y., Chung, A., and Ye, J. (2005). A new active contour method based on elastic interaction. In *Conf. Computer Vision Pattern Recognition*, page 452Ű457.

Xie, X. and Mirmehdi, M. (2004). Rags: Region-aided geometric snake. *IEEE Trans. on Image Processing*, 13:640Ű652.

Xie, X. and Mirmehdi, M. (2006). Magnetostatic field for the active contour model: A study in convergence. In *Proc. of British Machine Vision Conference*, page 127Ű136.

Xie, X. and Mirmehdi, M. (2008). Mac: Magnetostatic active contour model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(4):632Ű646.

Xu, C., Jr., A. Y., and Prince, J. L. (2001). A summary of geometric level-set analogues for a general class of parametric active contour and surface models. In *Proc. of 2001 IEEE Workshop on Variational and Level Set Methods in Computer Vision*, pages 104–111.

Xu, C. and Prince, J. (1997). Gradient vector flow: A new external force for snakes. In *Proc. IEEE Conf. on Comp. Vis. Patt. Recog.*, pages 66–71.

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

Yang, R., Mirmehdi, M., and Xie, X. (2006). A charged active contour based on electrostatics. In *Proc. of Advanced Concepts for Intelligent Vision Systems*, page 173Ű184.

Yilmaz, A., Li, X., and Shah, M. (2004). Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Trans. Pattern Anal. Machine Intell.*, 26(11):1531–1536.

Zalesny, A. and Gool, L. V. (2000). A compact model for viewpoint dependent texture synthesis. In *European Conf. Comp. Vis., LNCS 2018/5*.

Zhao, H., Chan, T., Merriman, B., and Osher, S. (1996). A variational level set approach to multiphase motion. *Journal of Computational Physics*, 127:179Ű195.

Zhu, S. C. and Yuille, A. (1996). Unifying snake/balloon, region growing and bayes/mdl/energy for multi-band image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(9):884–900.