Global Optimization Using Embedded Graphs

HIROSHI ISHIKAWA

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Department of Computer Science
Courant Institute of Mathematical Sciences
New York University
May 2000

Approved: ____________________
Davi Geiger
Dissertation Advisor

精強博敏

辛酉孟秋
飞来象之

To my parents.

Acknowledgments

I would like to thank Professor Davi Geiger, my thesis advisor, for his continuous encouragement, support, and friendship.

I would also like to thank Ian Jermyn, my colleague and dear friend, for the fruitful collaboration that is a part of this thesis and for his help in English.

Special thanks go to Dan Cintron and Izumi Okubo Cintron, for their great help in bootstrapping my life in New York and their friendship ever since.

I am very grateful to Anina Karmen for her patient help since even before I came to New York, and to Rosemary Amico and Lourdes Santata for maintaining the great academic environment, which is essential to any research.

I would also like to thank Professor Alan Siegel, for giving me a chance and letting all this happen.

The members of the thesis committee, besides Davi, were Professor Richard Cole, Dr. David W. Jacobs, Professor Robert V. Kohn, and Dr. Ronen Basri. I would like to thank them very much.

I would like to thank my friends and fellow students Henning Biermann, Tyng-Luh Liu, Fabian Monrose, Hsing-Kuo "Ken" Pao, and Laxmi Parida for help and being good friends.

I would like to thank the Instituto de Matemática Pura e Aplicada in Rio de

Janeiro and L'Institut Henri Poincaré in Paris for their generous hospitality.

Finally, I would like to thank my parents, Sadao and Keiko Ishikawa, to whom this thesis is dedicated, for their understanding and constant support.

Abstract

One of the challenges of computer vision is that the information we seek to extract from images is not even defined for most images. Because of this, we cannot hope to find a simple process that produces the information directly from a given image. Instead, we need a search, or an optimization, in the space of parameters that we are trying to estimate.

In this thesis, I introduce two new optimization methods that use graph algorithms. They are characterized by their ability to find a global optimum efficiently. Each method defines a graph that can be seen as embedded in a Euclidean space. Graph-theoretic entities such as cuts and cycles represent geometric objects that embody the information we seek.

The first method finds a hypersurface in a Euclidean space that minimizes a certain kind of energy functional. The hypersurface is approximated by a cut of an embedded graph so that the total cost of the cut corresponds to the energy. A globally optimal solution is found by using a minimum cut algorithm. In particular, it can globally solve first order Markov Random Field problems in more generality than was previously possible. I prove that the convexity of the smoothing function in the energy is essential for the applicability of the method and provide an exact criterion in terms of the MRF energy.

The second method proposed here efficiently finds an optimal cycle in a Euclidean space. It uses a minimum ratio cycle algorithm to find a cycle with minimum energy in an embedded graph. In the case of two dimensions, the energy can depend not only on the cycle itself but also on the region defined by the cycle. Because of this, the method unifies the two competing views of boundary and region segmentation.

I demonstrate the utility of the methods in applications, with the results of experiments in the areas of binocular stereo, image restoration, and image segmentation. The image segmentation, or contour extraction, experiments are carried out in various situations using different types of information, for example motion, stereo, and intensity.

Contents

List of Figures

one

Introduction

The ultimate objective of computer vision is to make a machine *see*. The human vision system can extract varied and intricate information from a single picture, or even a line drawing. This has turned out to be an enormously difficult feat to simulate in a machine; for decades researchers in artificial intelligence and computer vision have been struggling to this end. Why is it so hard? One reason is this: *the kinds of information we try to extract are not even defined for almost all images.* The parameters that we wish to estimate are not defined in terms of images. To see what this means, we need to realize what an image is as raw data, and exactly what the desired information is.

What is an image? It is easy for us to forget about an image. When we *look* at an image, we usually do not *see* it; instead, we see what is shown in it. When we look at a portrait of a person, we do not contemplate an unimaginable number of tiny colored dots. It is like a hole through which we can see the other side, where the person looks back at us. It is remarkable that such a sense of reality is conveyed through a simple array of colors, which is what an image is. Photography films, CCDs, and retinas are all arranged in the same way for the same purpose—they record a collection of colors and their positions. As a data structure, a rectangular array of pixels is standard.

When considered this way, almost no images, as possible combinations of pixel values, are what we would think of as images—they are white noise. Imagine a television turned to a dead channel. What you see is amplified thermal noise, and you never see anything that looks like anything other than noise. In other words, if you choose an image randomly from this space of all possible combinations of color values for all pixels, the result does not make sense to a human observer. This is because the information we extract from an image is not about the image itself, but rather about the scene, or what has caused the image. The flow of information is not from the image to the scene to whatever parameter of the scene we are trying to find out. It is exactly the opposite. To find the origin of the image, we have to search. We naturally are looking for something in the image; something our knowledge (innate or acquired) tells us we should find. When the search fails, we sometimes see an illusion, but when we are looking at a completely random image, we are more likely to see nothing at all.

Thus the information we are trying to extract in computer vision, even low-level features such as presence of an object boundary or stereo correspondences, let alone high-level characteristics like facial expressions, are properties of the scene that gives rise to the image, rather than those of the image as an array of pixels. In trying to design an artificial vision system, this means that whatever we want to measure *is not a priori a function of images*; it is erroneous to assume that a scene property is defined on the space of images. Rather, the process of obtaining information from an image is a search, in the space of parameters of a model of the scene, for the values most likely to give rise to the image. Of the space of images, the part that consists of comprehensible images is vanishingly small; yet the process must be general enough to handle any image, because there is no simple way to define the comprehensible part.

This might seem somewhat at odds with the widely held view that low-level vision tasks such as contour extraction are "bottom-up" processes; features are extracted from an image and separate higher-level processes interpolate to find a suitable representation. Of course, bottom-up techniques can be applied before the search, perhaps replacing the space of images with the space of the results of this process; but a search must be done eventually. It might even be claimed that the necessity of such a search is what differentiates computer vision from image processing (though an image processing system might use models of images.)

Let us take a look at how the problem is usually put in terms of probability theory. A space of parameters—parameters we want to estimate—is defined; a model of the causal relationship between the parameters and the resulting images is given as a conditional probability distribution; then, for a given image, the parameters are estimated by maximizing the probability that they occur and give rise to the image. This technique is called Maximum A Posteriori (MAP) estimation, and is standard in computer vision and image processing. In this way, the whole system is represented by the probability that different values of the parameters occur *a priori* and the probability that the image occurs under the assumption that the parameters have some given value.

But where has the asymmetry gone? I claimed above that the flow of information is one way and not the other. Once everything is put in terms of probability distributions, however, there seems to be no reason to worry about what was originally defined where. After all, everything has a probability and the joint probability distribution is defined on the product of image and parameter spaces, which does not discriminate between the two factors.

The crucial point, however, is that these various probability distributions are always given as a function that is defined as an algorithm. Because the parameter space is usually vast, e.g., exponential in terms of the number of pixels, the information must be compressed to the limit; no distribution can ever be given as a simple table of data, even when it is empirically determined. Thus the model, defined by the distributions, is always given in a computable form *as an algorithm*. For instance, a Gaussian distribution is an algorithm that returns the exponential of the squared ratio of the standard deviation and the distance from the mean. Because these parameters of the models are not a function of the images, one direction is much harder than the other. The conditional probability that the parameters have particular values when the resulting image is given would be quite hard to express in algorithmic terms. It is akin to trying to decode an encrypted text by a brute-force attack, rather than encoding a plain text. Computer vision is inherently more difficult than computer graphics exactly because of this.

Thus every vision problem is conceptually a search process, as opposed to a deterministic encoding. The solution to a vision problem needs three components: a prior model for the parameter space, a model of image formation, and an efficient search method to carry out the MAP estimation. The first two vary greatly from problem to problem; they depend on what we are trying to estimate in a unique way. In this thesis, I mainly address the third component. I introduce two new optimization methods that use graph algorithms. They are characterized by their ability to find a global optimum efficiently. Each method defines a graph that can be seen as embedded in a Euclidean space. Graph-theoretic entities such as cuts and cycles represent geometric objects that embody the information we seek.

In the next chapter, I give an overview of optimization methods used in computer vision. In particular, I first review the notion of a Markov Random Field, and then two classes of general techniques: gradient descent and stochastic optimization. After that, I examine graph-based schemes, including those that are particularly close to the methods I propose.

In chapter 3, I discuss the low-level vision problems that I address in this thesis: specifically, stereo and segmentation/contour extraction.

In chapters 4 and 5, I introduce the methods in a general setting.

The first method finds a hypersurface in a Euclidean space that minimizes a certain kind of energy functional. The hypersurface is approximated by a cut of an embedded graph so that the total cost of the cut corresponds to the energy. A globally optimal solution is found by using a minimum cut algorithm. In particular, this method can globally solve first order Markov Random Field problems in more generality than was previously possible. I prove that the convexity of the smoothing function in the energy is essential for the applicability of the method and provide an exact criterion in terms of the MRF energy.

The second method, described in chapter 5, can efficiently find an optimal cycle in a Euclidean space. It uses a minimum ratio cycle algorithm to find a cycle with minimum energy in an embedded graph. In the case of two dimensions, the energy can depend not only on the cycle itself but also on the region defined by the cycle. Because of this, the method unifies the two competing views of boundary and region segmentation.

In chapter 6, I demonstrate the utility of the methods in applications with the results of experiments in stereo, image restoration, and image segmentation. The image segmentation, or contour extraction, experiments are carried out in various situations using different types of information, for example motion, stereo, and intensity.

Finally, in chapter 7, I conclude with a discussion of future directions.

two

Optimization in Computer Vision

Here I give an overview of the optimization framework and the methods used in computer vision. In particular, I first review the notion of Markov Random Field, and then two classes of general optimization techniques: gradient descent and stochastic optimization. After that, I examine graph-based schemes, including those that are particularly close to the methods that I propose in this thesis.

2.1 Optimization Framework

In this thesis, by an image I mean a simple array of real non-negative values:

$$I_{xy} \in \mathbf{R}^{\geq 0}; \quad x = 1, \ldots, N; \; y = 1, \ldots, M,$$

or a function on some rectangular region $\check{I}$ in $\mathbf{R}^2$:

$$I(x,y) \in \mathbf{R}^{\geq 0}; \quad (x,y) \in \check{I}.$$

We denote the domain of the image function I by $\check{I}$. The space of images is denoted by $\mathcal{I}$.

In computer vision, images are usually *given*—i.e. they are the input. The output, what is not given and which we wish to obtain, is the *parameters*—that is, the quantities that we want to estimate. Because we do not know what particular values the parameters have, we talk about the whole space of parameters, or the space of possible values the parameters can take. The optimization with which we are concerned in this thesis is a process of finding the most likely value in this space given one or more images. We do this by estimating the probability that the given image is observed, given each parameter value.

Let us denote the space of parameters by $\mathcal{X}$. A probability distribution P on $\mathcal{X}$ gives a number between 0 and 1 for an event, i.e., a subset of $\mathcal{X}$. For a finite $\mathcal{X}$ and a parameter value $X \in \mathcal{X}$, $P(X)$ denotes $P(\{X\})$, the probability for X.

The probability $P(X)$ is called the prior probability, because its distribution does not depend on the data, which is usually the image. For a given vision problem, a prior distribution is either implicitly or explicitly defined. Though in principle this is a very important part of the model, exerting great influence on the behavior of the system, estimation of the prior is not very easy. I discuss the prior function in each section dealing with specific problems.

The second part of the probabilistic model is the image formation model. For a given image $I \in \mathcal{I}$ and a parameter value $X \in \mathcal{X}$, $P(I|X)$ gives the conditional probability that the image I is observed under the assumption that the parameter in fact has the value X. This should give a high probability to the image that is consistent with the value. For instance, if the parameter describes a configuration of objects in front of the camera, an algorithm can simulate the physical situation of the world and the camera, thus evaluating how consistent the given image is with the parameter value, by checking, for instance, if the intensity boundaries in the image coincide with the projection of the object boundaries that are expected from the physical model and the parameter value.

Supposing these two probability distributions are given, the MAP estimation seeks, for a given image I, the parameter X that maximizes the posterior probability $P(X|I)$, which is the conditional probability that the parameter has value X under the assumption that the image I is observed. By Bayes' rule,

$$P(X|I) = \frac{P(I|X)P(X)}{P(I)},$$

where $P(I)$ is the prior probability for the image I. Since I is fixed, this value can be optimized by maximizing the $P(I|X)P(X)$ using the prior model $P(X)$ and the image formation model $P(I|X)$.

In general, for a probability distribution $P(X)$, it is customary to minimize $E(X) = -\log P(X)$ rather than to maximize $P(X)$; and the negative log probability $E(X)$ is called the energy. This is a tradition from statistical thermodynamics, where the Gibbs probability distribution:

$$P(E) \sim \mathrm{e}^{-\frac{E}{kT}},$$

gives the probability of a microscopic state with energy E at temperature T.

2.1.1 Markov Random Field

Note that usually both the image space $\mathcal{I}$ and the parameter space $\mathcal{X}$ are huge. The large size of the parameter space, in which we perform the search, is especially problematic; there seems to be no hope of solving the problem without exploiting some regularity in the space. As the large size is often due to the space being the product of many state spaces belonging to smaller parts, one common way to reduce the complexity is to assume a conditional statistical independence between the parts: a Markov Random Field (Dobrushin [23]), or an MRF, is a set of small state spaces (random variables) such that each variable is statistically independent of all but a few "neighbor" variables. It is a popular and often useful way to model vision parameter spaces.

An MRF consists of an undirected graph $G = (V, E)$, a finite set L of labels, and a probability distribution P on the space $\mathcal{X}(G) = L^V$ of parameters. That is, $\mathcal{X}(G)$ is a set of assignments $X : V \ni v \mapsto X_v \in L$. Let $\mathcal{N}_v$ denote the set of neighbors $\{u \in V | (u,v) \in E\}$ of v. Also, for $X \in \mathcal{X}(G)$ and $S \subset V$, let X_S denote the event $\{Y \in \mathcal{X}(G) | Y_v = X_v, \forall v \in S\}$, that is, the subset of $\mathcal{X}(G)$ defined by values at vertices in S. The probability distribution must satisfy the condition:

$$P(X) > 0 \ \text{ for all } X \in \mathcal{X}(G)$$
$$P(X_{\{v\}} | X_{V \setminus \{v\}}) = P(X_{\{v\}} | X_{\mathcal{N}_v}).$$

This condition states that the assignment at a vertex is conditionally dependent on other assignments only through its neighbors.

Note that the MRF is a conditional probability model. An important theorem (Besag[9], Kinderman and Snell[51]) connects it to a joint probability model: a probability distribution P on $\mathcal{X}(G)$ is an MRF exactly when it is a Gibbs distribution relative to G:

$$P(X) \sim \mathrm{e}^{-E(X)},$$

$$E(X) = \sum_{C \in \mathcal{C}} E_C(X),$$

where $\mathcal{C}$ denotes the set of cliques in G and E_C a function on $\mathcal{X}$ with the property that $E_C(X)$ depends only on values X_v for $v \in C$.

The simplest case is when only the edges and vertices influence the potential:

$$E(X) = \sum_{(u,v) \in E} g(u, v, X_u, X_v) + \sum_{v \in V} h(v, X_v). \tag{2.1}$$

This is called a first order MRF.

Note that in the above definition no data or observation, such as the image, appears. Any data or observation that affects the statistics is implicit. For instance, in (2.1), $h(v, X_v)$ might depend on the image through v, as in $h(v, X_v) = |I(v) - X_v|$.

2.2 Optimization Methods

In this section, I review gradient descent and stochastic optimization as examples of traditional optimization methods.

2.2.1 Gradient Descent

Gradient descent, or hill climbing, is a basic optimization technique with a very broad range of applications. It is essentially a local approximation of the objective function by a linear function, and can be used whenever the objective function has an easily computable gradient. If higher derivatives are computationally cheap, higher order approximations (quadratic, etc.) can also be used.

Gradient descent has two major problems: initialization and local optima. Since it is an iterative algorithm it must begin somewhere, and the result and performance of the algorithm depend greatly on the initialization; it is often difficult to find a way to initialize reliably, especially in a high-dimensional problem such as computer vision. Secondly, because the algorithm can determine only whether a local optimum has been achieved, it often converges to a local optimum and fails to find a better solution.

Despite these shortcomings, it is conceptually simple and applicable in a very broad class of problems (and thus the last resort when there is no other way), and it has been used extensively in computer vision. Besides the general use of the method, the following important techniques in vision use gradient descent:

Snakes The "snake" method, also known as the active contour method, is a popular technique that uses gradient descent to optimize locally a functional on the space of curves in an image and hence extract a contour from the image (Kass, Witkin, and Terzopoulos[50]; Blake and Zisserman[11].) The energy functional usually has both an image formation model, which looks for places that are more suitable for the contour to reside (by looking for a higher intensity gradient for example,) and a prior model, which biases the search towards the desired kind of curves (smoother curves, for example.) The main advantages of snakes are those of all optimization techniques: the image data, desired contour properties, and knowledge-based constraints are integrated into a single extraction process. On the other hand, in addition to the disadvantages that plague all gradient descent techniques—the final extracted contour is highly dependent on the position and shape of the initial contour as a consequence of the many local minima in the energy function; the initial contour must be placed near the required feature otherwise the contour can become obstructed by unwanted features—it has more domain-specific problems stemming from the energy functional: the energy has many parameters that must be chosen carefully to obtain meaningful results, due to scale variance and parameter inter-dependence.

Level Set Method The level set method (Osher and Sethian[68], Sethian[75]) tracks the motion of an interface by embedding the interface as the zero level set of the signed distance function. The motion of the interface is matched with the zero level set of the level set function, and the resulting initial value partial differential equation for the evolution of the level set function resembles a Hamilton-Jacobi equation. In this setting, curvatures and normals may be evaluated easily, topological changes occur in a natural manner, and the technique extends trivially to three dimensions. The equation is solved using entropy-satisfying schemes borrowed from the numerical solution of hyperbolic conservation laws. These schemes produce the correct viscosity solution.

Faugeras and Keriven[27] use this method to evolve surfaces in a 3D space in order to reconstruct surfaces from stereo data. It is based on a variational principle, which provides a set of PDE's that are used to deform an initial set of surfaces, which then move towards the objects to be detected.

2.2.2 Stochastic Optimization

The gradient descent method goes straight for the nearest minimum as fast as possible, always going downhill and never going uphill. Stochastic optimization algorithms randomly allow occasional uphill jumps, enabling escape from local minima and the possibility of finding the global minimum (Metropolis et al[61]). Among them, the simulated annealing algorithm initially allows such jumps with a high probability. Then, according to some "annealing schedule", it gradually lets the probability go down to zero. This method is based on an analogy with the thermodynamics of annealing real metals. Thus, the annealing schedule is usually given in terms of the Gibbs distribution

$$P(x,x') = \mathrm{e}^{-\frac{E(x)-E(x')}{kT}},$$

by a schedule to lower the temperature T, where $P(x,x')$ denotes the probability of a jump from $x \in \mathcal{X}$ to x' ($P(x,x') > 1$ signifies certainty.) The annealing schedule and a method for selecting a new point to which to jump are required for each specific problem. Usually experimentation is necessary to obtain these for a particular problem.

Simulated annealing proved to be particularly suitable for MRF problems, although it is also notorious for its slowness. Geman and Geman[32] popularized the MRF in the vision/image processing community. They use simulated annealing for image restoration by MAP estimation, and even prove the "annealing theorem," which roughly says that if the temperature $T(k)$ at time k goes down at the rate $(\log(1+k))^{-1}$, the algorithm is guaranteed to reach the MAP solution, although it takes "forever" to guarantee correct solution.

There is a similar but different class of techniques known as "Relaxation Labeling" (Rosenfeld, Hummel, and Zucker[72]; Hummel and Zucker[39]), which is very popular in low-level vision tasks in computer vision and image processing.

2.3 Methods Using Graphs

2.3.1 Approximation Using Graphs

Multi-way Cut Boykov, Veksler, and Zabih[12] introduced a new technique that approximately solves MRF problems efficiently, although for a rather specific prior. It poses the problem as an instance of the minimum N-way cut problem for an undirected graph as follows: consider an MRF problem on $G = (V, E)$ with a label set L and an energy that is a special case of (2.1):

$$E(X) = \sum_{(u,v)\in E} g(u,v)(1 - \delta_{X_u,X_v}) + \sum_{v\in V} h(v, X_v),$$

where $g(u,v)$ is a nonnegative real number. The technique defines a new graph $\tilde{G} = (V \cup \tilde{L}, E \cup E_L)$, where $\tilde{L} = \{u_l | l \in L\}$ is a set of additional vertices, each of which corresponds to a unique label, and edges in $E_L = \{(v, u_l) | v \in V, u_l \in \tilde{L}\}$ connect these new vertices and each of the original vertices. Every edge has a weight—a real number—as follows: Each original edge $(u,v) \in E$ has a weight $g(u,v)$. Each new edge $(v, u_l) \in E_L$ has a weight $-h(v,l)$. Now, an N-way cut is a partition of the graph into $N = |L|$ parts, each of which includes exactly one vertex from $\tilde{L}$. For a given N-way cut, we sum up the weights of all edges that have their ends in different partitions (they are said to be cut); this is called the cost of the cut. It can be shown that minimizing this sum is equivalent to minimizing the MRF energy. A large enough number is added to the weight of each edge in E_L so that every edge has a nonnegative weight. The number that is added to the cost of any N-way cut is the same regardless of the cut, because there are always exactly $|V|(|L| - 1)$ cut edges in E_L for any N-way cut. This can be seen by counting the number of edges in E_L that originate from a vertex v in V. An edge $(v, u_l) \in E_L$ is cut unless v belongs to the same partition as u_l. Thus, of $|L|$ edges in E_L that originate from v, $|L| - 1$ are cut. Hence there are $|V|(|L| - 1)$ cut edges in E_L. In this way, we can make all weights nonnegative without changing the minimum cut or any relative cost difference between cuts. For a graph with nonnegative edge weights, there exists an efficient approximation algorithm to solve the problem. For this technique to work, the prior between two sites must be of the above very specific form, i.e., zero if the labels are the same and some constant value otherwise. In other words the space of labels cannot have much structure, such as an order that influences the prior. This technique was used in [12] for binocular stereo, and was improved by Birchfield and Tomasi[10].

Normalized Cut Shi and Malik[77] use a generalized eigenvalue method to approximately solve a graph-partition problem, which they derive from an image-partition problem.

2.3.2 Global Optimization

There has been limited success in finding global optima of MRF models.

In a situation where the state is described as a string of linearly ordered local states, dynamic programming can be used (Amini et al[3], Geiger, Gupta, Costa, and Vlontzos[29].) Montanari[62] uses it to find the minimum energy path between given endpoints. Baker and Binford[6] use this approach for the stereo correspondence problem (see 3.2.4).

Binary MRF Consider an MRF problem on $G = (V, E)$ with a label set $L = \{0, 1\}$, and a form of first order energy (2.1):

$$E(X) = \sum_{(u,v)\in E} g(X_u - X_v) + \sum_{v\in V} h(v, X_v),$$

where $g(X_u - X_v) \geq 0$. Greig, Porteous, and Seheult[35, 36] give an efficient way to find a globally optimal solution for this problem using a minimum-cut algorithm. The solution is exactly the same as for the multi-way cut case, except that there are efficient algorithms that solve the problem exactly. Also, the condition that the function $g(X_u - X_v)$ must be of the form $g(u, v)(1 - \delta_{X_u, X_v})$ is almost irrelevant, since there are only three possible values of $X_u - X_v$.

The multi-way cut[12] is the result of an attempt to generalize this result. This is a rare situation where the exact solution to an MRF can be obtained. One of the methods I propose is a generalization of this algorithm, which provides an efficient exact solution, and which differs from the generalization used in the multi-way cut method.

Greig, Porteous, and Seheult also give a good reason for using a guaranteed method. They compared their method experimentally with simulated annealing in image restoration. Although simulated annealing is also theoretically guaranteed to reach the optimal solution eventually, they found that in practice simulated annealing tended to over-smooth the noisy image.

three

Problems Defined

The low-level vision problems that are addressed in this thesis are both prototypical problems in the area. Specifically, they are stereo and segmentation/contour extraction.

3.1 Segmentation and Contour Extraction

The transparency of the medium in which we live, and the relative cohesiveness and opaqueness of the objects in it, make image segmentation an important step in visual information extraction. That is, object identity tends to stay constant in this world; hence it makes sense to detect boundaries and segregate the scene into objects, which can subsequently (for instance) be tracked or identified, rather than trying to track and detect all events in the view independently (and making sense of it all.) Also, and consequently, the shapes of the regions and boundaries are important information in themselves.

In a single image, objects and their boundaries manifest themselves as regions and their 1D boundaries. The main low-level cues are local intensity edges. Local edges are detected in areas of the image where the intensity level fluctuates sharply; the more rapid the intensity change, the stronger the edge. If a number of images of the same scene are available, as is the case for stereoscopy and video sequences, there are other clues we can utilize for segmentation by correlating between images. With a correspondence of points in images, we can deduce characteristics such as the depth and the motion velocity of each pixel, using this information to classify them into groups.

The problem generally divides into dual approaches: a) determining boundaries between regions by detecting local discontinuities and grouping them to form object boundaries; or b) mapping individual pixels into sets of pixels with similar properties (intensity, color, texture etc.) which define regions within the image. Both methods have their advantages and drawbacks.

3.1.1 Region Segmentation

Region segmentation is probably less common than edge detection as a low level processing operation in computer vision systems, but it is applicable in environments that are highly textured or colored, for example outdoor scenery viewed by a mobile vehicle. Its ontology—the space of pixel classifications—is simple, and well suited to the convenient machinery of MRFs. Thus, region-based methods tend to be global, optimizing a functional of the image segmentation. On the other hand, they often ignore important boundary properties such as smoothness. Region approaches have shown promise in motion segmentation (Weiss[78]), and they yield segmentation directly (Darrell and Pentland[22]). Shi and Malik[77], in an approach related to the work by Sarkar and Boyer[74], use a generalized eigenvalue method to find normalized cuts. Leung and Malik[54] extend their work by incorporating weak contour continuity information into the region-based model. There are also approaches using MRFs (Blake and Zisserman [11], Geman and Geman[32].) Mumford and Shah[64] give a variational formulation, representing an image by a piecewise-smooth function. There is also a large clustering community, which has produced various approaches to this problem, for example merge and split techniques, the K-means approach, and others (Jain and Dubes[44].)

Parameter Space, Image Formation Model, and Prior Model The parameter space of the region segmentation model is often very close to the image representation. The prototype is a space of label-assignments to pixels. This is so simple that it does not seem to warrant the model-based optimization scheme. However, this is not so; we can see this by recalling that the model consists not only of the parameter space but also of the probability distributions. For instance, consider a simple clustering approach. It groups together pixels with similar properties, because points on the same object should possess similar properties according to the prior model, and these points should be projected onto the image with such similarities intact according to the image formation model; these properties are not inherent properties of the image. Also, as a method becomes more sophisticated, the bottom-up scheme gets more and more conceptually obscure. For example, if the method incorporates extra structure such as the presence of edges or junctions, it is much clearer to include it as part of the model, rather than trying somehow to devise a way to produce such structures bottom-up.

3.1.2 Boundary Segmentation

Boundary-based approaches divide into two classes. One of them, local edge detection (J. F. Canny[14]) and the grouping thereof, has been one of the central problems of computer vision. Shashua and Ullman[76] use a "Saliency Network" of locally connected elements, which gives a rating of the best connections between each primitive (e.g. an edge) and its neighbors. Parent and Zucker[69] use co-circularity as a constraint to assign tangent and curvature to every point in the image, using the relaxation labeling technique. Guy and Medioni[38] use a voting scheme to estimate at each pixel in the image a tangent and a salience. We can see that, after these treatments, the results are still a pixel-by-pixel field of features and a further grouping is needed to find a contour. Alter and Basri[2] analyze some problems of the Saliency Network.

The second approach is more explicitly model-based, and is represented by the snakes method mentioned in section 2.2.1. In this class of techniques, the search space is explicitly the space of contours.

A prior is defined in terms of intrinsic properties of the contour, for example curvature. Mumford[63] describes how a particular form of prior distribution can be seen as a random walk process. Williams and Jacobs[79] use a similar prior model expressed in terms of random walks, which they claim is feasible as a neural model of the visual cortex.

Image formation models are generally formulated in terms of boundary information like the intensity gradient. This is a problem with boundary-based approaches; they cannot incorporate region information easily. It is utterly impossible if the contour is open-ended; it simply does not have anything to do with any region. This is another reason to try to find closed contours, in addition to the wealth of evidence of their importance in human vision (Kanizsa[47, 48], Elder and Zucker[24, 25], Kovács and Julesz[52].)

Also, in this class of approaches, optimization techniques are used explicitly: the original snake model uses gradient descent. Montanari[62] uses dynamic programming to find the minimum energy path between given endpoints. Geiger, Gupta, Costa, and Vlontzos[29] use initialization with a series of points, and a choice of window around those points, to delineate the space of contours considered. Cox, Rao, and Zhong[20] use a graph algorithm known as the pinned ratio algorithm to find closed contours in an image, and can find a global optimum for a suitably constrained problem class.

3.2 Stereo

The human brain can fuse two slightly different views from left and right eyes and perceive depth. This process of stereopsis entails identifying matching locations in the two images and recovering the depth from their disparity. This can be done only approximately: ambiguity arising from such factors as noise, periodicity, and large regions of constant intensity makes it impossible to identify all locations in the two images with certainty. Indeed, this “correspondence problem” is the most difficult part of the process.

There has been much work on stereo, both in psychophysics (Julesz[46]; Burt and Julesz[13]; Gillam and Borsting[33]; Pollard, Mayhew, and Frisby[71]) and in computer vision (Marr and Poggio[58, 59]; Grimson[37]; Champolle, Geiger, and Mallat[16]; Okutomi and Kanade[67]; Marapane and Trivedi[57]; Ayache[5]; Roy and Cox[73]).

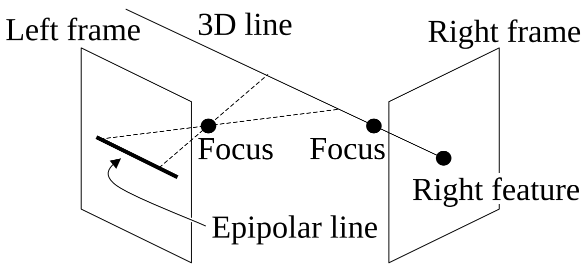


Figure 3.1: A pair of frames (eyes) and an epipolar line in the left frame.

3.2.1 Parameter Space

In binocular stereo, there are left and right images I_L and I_R; the parameter to be estimated is the matching between visible sites in the two images, which is directly related to the depth surface (3D scene) S in front of the cameras. A match between the two images can naturally be represented as a surface in a 4D space $\check{I}_L \times \check{I}_R$, which is called the match space. Remember that we denote by $\check{I}$ the domain of image function I; here we are assuming the two domains are identical rectangles. A point in the match space is a pair of points in the left and right images, which is interpreted as a match between the points. Note that the parameter space is not the match space, but the space of surfaces therein (with certain constraints.)

Two constraints in the geometry of stereo make the parameter space smaller.

Epipolar Constraint Each point in the scene goes through a unique plane in the 3D space defined by the two focal points of the cameras and itself; thus the points sharing such a plane form a line on each image. Hence each image is stratified by such *epipolar lines* and there is a one-to-one correspondence between epipolar lines on the two images (see Figure 3.1.) Because of the epipolar constraint, we can assume that the surface in the match space is always included in the subspace

$$\{(x_{\mathrm{L}}, x_{\mathrm{R}}) \in \check{I}_{\mathrm{L}} \times \check{I}_{\mathrm{R}} | x_{\mathrm{L}} \text{ and } x_{\mathrm{R}} \text{ belong to the corresponding epipolar line}\}.$$

Thus, a match can be seen as a surface in a 3D space. In the rest of the present thesis, the two images are always assumed to be rectified, i.e., points that belong to corresponding epipolar lines have the same *y*-coordinate in both images; a match occurs only between points with the same *y*-coordinate. Thus, a match is represented as a surface in the 3D space $\{(l, r, y)\}$.

Ordering Constraint There is also another constraint known as the ordering constraint (Marr and Poggio[58], Baker and Binford[6]). It states that if a point moves from left to right on the epipolar line in the left image, the corresponding point also moves from left to right in the right image. This can be characterized as a local condition (monotonicity constraint) on the tangent plane of the surface representing the match: $\frac{\partial l}{\partial r}$ and $\frac{\partial r}{\partial l}$ must be positive everywhere on the surface. This is not always strictly true for the real 3D scene in the sense that there can be a surface such that corresponding points move from left to right in the left image and from right to left in the right image. For instance, a plane that equally and perpendicularly divides the line segment between focal points would have this property. However, this is a rare situation and even the human visual system cannot handle this. The ordering constraint further reduces the size of the search space.

Multiple Cameras The situation is essentially the same in the case where there are more than two cameras. In the *n*-camera case, the natural match space is the $2n$-dimensional space $\check{I}_1 \times \cdots \times \check{I}_n$. On the other hand, we can pull back the energy to the space of surfaces in the 3D scene from the match space, since there is a unique map from the space of surfaces to the match space. Although at first sight this seems to reduce the search space greatly, this is an illusion, as can be seen easily as follows. Suppose we know the correspondence between $\check{I}_1$ and $\check{I}_2$. Take a pair of corresponding points p_1 and p_2 on epipolar lines e_1 and e_2 in $\check{I}_1$ and $\check{I}_2$, respectively. We know corresponding epipolar lines e'_1 and e'_2 in $\check{I}_3$. Thus, the corresponding point in $\check{I}_3$ is essentially determined, since it has to be the intersection of e'_1 and e'_2. In other words, the epipolar constraint keeps the match space three-dimensional. The ordering constraint applies in an obvious way, too.

3.2.2 Image Formation Model

The image formation model describes what images the cameras record when a 3D scene S is presented in front of them. It is basically a photometric model and is expressed as a conditional probability distribution $P(I_\mathrm{L}, I_\mathrm{R}|S)$ of forming images I_L and I_R, given a 3D scene S. A detailed account of how to determine the camera parameters, triangulation between the cameras, and an error analysis can be found in Faugeras[26].

Also modeled in the image formation model are half-occlusions, or appearances of scene locations in only one of the two images, which correspond to discontinuities in the match surface or a match surface that is perpendicular to the l or r axis, depending on how this situation is modeled (see Figure 3.2.) It has been shown that the detection of half-occlusions is especially important in human stereopsis (Anderson[4], Nakayama and Shimojo[65]). Half-occlusions

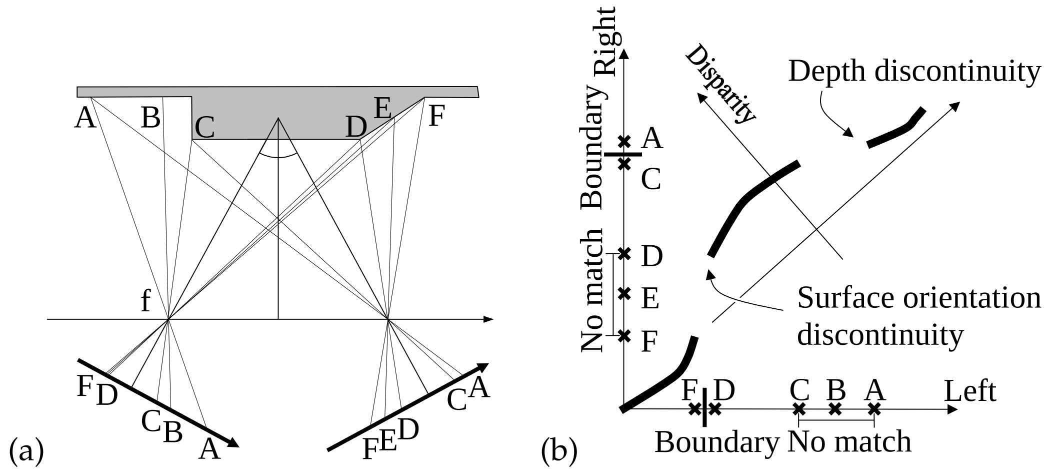


Figure 3.2: (a) A polyhedron (shaded area) with self-occluding regions and with a discontinuity in the surface-orientation at feature D and a depth discontinuity at feature C. (b) A diagram of left and right images (1D slice) for the image of the ramp. Notice that occlusions always correspond to discontinuities. Dark lines indicates where the match occurs.

have also been modeled in artificial vision systems (Belhumeur and Mumford[8]; Geiger, Ladendorf, and Yuille[30]).

Any effect on the images due to the shape and configuration of the 3D surface can be modeled in the image formation model. For instance, intensity edges and junctions can be seen as cues for the depth discontinuities. The significance of junctions in stereo vision has been pointed out by Anderson[4] and Malik[55].

3.2.3 Prior Model

The prior model is an a priori statistical assumption about the 3D scenes, that reveals which surfaces the system expects to find most often in a scene. It is described as a prior probability distribution $P(S)$ that gives a probability to each possible 3D scene output S of the process. In particular, the prior models how any ambiguity is resolved. Belhumeur[7] analyzed stereo prior models in explicitly Bayesian terms. As in other low-level problems, commonly used prior models are local. They generally favor small disparity changes (fronto-parallel surfaces) and small disparity curvature (smooth surfaces).

3.2.4 Search Algorithms

There have been many approaches to stereo algorithms. Search-oriented approaches include the multi-way cut approximation, level set methods, and dynamic programming.

Boykov, Veksler, and Zabih[12] solve the problem with the multi-way cut approximation explained in section 2.3.1. Faugeras and Keriven[27] use the level set method to reconstruct surfaces by evolving surfaces in a 3D space (cf. section 2.2.1.)

The dynamic programming approach takes advantage of the epipolar constraint (Ohta and Kanade[66], Geiger, Ladendorf, and Yuille[30].) As we have seen, in binocular stereo the search space is essentially a space of surfaces in a 3D space. Luckily there is a natural stratification in this space, each stratum being a matching space between the corresponding epipolar lines. This subproblem can be solved efficiently by dynamic programming, since a match between lines is a 1D object with a linear order. The problem with this approach is that it must assume statistical independence between epipolar lines, which is far from the truth.

In this thesis, I use a minimum-cut algorithm to solve this problem (see section 6.2.) Using this method, we can find a match between the different pairs of corresponding epipolar lines simultaneously, and thereby incorporate the interaction between epipolar lines. Although Roy and Cox[73], independently from the present method, used the minimum cut algorithm for stereo in a similar way, their technique lacks the support of a clear analysis of what is searched and, perhaps because of that, is suboptimal.

four

Codimension 1 Method

In this chapter, I propose a method that uses a minimum cut algorithm to solve MRF problems globally. In a graph embedded in a Euclidean space, a cut of the graph can be thought of as a hypersurface, a boundary of codimension one that separates the space into two parts.

In the remainder of this thesis, all graphs are treated as directed graphs. Undirected graphs can always be thought of as directed by making two copies of each edge. A graph $G = (V, E)$ consists of a set V of vertices and a set $E \subset V \times V$ of edges. An edge from vertex u to vertex v is denoted by (u, v).

4.1 Maximum Flow and Minimum Cut

The maximum flow problem and its dual, the minimum cut problem, are classical combinatorial problems with a wide variety of scientific and engineering applications. The maximum flow problem and related flow and cut problems have been studied intensively for over three decades and are standard in textbooks (e.g., [19]).

Consider a graph $G = (V, E)$ with a cost, or capacity function $c : V \times V \to \mathbf{R}^{\geq 0}$ such that $c(u, v) = 0$ if $(u, v) \notin E$. Choose two special vertices, a *source* s and a *sink* t. A *flow* with respect to (G, s, t) is a function $f : V \times V \to \mathbf{R}$ such that:

1. For all $u, v \in V$, $f(u, v) \leq c(u, v)$.
2. For all $u, v \in V$, $f(u, v) = -f(v, u)$.
3. For all $u \in V \setminus \{s, t\}$, $\sum_{v \in V} f(u, v) = 0$.

The *value* $|f|$ of a flow f is defined as

$$|f| = \sum_{v \in V} f(s, v).$$

A maximum flow is a flow with the maximum value.

For the same setting, a cut is a partition of V into two subsets S and $T = V \setminus S$ such that $s \in S$ and $t \in T$. For a given cut, the total *cost* of the cut is defined as

$$\sum_{u \in S, v \in T} c(u, v).$$

When an edge has its tail in S and head in T, the edge is said to be cut. A minimum cut is a cut with the minimum total cost.

The max-flow min-cut theorem says that by finding a maximum flow, a minimum cut can be found. There are efficient algorithms to find maximum flow. The simple and fast algorithm I chose to use is called the push-relabel method with global relabeling (e.g., Goldberg and Tarjan[34], Cherkassky and Goldberg[17]).

4.2 Solving MRF by Minimum Cut

As defined in section 2.1, the most general first order MRF for a graph $G = (V, E)$ and a label set L has an energy

$$E(X) = \sum_{(u,v)\in E} g(u, v, X_u, X_v) + \sum_{v\in V} h(v, X_v).$$

So far, the binary case (see section 2.3.2) has been the only case where the problem can be solved exactly and efficiently. Here, I extend the class of first order MRF problems that can be solved thus:

> **Assumption.** The label set L is a set of evenly spaced real numbers, and the function $g(u, v, X_u, X_v)$ is of the form $g(X_u - X_v)$ with a convex function $g(x)$.

In other words, we will be concerned with energies of the form:

$$E(X) = \sum_{(u,v)\in E} g(X_u - X_v) + \sum_{v\in V} h(v, X_v) \tag{4.1}$$

with a convex function $g(x)$.

The main idea of the method is to define a graph such that

1. there is a one to one correspondence between configurations of the MRF and cuts of the graph, and
2. the total cost of the cut is exactly the same as the total energy of the configuration.

With such a graph, we can find a minimum energy configuration of the MRF by finding a minimum cut of the graph.

4.2.1 The Graph

Let l be the interval between the label values $l_1 < l_2 < \cdots < l_k$ in L, i.e., $l_i + l = l_{i+1}$. Define a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as follows.

$$\begin{aligned} \mathcal{V} &= V \times L \cup \{s,t\} = \{u_{w,i} \mid w \in V;\ i = 1, \dots, k\} \cup \{s,t\}, \\ \mathcal{E} &= \mathcal{E}_\mathrm{D} \cup \mathcal{E}_\mathrm{C} \cup \mathcal{E}_\mathrm{P}. \end{aligned}$$

Below, each of the three subsets of edges is defined and their capacities are specified.

4.2.2 Data Edges

Data edges implement the data term $h(v, X_v)$ in the energy. The set of data edges is defined by:

$$\begin{aligned} \mathcal{E}_\mathrm{D} &= \bigcup_{v \in V} \mathcal{E}_\mathrm{D}^v, \\ \mathcal{E}_\mathrm{D}^v &= \{(s, u_{v,1})\} \cup \{(u_{v,i}, u_{v,i+1}) | i = 1, \dots, k-1\} \cup \{(u_{v,k}, t)\}. \end{aligned}$$

For each vertex v of the original graph G, $\mathcal{E}_\mathrm{D}^v$ is the series of edges $s \to u_{v,1} \to u_{v,2} \to \cdots \to u_{v,k} \to t$, which we call the column over v.

The capacities of these edges are defined by

$$\begin{aligned} c(s, u_{v,1}) &= +\infty, \\ c(u_{v,i}, u_{v,i+1}) &= h(v, l_i);\ \ i = 1, \dots, k-1, \\ c(u_{v,k}, t) &= h(v, l_k). \end{aligned} \tag{4.2}$$

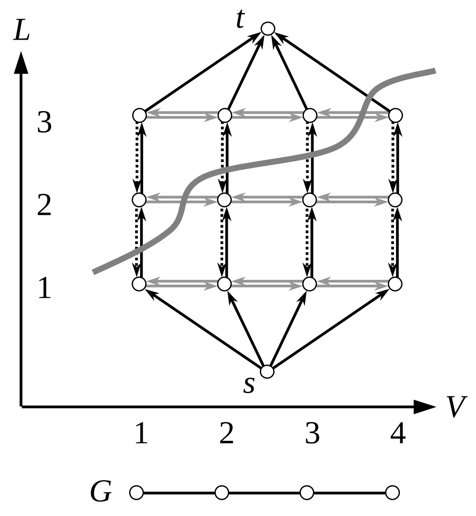


Figure 4.1: Data edges are depicted as black arrows. Four of them are cut here, representing an assignment $X_1 = 1$, $X_2 = 2$, $X_3 = 2$, and $X_4 = 3$. Penalty edges are represented by gray arrows. By crossing consecutive penalty capacities, the cost is added linearly, accounting for the function $g(x,y) = |x-y|$. Constraint edges are depicted as dotted arrows. They ensure that the assignment X_v is uniquely determined. These edges cannot be cut, preventing the cut from "going back".

4.2.3 Constraint Edges

Since any cut of the triple $(\mathcal{G}, s, t)$ separates s and t, it cuts at least one data edge in $\mathcal{E}_\mathrm{D}^v$ for all $v \in G$. Constraint edges guarantee that each column is cut exactly once. They are the edges opposite to data edges:

$$\mathcal{E}_\mathrm{C} = \bigcup_{v \in V} \mathcal{E}_\mathrm{C}^v,$$

$$\mathcal{E}_\mathrm{C}^v = \{(u_{v,i+1}, u_{v,i}) | i = 1, \ldots, k-1\}.$$

The capacity of each constraint edge is set to infinity. This precludes more than one data edge being cut in each column $\mathcal{E}_\mathrm{D}^v$ of data edges over vertex v.

To see this, consider the changing assignments of vertices to S or T as we proceed from s to t on a column. The first vertex s belongs to S, and the last vertex t belongs to T. Thus, there must be at least one boundary in the progression where the membership changes. Moreover, the direction of the change must alternate (if you go from S to T, next time you have to come back from T to S.) Suppose that there is more than one boundary. Then the change across at least one of them must be from T to S. There is an edge going each way at this boundary. The edge from S to T is a constraint edge, and by the definition of a cut, it is cut. Therefore, if constraint edges have infinite capacities, there cannot be more than one boundary on the column. In Figure 4.1, constraint edges are depicted as dashed arrows, and none is cut.

Because of this, we can interpret a cut as a configuration of the MRF. Remember that the space $\mathcal{X}(G) = L^V$ is a set of assignments $X : V \ni v \mapsto X_v \in L$, and we want to find the assignment X that minimizes the energy (4.1).

Convention. For any cut, we interpret the cut as an assignment X such that $X_v = l_i$ for $v \in G$, where i specifies the vertex $u_{v,i}$ at the tail of the uniquely determined cut edge in the column over v.

If an MRF configuration, or an assignment X, assigns l_i to vertex v, the corresponding cut cuts the data edge $(u_{v,i}, u_{v,i+1})$ (or if $i = k$, $(u_{v,i}, t)$), whose cost is $h(v, l_i)$ by (4.2). Thus, with the capacities of the edges defined so far, the total cost of any cut is exactly the second sum in the energy (4.1) for the MRF configuration.

The rest of the energy is realized by the cost of the cut of a third kind of edge, the penalty edges.

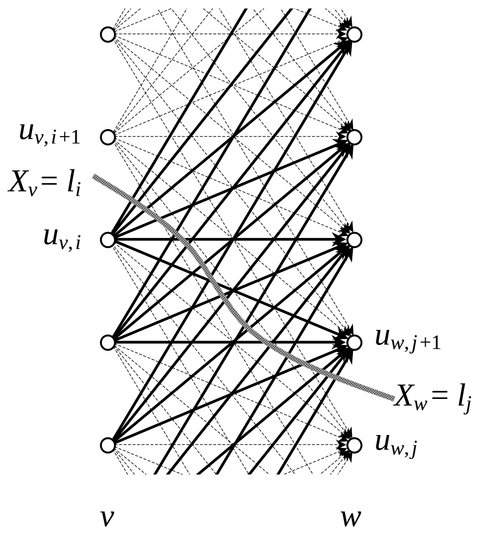


Figure 4.2: General penalty edges. Only the edges originating from the column over vertex v are shown. Edges going from "below" $u_{v,i+1}$ to "above" $u_{w,j}$ are cut, shown here as solid arrows.

4.2.4 Penalty Edges

Penalty edges go between columns:

$$\mathcal{E}_{\mathrm{P}} = \{(u_{v,i}, u_{w,j}) | (v,w) \in E;\ i,j \in \{1,\ldots,k\}\}.$$

Figure 4.1 shows the special case where the capacity of such an edge $(u_{v,i}, u_{w,j})$ is zero unless $i = j$, that is, where only horizontal edges exist. It is easily seen that the number of horizontal edges that are cut between columns over neighboring vertices v and w is proportional to the change $|X_v - X_w|$ of assignments at the vertices. For instance, if the assignment does not change, none of the horizontal edges between the two columns is cut. If they change by one, one would be cut, as at the leftmost pair of columns in the figure. This is the simplest case. Now, we consider the general case where each pair $(u_{v,i}, u_{w,j})$ of vertices in neighboring columns can have an edge between them with a nonnegative capacity (see

Figure 4.2.)

It is important to note that specifying an assignment at a vertex v, which determines the unique data edge that is cut in the column over v, also determines the membership of all vertices in the column in either S or T. Therefore, given assignments X_v and X_w at neighboring vertices v and w, the status (cut or uncut) of each edge between vertices in the columns over these vertices is determined. Thus, the sum of the capacities of cut edges between the two columns is determined by X_v and X_w.

Suppose we have label $X_v = l_i$ at vertex v and $X_w = l_j$ at vertex w. By the convention, this is expressed as a cut that cuts $(u_{v,i}, u_{v,i+1})$ and $(u_{w,j}, u_{w,j+1})$. As a side effect, various other edges are also cut. Among the edges going from the column over v to the one over w, edges of the form $(u_{v,a}, u_{w,b})$ with $a \leq i$ and $b > j$ are cut. Similarly, edges of the form $(u_{w,b}, u_{v,a})$ with $a > i$ and $b \leq j$ are cut. The sum of the capacities of these edges amounts to:

$$f(i,j) = \sum_{a=1}^{i} \sum_{b=j+1}^{k} c(u_{v,a}, u_{w,b}) + \sum_{a=i+1}^{k} \sum_{b=1}^{j} c(u_{w,b}, u_{v,a}). \tag{4.3}$$

Now, we *assume* that the sum represents the given penalty function $g(x)$ of the change:

$$f(i,j) = \tilde{g}(i-j) = g(l_i - l_j).$$

Although we do not even know if there exists a capacity map that makes the sum (4.3) depend only on the difference between i and j (we will see below that it does exist for any convex function $g(x)$,) here we just assume such capacities and derive a necessary condition.

Proposition. *Assume that the function $f(i,j)$, defined by (4.3) as the sum of edge ca-*

pacities between neighboring columns, depends only on the difference between i and j, and thus can be written as $f(i,j) = \tilde{g}(i-j)$. *Then the function* $\tilde{g}(x)$ *is convex.*

Proof. We take the second difference of $\tilde{g}$:

$$\begin{aligned}\frac{\delta^2 \tilde{g}(x)}{\delta x^2} &= \{\tilde{g}(x+1) - \tilde{g}(x)\} - \{\tilde{g}(x) - \tilde{g}(x-1)\} \\ &= f(i,j-1) - f(i,j) - f(i-1,j-1) + f(i-1,j),\end{aligned}$$

where $i-j=x$. When we use the definition (4.3) and simplify the sum, we get:

$$\frac{\delta^2 \tilde{g}(x)}{\delta x^2} = c(u_{v,i}, u_{w,j}) + c(u_{w,j}, u_{v,i}). \tag{4.4}$$

Since the right hand side is nonnegative, this shows that the function $\tilde{g}(x)$ must have a nonnegative second difference; thus $\tilde{g}(x)$ is convex. □

Conversely, suppose we are given a convex function $g(x)$. First, note that since L is finite, the range of the target function is bounded, and thus the range of possible x, which is the difference between the two values of the target function at neighboring vertices, is bounded. In other words, the behavior of $g(x)$ outside of this range does not make any difference to the problem. Specifically, we can assume without loss of generality that the second derivative of g is zero for large enough $|x|$.

We define the capacities of the edges by the appropriate second difference:

$$c(u_{v,i}, u_{w,j}) = \frac{g((i-j+1)l) - 2g((i-j)l) + g((i-j-1)l)}{2}, \tag{4.5}$$

where the right hand side is nonnegative because of $g(x)$'s convexity. Because of the manner in which it is defined, the capacity of an edge depends only on the

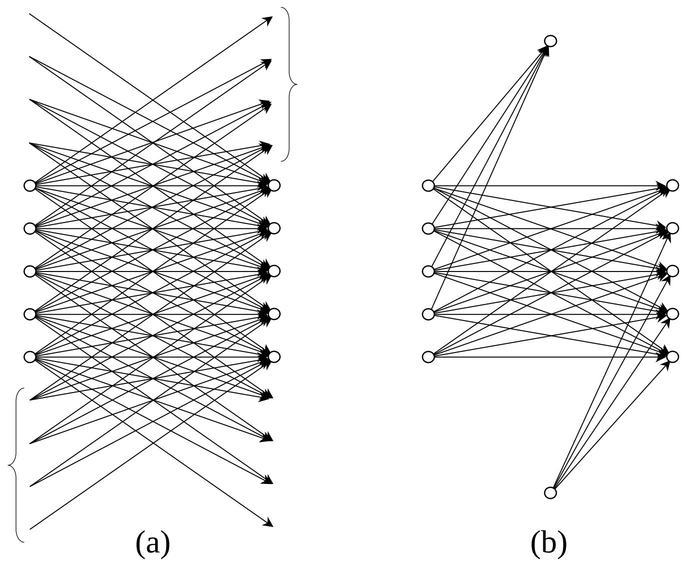


Figure 4.3: (a) Each penalty edge has a capacity that depends only on the label change. (b) Contributions from out-of-bounds edges are consolidated.

label change $i - j$ between the head and the tail of the edge. Also, because of the assumption that the second derivative of $g(x)$ vanishes for sufficiently large $|x|$, we can safely ignore edges with a large label difference.

Therefore, the function defined by the sum of these capacities depends only on the label difference, except for one detail: the boundary effect. That is, for this argument to work, we have to add the capacities of edges that have an end that is out of bounds (see Figure 4.3 (a).) The problem is that some edges whose capacities are necessary to make the sum depend only on the label change do not actually exist. Note that there are only a finite number of such edges because the capacity is zero for sufficiently large label differences. Therefore we can add these capacities to the capacities of other edges that do exist (Figure 4.3 (b)). That is, whenever vertices of the form $u_{v,i}$ with $i > k$ appear, we replace them by t and add the capacity to the existing edge; similarly, we replace $u_{v,i}$ by $u_{v,1}$ if $i < 1$.

In this way, for a given convex function $g(x)$, we can realize it up to a difference of a constant and a linear term in x. The linear term does not affect the optimization, since in the energy (4.1) $g(X_u - X_v)$ and $g(X_v - X_u)$ always appear together, canceling out linear terms. A constant term in $g(x)$ results in a constant in the energy, since there are a fixed number of vertices in the sum, and is therefore immaterial to the optimization. Thus, by finding a minimum cut, we can find a global MAP estimate of the MRF. We have proved:

Theorem. *The method applies if and only if the function $g(x)$ in the MRF energy (4.1) is convex.*

five

Dimension 1 Method

In this chapter, I describe our second graph-based method, which is used to seek a closed contour in one or more images. Using the minimum ratio cycle algorithm, we try to find a one-dimensional entity, a cycle, approximating a closed curve in the image. The basic idea is simple: we embed a graph in the pertinent space (for example the image plane) arranging the vertices regularly, with each vertex represent a position in the space and an edge an oriented line segment. In this setting, a path in the graph naturally approximates a curve in the space.

5.1 Finding Curves with a Minimum Energy

In this section, I discuss the difficulties we encounter when we try to minimize an energy for curves and closed contours in general. Then I explain how these problems are overcome.

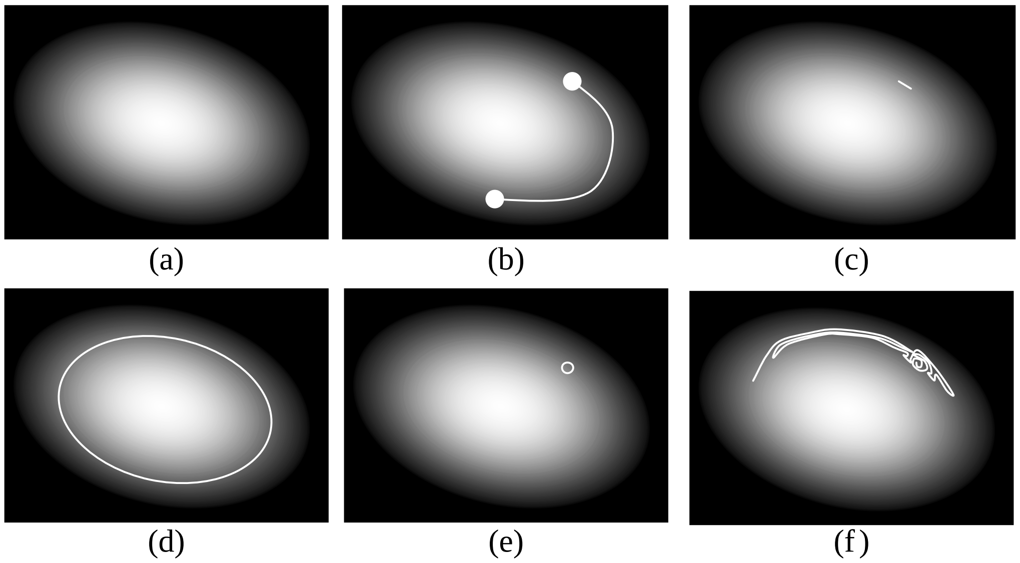


Figure 5.1: Finding curves with minimum energy.

5.1.1 Length Dependence

Suppose we have expressed the desirable properties of a curve as the integral of a local energy:

$$\int e(s)ds. \tag{5.1}$$

A nice property of this kind of energy is that we can use a shortest path algorithm to find a minimum energy path in a graph approximation; we assign the local energy to each edge and minimize its sum along the path. Unfortunately, it turns out that the problem needs to have some restriction placed on the kind of curves to be found, or the algorithm tends to find a trivial curve. The trouble is that often a minimum energy curve is also a “minimum curve.”

Let us for a moment suppose that the energy is positive. We can try, for instance, to find a curve with high gradient by minimizing

$$\int \frac{1}{|\vec{\nabla} I|^2} ds,$$

where $\vec{\nabla} I$ is the gradient of the image (Figure 5.1 (a).) We can impose a restriction by, for instance, fixing the two endpoints of the curve (Figure 5.1 (b),) which works fairly well. If, however, we try to find a minimum energy path with free ends, we will always find an edge with minimum local energy—maximum gradient—as a solution (Figure 5.1 (c).) The inherent difficulty here is that if we add up a positive local energy, part of a path always has less energy than the whole.

This happens also when we try to find a closed curve hoping to get a contour such as the one shown in Figure 5.1 (d). What we end up finding instead is a miniscule circle at the locus of the largest gradient (Figure 5.1 (e).)

What happens if we allow a negative energy? Let us consider a local energy $-|\vec{\nabla} I|$ instead of $|\vec{\nabla} I|^{-2}$. This time we have the opposite problem: the longer a path is, the less energy it tends to have. This is akin to looking for the longest curve—you can find as long a curve as you want (Figure 5.1 (f)).

We may try to balance these two situations by adding more than one kind of energy, say gradient and length. This sometimes works. However, such methods always have implicit dependence on length and often depend uncontrollably on the scale.

5.1.2 Energy Density

A partial remedy for this problem of length dependence is to minimize the energy density:

$$\frac{\int e(s)ds}{\int ds},$$

rather than the total energy. This eliminates the direct dependence on the length; a longer curve is no better than a shorter one if they have the same density. More generally, we can think of minimizing the ratio of two integrals:

$$\frac{\int e(s)ds}{\int f(s)ds},$$

where the length can be weighted by a positive function f.

Even in the case of an energy density, it turns out that there is still a tendency to favor a small segment, or a small circle in the case where we are looking for a closed contour. This is because we can always find part of a curve with a smaller density than the whole unless the local energy is constant on the curve. This difficulty is, however, much less problematic than the previous one, and seems to be manageable in practice, especially in the case of closed contours.

5.1.3 Oriented Contours and Regions

In the case of a closed contour, another way to avoid finding trivial contours is to take the direction of the curve into account when assessing the local energy. That is, we use the energy

$$\int e(s,\vec{t}(s))ds,$$

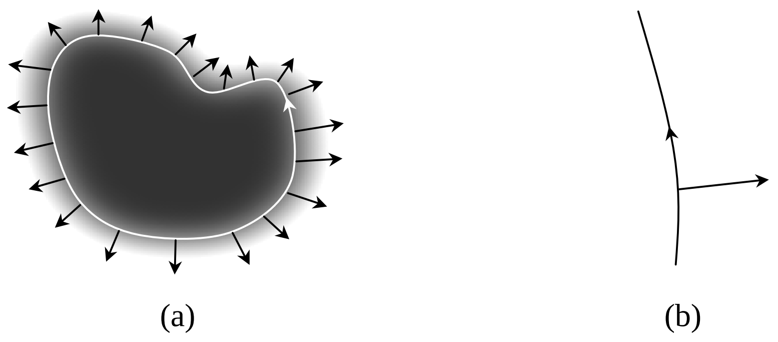


Figure 5.2: A gradient vector field and a closed contour. The cross product is minimum when the vector field is perpendicular to the tangent vector according to a fixed orientation (e.g. clockwise).

which depends on the tangent vector $\vec{t}(s)$ of the curve at the point, instead of the energy (5.1) that depends only on the position. By using the tangent vector, we can define an energy that cancels itself out when integrated around a small cycle. The most natural instance of this is

$$e(s,\vec{t}) = \vec{\nabla} I(s) \times \vec{t}(s), \tag{5.2}$$

where "$\times$" is the cross product of the two vectors (see Figure 5.2.) This way, the modulus of the integral of a small cycle will be small, because two segments going in opposite directions at nearby points tend to cancel each other.

In addition to this benefit, this form of energy has a great advantage. A slightly more general version of (5.2) is

$$e(s,\vec{t}) = \vec{v}(s) \times \vec{t}(s), \tag{5.3}$$

where a general vector field $\vec{v}$ is integrated along the contour. This form of energy allows us to use region energy that depends on properties of the region that is surrounded by the closed contour. This follows from Stokes' theorem. It states (in this case) that, for any vector field $\vec{v}$, an integral over the boundary $\partial\mathcal{R}$ of a region $\mathcal{R}$ can always be rewritten as an integral over the region:

$$\int_{\partial\mathcal{R}} \vec{v} \times \vec{t}\, ds = \int_{\mathcal{R}} \operatorname{div} \vec{v}\, dxdy,$$

where $\operatorname{div}\vec{v} = \vec{\nabla} \cdot \vec{v}$ is the divergence of the vector field $\vec{v}$. Note that both the region and the boundary must be considered oriented.

Now, the crucial observation is this: in a topologically simple space such as this one, there always exists a vector field whose divergence is the given function. In other words, for any function $e(x,y)$, we can always find a vector field $\vec{v}(x,y)$ such that

$$\operatorname{div}\vec{v}(x,y) = e(x,y)$$

and, using Stokes' theorem, convert a region integral of e over an oriented region $\mathcal{R}$ into a contour integral of $\vec{v} \times \vec{t}$ over the boundary of the region:

$$\int_{\mathcal{R}} e(x,y)\, dxdy = \int_{\partial\mathcal{R}} \vec{v} \times \vec{t}\, ds.$$

On $\mathbf{R}^2$, the vector field $\vec{v} = (\vec{v}_x, \vec{v}_y)$ can for instance be given by the following integrals:

$$\begin{aligned} \vec{v}_x(x,y) &= \frac{1}{2}\int_0^x e(x',y)dx', \\ \vec{v}_y(x,y) &= \frac{1}{2}\int_0^y e(x,y')dy'. \end{aligned}$$

There is a choice of constant functions that can be added to the components of this vector field, but the choice does not affect the value of the integral. Indeed, we can add any divergence-free vector field (i.e., the gradient of some function) and still have the same integral.

5.2 Globally Optimal Regions and Boundaries

As I discussed in the previous section, we have many choices when we try to find curves with an optimal energy. The curve can be either open or closed. The local energy can be oriented or otherwise. We can optimize either the energy integral itself or the density thereof.

As long as we are considering an additive energy, the search for an open contour has the inherent difficulty of length dependence (section 5.1.1.) Even when we consider an energy density (section 5.1.2,) the shortest segment with the minimum energy has the minimum energy density. Thus, we need more constraints. One such constraint would be to fix the endpoints, which would require a way to choose the points. Looking for a closed contour without orientation seems to be plagued by the trivial cycle problem.

Indeed, among these possibilities, the search for an oriented closed contour with an optimal energy density seems to be the only case that has a nontrivial solution without an initialization such as specifying endpoints.

In addition to this benefit, the problem has an additional advantage, discussed in section 5.1.3; moreover, there is an efficient algorithm to solve the problem—the minimum ratio cycle algorithm.

5.2.1 Minimum Ratio Cycle

The minimum ratio cycle problem is defined on a graph $G = (V, E)$ with two cost functions: an integral cost $c : E \to \mathbf{Z}$ and a nonnegative integral time $t : E \to \mathbf{Z}^{>0}$. The task is to find a cycle with the smallest ratio of its cost to its time. A cycle in

G is a series $C = (e_1, \dots, e_k)$ of edges that forms a closed path, i.e., the head of e_i and the tail of e_{i+1} coincide for $i = 1, \dots, k-1$ and the head of e_k and the tail of e_1 coincide. For a given function on E, we use the same symbol for its sum on the edges in a cycle, e.g.,

$$c(C) = \sum_{e \in C} c(e),$$
$$t(C) = \sum_{e \in C} t(e).$$

We define the ratio weight by

$$w(C) = \frac{c(C)}{t(C)}.$$

Then, the problem is to find a cycle C with minimum ratio weight $w(C)$. There are algorithms (Lawler[53]; Dantzig, Blatner, and Rao[21]; Meggido[60]) that solve this problem in polynomial time. In section 6.4.2 I will describe in more detail the particular version that is used in this thesis.

5.2.2 Finding Optimal Regions and Contours

We use the minimum ratio cycle algorithm to *maximize* an energy of the following form:

$$E(I, \mathcal{R}) = \frac{\left|\int_{\mathcal{R}} e(x,y)\, dx\, dy\right|}{\int_{\partial\mathcal{R}} f\, ds}, \tag{5.4}$$

where e is any real-valued function on $\mathbf{R}^2$, which generally reflects the contribution of the image to the energy, and f, integrated by the arc length parameter s, is any positive real-valued function.

As noted in section 5.1.3, the numerator of (5.4) can always be rewritten as an integral over the boundary $\partial\mathcal{R}$ of the region $\mathcal{R}$:

$$E(I, \partial\mathcal{R}) = \frac{\left| \int_{\partial\mathcal{R}} \vec{t} \times \vec{v}\, ds \right|}{\int_{\partial\mathcal{R}} f\, ds} \tag{5.5}$$

where $\vec{t}$ is the tangent vector to the boundary, and $\vec{v}$ is defined by the equation $\vec{\nabla} \cdot \vec{v} = e$.

Thus we can convert freely between (5.4) and (5.5); therefore we can use any function e that measures the merit of the region $\mathcal{R}$, any vector field $\vec{v}$ to be integrated around the boundary $\partial\mathcal{R}$ to measure the desirability of the contour, or any combination thereof, as the criterion for an optimal region and its boundary.

The denominator is a (possibly weighted) measure of the length of the boundary, and has the effect of damping the scaling behavior of the energy, which would otherwise have a strong preference for large regions. It also functions as a boundary smoothing term, as follows. If e and f were unity we would be maximizing the area over the length, and fixing the length of the boundary would produce a disc as the solution to the optimization problem. This is also the solution to an active contour model with a fixed length and a smoothing term that is the square of the curvature. In general, the effect of the dependence on area divided by that on length will be to produce smoother boundaries.

We define the solution to the optimization problem as the global *maximum* of E over all regions $\mathcal{R}$. Note that by assigning two energies to each region, $\pm E(I, \mathcal{R})$, and then minimizing over all such energies, we achieve the same solution. That is, by minimizing, we actually find the largest modulus. This can be viewed as an assignment of two orientations to each region, and then a minimization over all oriented regions. Removing the modulus signs and minimizing over all oriented boundaries is then equivalent to the original maximization problem over un-oriented regions.

Thus, by using a standard embedded graph in an image, we can translate the minimizing problem over all oriented boundaries into a minimization over all directed cycles in the graph, and the energy becomes exactly what is minimized by the minimum ratio cycle algorithm.

The form of equation (5.5) allows us elegantly to include boundary as well as region information in our model. We can add to $\vec{v}$ any other vector field $\vec{w}$ and still compute the global optimum.

5.3 General Formulation

Here I include a general formulation of the problem using the language of Riemannian geometry, which more easily generalizes to higher dimensions as a notation. I refer the reader to [18] for an introduction to this language.

We define a closed contour s in an orientable Riemannian manifold M (such as $\mathbf{R}^2$) by an embedding

$$s : S^1 \to M.$$

That is, s is a smooth map that induces a diffeomorphism between the standard (oriented) one-dimensional sphere S^1 and its image $s(S^1)$ with the induced topology and differential structure from M. In particular, the contour does not cross itself (otherwise the topology would be different from that of S^1.)

There exists a one-form ω_s on S^1 such that the length of the contour in M is defined by the integral

$$\int_{S^1} \omega_s.$$

It is uniquely determined by $s^*g = \omega_s \otimes \omega_s$ and the orientation of S^1, where g is the metric on M. The Hodge star $*_s$ on S^1 is defined with respect to the metric on M pulled back by s; for a function f on S^1, $*_s f = f\omega_s$, and for a one-form A, $*_s A$ is defined by $(*_s A)\omega_s = A$ (thus $(*_s)^2$ is an identity.) Suppose μ is a function or a one-form on M that has its value in some vector space endowed with an inner product $\langle \cdot, \cdot \rangle$. We define

$$|\mu|^2_{s,0} = *_s\langle s^*\mu, *_s s^*\mu\rangle, \text{and} \tag{5.6}$$

$$|\mu|^2_{s,1} = \langle s^*\mu, *_s s^*\mu\rangle, \tag{5.7}$$

which are a function and a one-form on S^1, respectively. When μ is a one-form, $|\mu|^2_{s,0}$ depends not only on the point that s goes through but also on its direction at that point, while if it is a function, $|\mu|^2_{s,0}$ depends only on the position. The one-form $|\mu|^2_{s,1}$ behaves in the same way.

We define the energy functional

$$E(s) = \frac{\int_{S^1} s^*A}{\sum_i \int_{S^1} |f_i|^2_{s,1}/|g_i|^2_{s,0}}, \tag{5.8}$$

where A is a one-form on M and f_i and g_i are each either is either functions or one-forms on M.

Let us denote by D the standard two-dimensional disk and by r an embedding of D into M that coincides with s on $S^1 = \partial D \subset D$. Then, for any one-form A, Stokes' theorem states that

$$\int_D r^* dA = \int_{S^1} s^*A.$$

When the second cohomology group $H^2(M;\mathbf{R})$ of M is trivial (as in the case of a Euclidean space,) any closed two-form F on M is exact, i.e., if $\mathbf{d}F = 0$, there exists a one-form A such that $F = \mathbf{d}A$.

In particular, if M is two-dimensional, all two-forms are closed. Thus, if $H^2(M;\mathbf{R})$ vanishes, we can find for any function e on M a one-form A such that $*e = \mathbf{d}A$, where $*$ is the Hodge star operator on M. This corresponds to the special case that was described in the previous section, where $A = *\mathbf{d}I$, corresponding to the vector field $\vec{v}$.

When, as in the case of the stereo and motion segmentation in the next chapter, M is a match space of two or more image planes $\check{I}_1, \ldots, \check{I}_n$, there is a natural projection from M to each image plane:

$$p_i : M \to \check{I}_i.$$

Then we can define a two-form F on M by a linear combination of pullbacks of two-forms on the images:

$$F = \sum_i a_i p_i^* F_i.$$

Then F is closed, and since M is a Euclidean space, exact.

six

Variations and Details

Here I describe applications of the methods explained in the previous chapters in specific areas, and show the results of experiments. I begin with a simple use of the MRF optimization for image restoration and proceed to a more complex variation for binocular stereo. Then I describe a variety of image segmentation and region extraction applications, using both the minimum cut method and the minimum ratio cycle method.

6.1 Image Restoration

Before going into computer vision problems, first I give a simple example of MRF optimization. In image restoration, we are given an image that is corrupted by noise. The goal is to recover the "true" image from the noisy one. Here, we assume a simple statistical model and use an MRF to find the image most likely to have given the corrupted one. The image formation model here is a simple

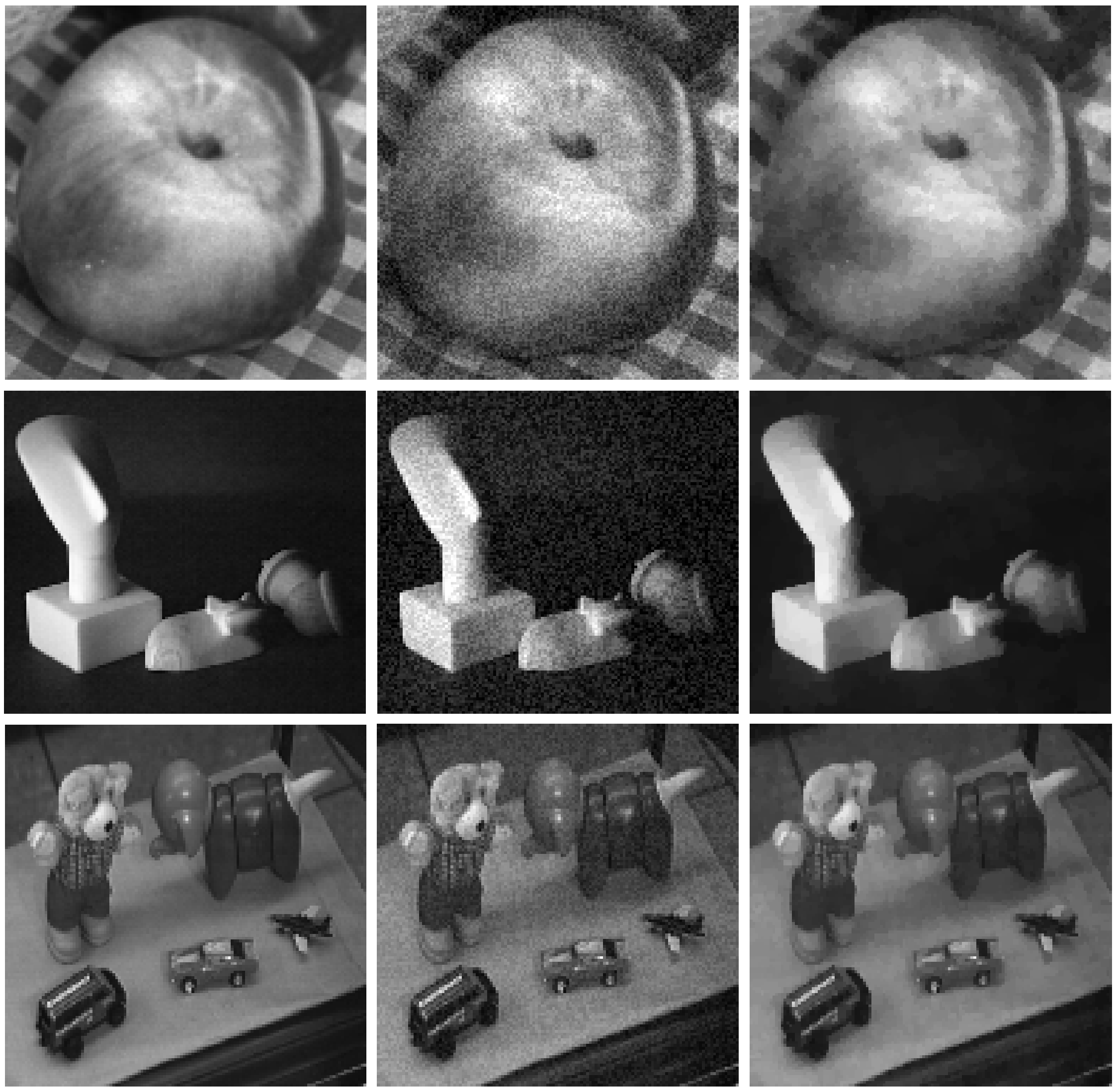


Figure 6.1: Each row shows an example of image restoration; the original image (left column), the noisy image (middle), and the restored image (right). Note that the restoration does not blur the intensity edges.

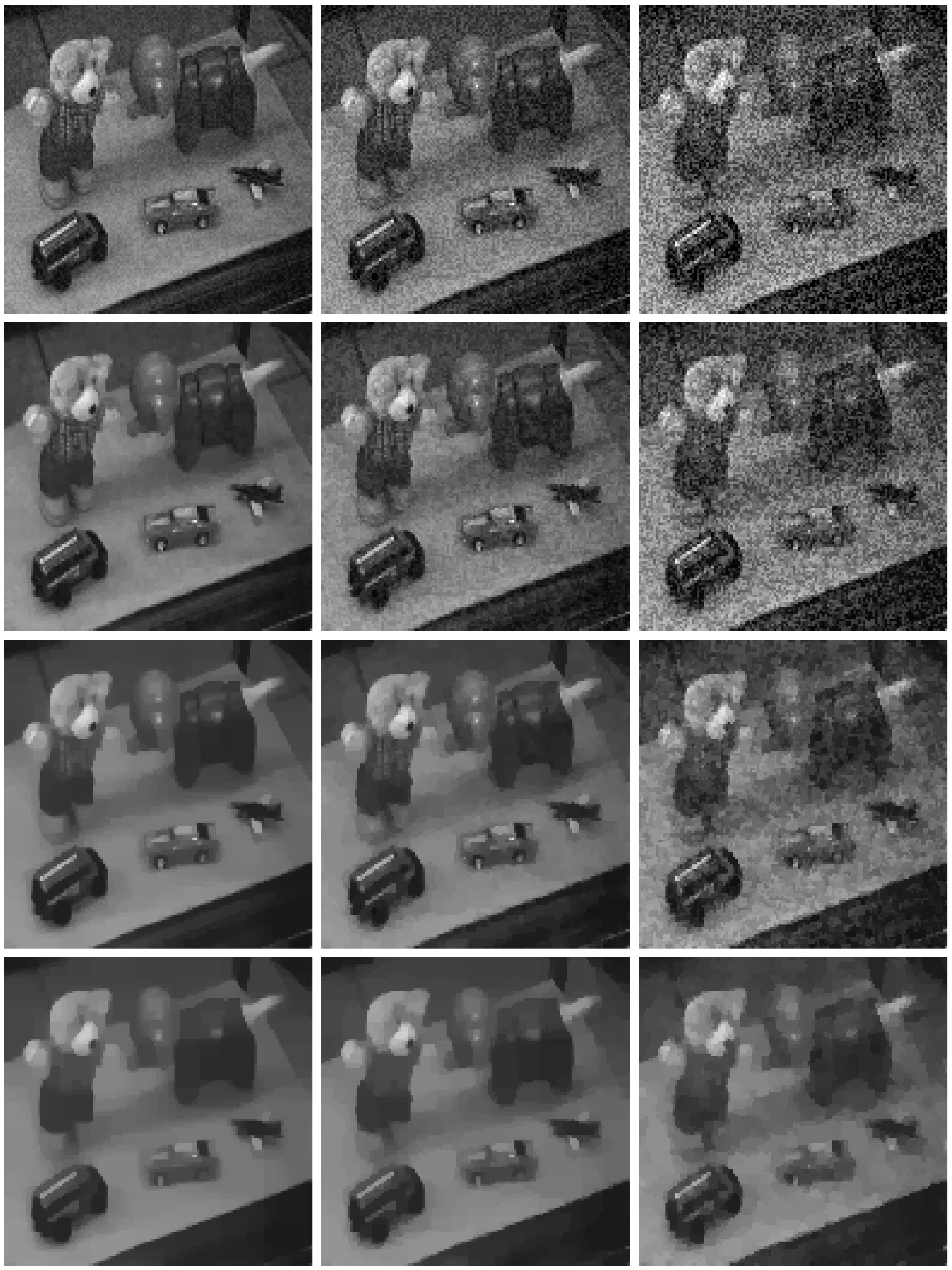


Figure 6.2: Comparison of three restorations of images with three noise levels. Each column shows the input (top row) and the restoration using $\mu = 6$ (second row), $\mu = 20$ (third row), and $\mu = 40$ (bottom), where $g(x) = \mu|x|$.

pixel-wise Gaussian noise model. That is, the probability $P(I|I_{\mathrm{t}})$ that the image I is obtained by corrupting the true image I_{t} is given by

$$P(I|I_{\mathrm{t}}) \sim \prod_{x,y} \mathrm{e}^{-\{I(x,y)-I_{\mathrm{t}}(x,y)\}^2}.$$

The prior model is a smoothing model:

$$P(I_{\mathrm{t}}) \sim \prod_{x,y} \prod_{x',y'} \mathrm{e}^{-\mu|I_{\mathrm{t}}(x,y)-I_{\mathrm{t}}(x+x',y+y')|},$$

where x' and y' give appropriate relative neighbor coordinates.

The corresponding MRF model is defined by i) an image graph $G = (V, E)$ that has a vertex for each pixel and an edge for each pair of neighboring pixels, ii) a label set $L = \{0, \ldots, 255\}$ that consists of gray-scale values, and an energy:

$$E(X) = \mu \sum_{(u,v)\in E} |X_u - X_v| + \sum_{v \in V} \{I(v) - X_v\}^2,$$

where the configuration X of the MRF corresponds to the true image I_{t}.

Among convex functions, the linear function on the magnitude of the change (i.e., $g(x) = |x|$) is most preferable as it penalizes large discontinuities least.

Some results are shown in Figure 6.1. Each row shows an example of image restoration; the original image (left column), the noisy image (middle), and the restored image (right). Note that the restoration does not blur the intensity edges.

Figure 6.2 shows a comparison of three restorations of images with three noise levels. The original image is the third in Figure 6.1. Each column shows the input (top row) and the restoration using three smoothing constants; $\mu = 6$

(second row), $\mu = 20$ (third row), and $\mu = 40$ (bottom).

For the maximum-flow algorithm, we used a standard push-relabel method with global relabeling (see Cherkassky and Goldberg[17].)

The algorithm has an asymptotic complexity of $O(|V|^3)$. Although there are other algorithms that are faster in an asymptotic sense (e.g., the algorithm in Goldberg and Tarjan[34] runs in $O(|V||E|\log(|V|^2/|E|))$ time), they are generally more complex and in the range of the graph sizes that is relevant here, the simpler algorithm seems to be faster. In the implementations, the global relabeling heuristic [17], which does not improve the asymptotic complexity at all, is much more important, since it improves the actual running time by a factor of ten or more.

6.2 Stereo

Next, I describe how a binocular stereo algorithm can be implemented by specializing the graph used in the above MRF method so that:

1. half occlusions are allowed detected,

2. disparity discontinuity along the epipolar line in one eye *always* corresponds to an occluded region in the other eye, and

3. in its prior, epipolar lines need not be statistically independent.

As explained in section 3.2.1, the parameter space for stereo is the space of surfaces in the product space $\check{I}_\mathrm{L} \times \check{I}_\mathrm{R}$ restricted by the epipolar constraint (the match space). The match space has a natural coordinate system (l, r, y), where y parametrizes epipolar lines, and l and r are the coordinates on the epipolar lines

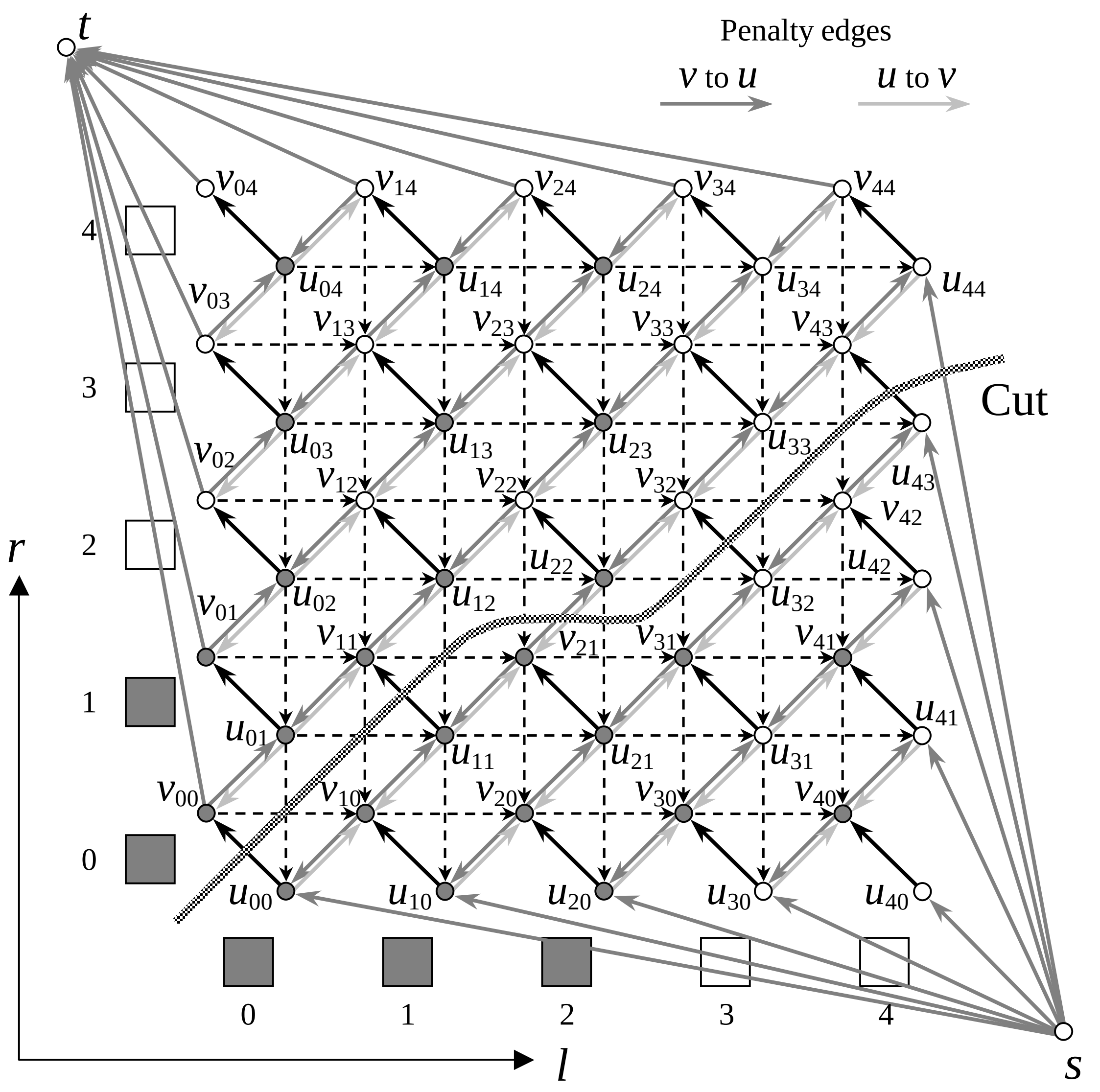


Figure 6.3: An epipolar slice of the graph representing the stereo model. The full graph is naturally represented in three dimensions, with the third axis parametrizing the epipolar lines. A cut of the graph can be thought of as a surface that separates the two parts; it restricts to a curve in an epipolar slice. The optimal cut is the one that minimizes the sum of the capacities associated with the cut edges. In this example, the cut shown yields the matches $(l, r) = (0,0), (1,1), (3,2)$, and $(4,3)$; the cut also detects an occlusion at gray (white) pixel 2 (4) in the left (right) image.

in the left and right images, respectively. We represent half-occlusions, or appearances of scene locations in only one of the two images, by a match surface that is perpendicular to the l or r axis.

We convert the (l,r,y) coordinate system into a "cyclopean" coordinate system (d,t,y), where $d = l - r$ and $t = l + r$. Because of the monotonicity constraint, the surface in this representation has a unique point for each (t,y) pair, i.e., it is the graph of some function on the (t,y) plane that gives a value d—the disparity—at each point. This enables us to consider the problem as an MRF problem. We define the MRF by considering a graph embedded in the t-y plane and a label set consisting of integral disparity values d. The graph has a vertex for each pair of integral t and y in the range, and has the standard four-neighbor structure: the vertex for (t,y) is connected to the vertices for the coordinates $(t+1,y)$, $(t-1,y)$, $(t,y+1)$, and $(t,y-1)$. Equation (4.1), the energy functional reads:

$$E(X) = \sum_{(t,y),(t',y')\text{neighbors}} g(d(t,y) - d(t',y')) + \sum_{(t,y)} h(t,y,d(t,y)). \tag{6.1}$$

For the prior, we again use the penalty function $g(x) = |x|$. For the image formation model, we use various functions $h(t,y,d)$ that give a measure of difference between the points $(\frac{t+d}{2},y)$ in the left image and $(\frac{t-d}{2},y)$ in the right image (see section 6.2.2 below.)

Then, we use the method discussed in section 4.2 for this problem and construct a graph. Figure 6.3 shows an epipolar slice (i.e., fixed y) of the graph. We can see that this is a version of the graph shown in Figure 4.1, but tilted by 45 degrees. The constraint edges have slightly different connectivity in order to enforce the ordering constraint, or rather its local version, the monotonicity constraint. The statistical dependence of the epipolar lines, i.e., the smoothing across them, is implemented by penalty edges bridging epipolar slices.

6.2.1 Graph Structure

Although the basic principle is that of the MRF method discussed in chapter 4.2, the graph used here has some modifications to accommodate the specific situation.

First, as already mentioned, the constraint edges enforce the monotonicity constraint. In Figure 6.3, the constraint means that the cut must "go" from left to right between angles of 0 and 90 degrees.

Assuming that the l and r coordinates are integral, half of the integral disparity values are impossible for any given integral t (e.g., if $t = l + r$ is even, then $d = l - r$ must be even, too.); there are no corresponding data edges for these values. As can be seen in the figure, this classifies the vertices into two kinds—a vertex other than s or t is either the tail of a data edge or the head of one. Moreover, since there exists a unique data edge for each possible combination of l, r, and y, there exists one of each kind of vertex for each such triple. We denote the vertex at the tail of the data edge corresponding to (l,r,y) by $u_{l,r,y}$ and the one at the head by $v_{l,r,y}$ (the y coordinate is omitted in the figure.) This classification also gives rise to two kinds of penalty edges—the ones going from u to v and the ones going from v to u. This difference turns out to have a geometric significance, which now I explain.

Occlusion vs. Tilted Surface In Figure 6.3, the pixel 2 in the left image does not match any pixel in the right image. The disparity changes from $d = 0$ at left-pixel 1 to $d = 1$ at left-pixel 3. Figure 6.4 (a) shows a close-up. Pixel 2 in the left

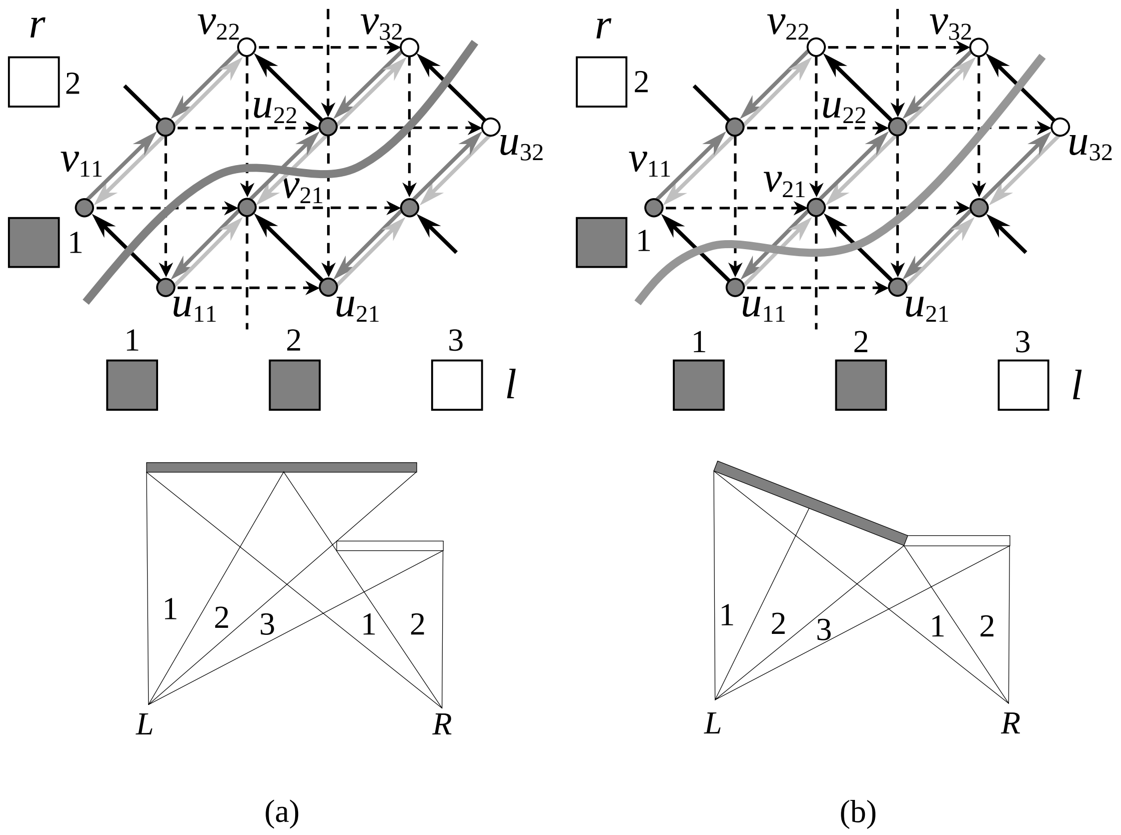


Figure 6.4: (a) A close-up of the Figure 6.3. The left-pixel 2 (the middle) does not have a matching right-pixel, i.e., it is occluded. (b) Another possibility, where the left-pixel 1 and 2 match the same right pixel; this happens when the surface is tilted. Note the different kinds of the penalty edges are cut in the two cases.

image does not have a matching right pixel, i.e., it is occluded; the scene point corresponding to this pixel in the left image cannot be seen from the right eye. To realize the same disparity change, there is another possibility: pixel 2 in the left image can match pixel 1 in the right. This represents the case where the surface is tilted (see Figure 6.4 (b).) Comparing the graph cuts in the two cases, we notice that in (a), a penalty edge going from v to u (shown in dark gray) is cut whereas in (b), a penalty edge going from u to v (shown in light gray) is cut. Using this fact, we can control the tendency of the algorithm to find one of the two cases by changing the weights for the two kinds of penalty edges.

6.2.2 Feature Selection

In order to find corresponding points in the two images, an algorithm must have some notion of similarity, or likelihood that points in each image correspond to one another. To estimate this likelihood various features are used, e.g., intensity difference, edges, junctions, and correlation. Since none of these features is clearly superior to the others in all circumstances, using multiple features is preferable to using a single feature, if one knows which feature, or which combination of features, to use when. Unfortunately, features are difficult to cross-normalize. How can we compare the output from an edge matching with one from a correlation matching? We would like not to have to cross-normalize the outputs of the feature matchings, and still be able to use multiple features. Here, we use a consequence of the monotonicity constraint to select an optimal feature or combination of features for each set of mutually exclusive matching choices.

In the energy functional (6.1), the local feature energy function $h(t,y,d)$ gives a measure of difference between the points $(\frac{t+d}{2},y)$ in the left image and $(\frac{t-d}{2},y)$ in

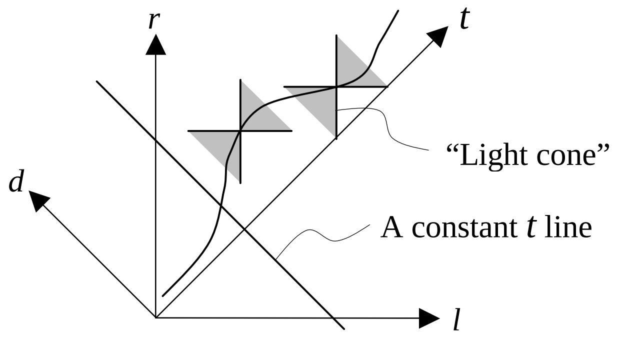


Figure 6.5: An epipolar slice of the match space. The matching surface appears as a curve here. The monotonicity constraint means that this curve crosses every constant t line once.

the right image. We assume it gives a nonnegative value; a smaller value means a better match. In what follows, the y coordinate will be omitted. Also, note that these functions of course depend on the images, although the notation does not show this explicitly.

Suppose we have a finite set $\mathcal{H}$ of local feature energy functions. On what basis should we choose from the set? Different features are good in different situations. For instance, edges and other sparse features are good for capturing abrupt changes of depth and other salient features, but can miss gradual depth change that can instead be captured by using dense features. What one cannot do is to choose functions at each point in the match space; the values of different local energy functions are in general not comparable. In general, the same local function must be used over the set from which a selection is made. In other words, across these sets of selections, different functions can be used. Then, what is the set of selections? Figure 6.5 shows an epipolar slice of the match space. The surface that represents the matching appears as a curve here. In this figure, the monotonicity constraint means that the tangent vector of the curve must reside in the "light cone" at each point of the matching curve. This implies

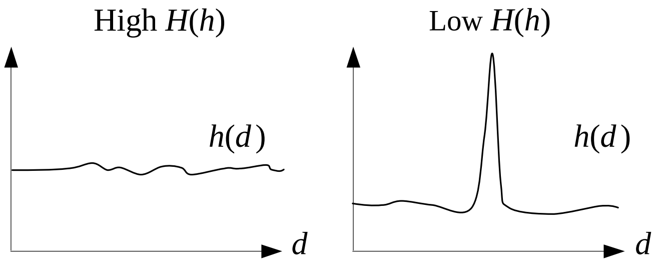


Figure 6.6: The functional H on function h. It measures the degree of concentration of the value of h.

that the matching curve crosses each constant t line at exactly one point. This means that on each such line the matching problem selects one point (the match) from all the points on the line. Thus we can choose one particular local energy function on this line and safely choose a different one on another line. In the following, I will call these lines the ″selection lines.″

The partition of the match space into selection lines is minimal in the sense that, for any sub-partition, the selection of the energy function cannot be local to each partition. There are, however, other minimal partitions with this local-selection property. For instance, the match can be partitioned into other “space-like” lines with an l to r tilt different from $-1 : 1$, as long as the ratio is negative.

Selection Rule As we have said, on each selection line, we are free to choose any local energy function. Note that the information that we can easily utilize for the selection is limited. For instance, we cannot use any information concerning the matching surface that is eventually selected, as that would lead to a combinatorial explosion. Here, we employ a least “entropy” rule to select the energy function. It chooses the energy function that is most “confident” of the match on each selection line. After all, an energy function that does not discriminate between one match and another is of no use. Going to the other extreme, when we have ground truth, an energy function that gives the true match the value zero and every other match the value positive infinity is obviously the best; the energy function knows which match to choose with certainty. This intuition leads us to evaluate how "sure" each energy function is.

Let us define an "entropy" functional for a positive-valued function h on $\{d = D_0, D_0+1, \ldots, D_1\} \times \{t\}$ by:

$$E_t(h) = \sum_{d=D_0}^{D_1} h(d,t),$$
$$H_t(h) = -\sum_{d=D_0}^{D_1} \frac{h(d,t)}{E_t(h)} \log \frac{h(d,t)}{E_t(h)}.$$

This functional H_t gives a measure of the degree of concentration of the function h: it is smaller when h is more concentrated (see Figure 6.6.) The more peaked the function, the lower the value of the functional. We use this functional to choose a preferred local energy function for each selection line. To use this functional for our purposes, where we need a dipped function rather than a peaked one, we invert the function and feed the result to the functional.

Thus, for each selection line, we choose the function h with the least value of $H_t(h^{\max_t} - h)$, where $h^{\max_t}$ is the maximum value of h on the selection line corresponding to the coordinate value t:

$$h_t = \arg\min_{h \in \mathcal{H}} H_t(h^{\max_t} - h).$$

This selection rule prefers a function that has a distinguished dip, which means, in our situation, one or few disparity values that have an advantage over other values. This method of selection allows us to avoid irrelevant measures locally and ensures the most confident selection of the disparity on each selection line.

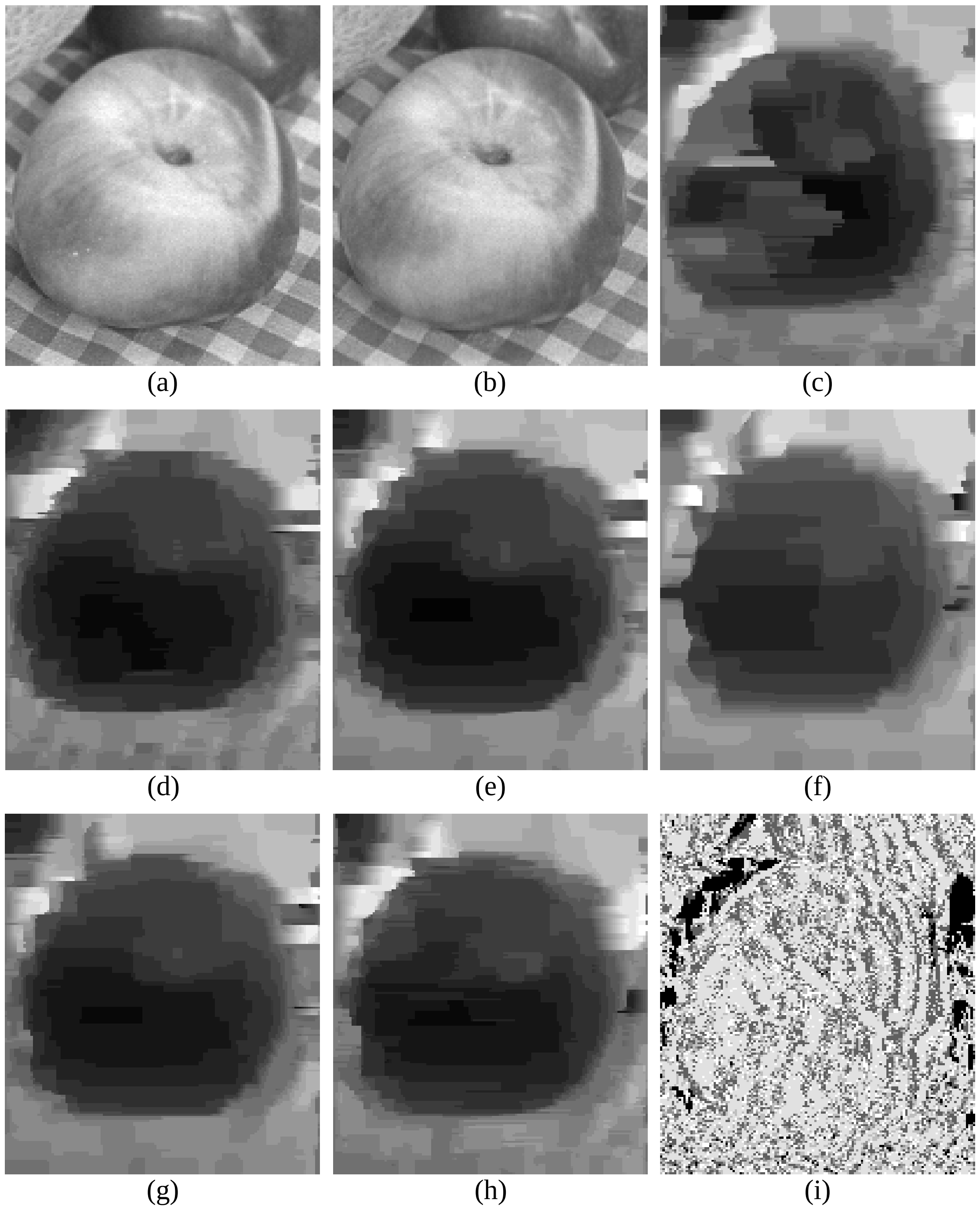


Figure 6.7: (a),(b) A sample image pair "Apple." Results (disparity maps) are shown using different local energy functions (c), (d), (e), (f), (g), and minimum-entropy selection from the five energies (h). The gray level in (i) shows which of five energy functions is used in (h) at each point of the left image.

6.2.3 Results

The following features are used for the experiments:

1. **Intensity**. This is a simple squared difference between the points, i.e.,

$$h_{\mathrm{I}}^2(t,y,d) = \{I_{\mathrm{L}}(\frac{t+d}{2},y) - I_{\mathrm{R}}(\frac{t-d}{2},y)\}^2.$$

2. **Wavelet edge**. The derivative of Gaussian wavelet that detects an edge in the vertical direction at various scales:

$$h_{\mathrm{E}}^{s}(t,y,d) = \left| W_s I_{\mathrm{L}}(\frac{t+d}{2},y) - W_s I_{\mathrm{R}}(\frac{t-d}{2},y) \right|,$$

where

$$\begin{aligned} W_s I(x,y) &= I * \psi_s(x,y), \\ \psi_s(x,y) &= s^{-1}\psi(s^{-1}x, s^{-1}y), \\ \psi(x,y) &= 2\pi^{-1}\exp(x^2 - y^2)x. \end{aligned}$$

See Mallat [56] Chapter 6 for the details of multi-scale edge detection.

3. **Multi-scale edges consistent across the scale**. This is a measure of the presence of an edge across scales.

$$h_{\mathrm{E}}(t,y,d) = \left| \sum_s W_s I_{\mathrm{L}}(\frac{t+d}{2},y) - \sum_s W_s I_{\mathrm{R}}(\frac{t-d}{2},y) \right|.$$

In Figure 6.7, a comparison of the results for a sample image pair ((a),(b); 135 $\times$ 172 pixel 8-bit gray-scale images) using these energy functions is shown. The

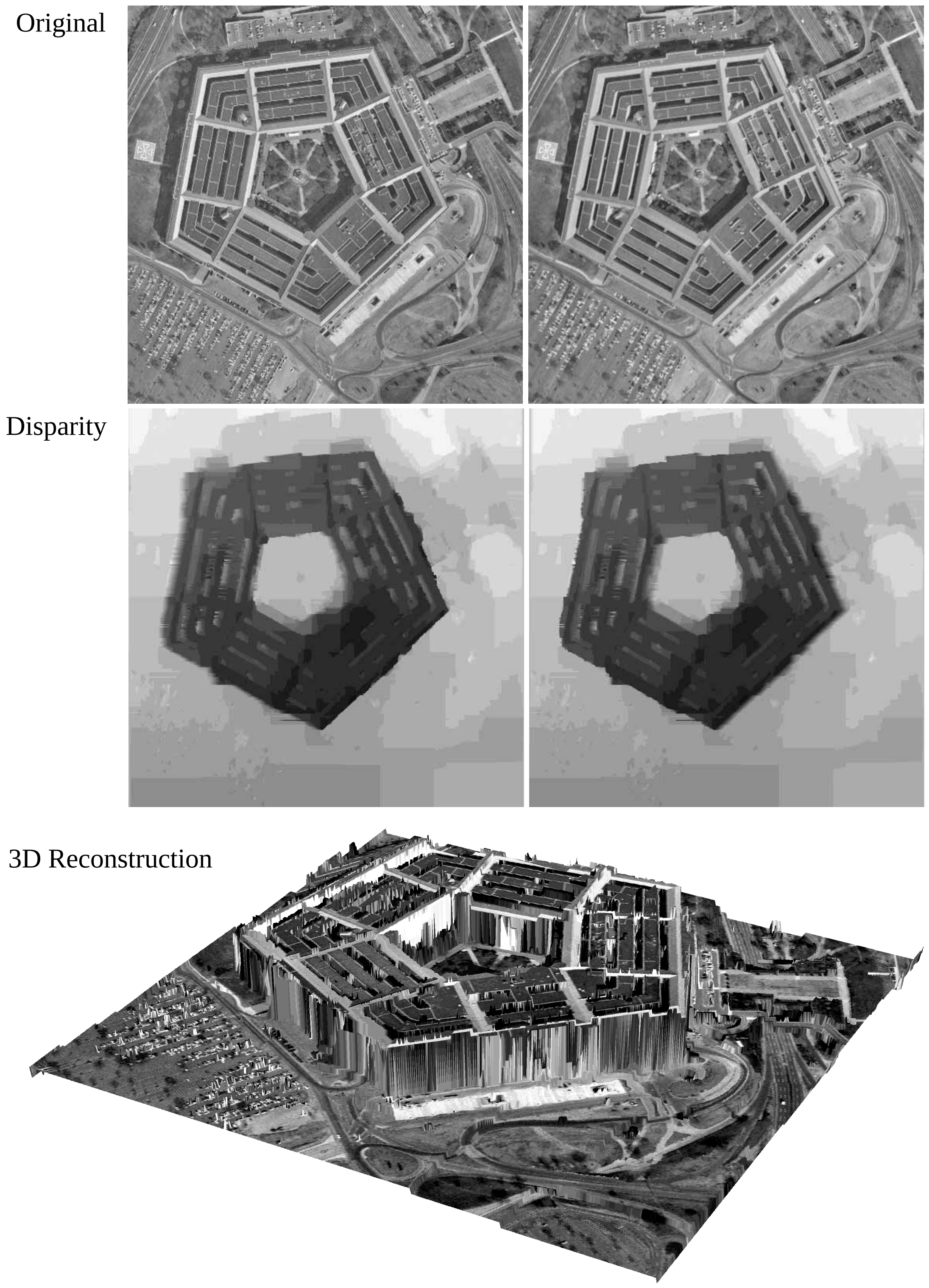


Figure 6.8: Stereo pair "Pentagon" (508 $\times$ 512 pixel 8-bit grayscale images,) disparity maps for both images, and a 3D reconstruction from the disparity.

results (disparity maps) are shown using the intensity square difference h_{I}^2 (c); the wavelet edge features $h_{\mathrm{E}}^s(t,y,d)$ with scale $s=1$ (d), $s=2$ (e), and $s=4$ (f); the multi-scale edge $h_{\mathrm{E}}(t,y,d)$ (g) (the square difference of the sum of the wavelet coefficients for $s=1,2,4$); and the minimum-entropy selection from the five energies (h). The Intensity feature h_{I}^2 (c) gives the poorest result in this example. Wavelet edges for $s=1,2,4$ (d), (e), and (f) are better, yet with a black artifact on the upper right, also present with the multi-scale edge (g). The gray-scale image (i) shows which of the five energy functions is used in (h) at each point of the left image. A black point represents an occluded point, where no match was found, resulting in no corresponding t defined for the l-coordinate. Other gray values are in the order (c) to (g), i.e., darkest: intensity h_{I}^2, lightest: multi-scale edge $h_{\mathrm{E}}(t,y,d)$.

Figure 6.8 shows a stereo pair "Pentagon" (508 $\times$ 512 pixel 8-bit grayscale images,) disparity maps for the left and right images, and a 3D reconstruction from the disparity map.

6.3 Region Segmentation Using the MRF

In an image, objects and their boundaries manifest themselves as regions and their 1D boundaries. Segmentation is the process of identifying groups of pixels that belong to different objects. As discussed in section 3.1, there are generally two approaches to the segmentation problem: we find either regions or their boundaries.

Among our methods, MRF optimization is suited to region segmentation, where we assign to each pixel a label that identifies the group to which the pixel belongs. An important question is how the number of labels is determined, since this must be fixed before the MRF optimization process can take place. In this section, I introduce a method that uses junction information to decide the number of labels and a criterion to assign one of them to each pixel.

We segment an image by assigning to each pixel a *level*. Each level has an associated gray-scale value, and we assign to each pixel the level with the gray-scale value closest to that of the pixel, subject to a smoothness constraint.

6.3.1 Determining the Set of Levels

We first determine the number of levels and their associated gray-scale values. We take junctions (e.g., corners, T-junctions) as strong indicators of a region boundary. T-junctions and corners often arise from overlapping surfaces, which we wish to obtain as distinct segments in the image. Though sometimes they arise from a mark on a surface, even in that case it is often appropriate to separate the mark as a distinct region. I have implemented a junction detector similar to the one described in Parida, Geiger and Hummel[70]. Each detected junction is a partition of a small disk in the image into M sectors ($M = 2$ for corners, 3 for T- or Y-junctions, 4 for X-junctions, etc.), each with an assigned gray-scale value. The detector has several parameters that can be used to filter junctions, including the contrast between partitions. We set the threshold for the contrast relatively high, so that only high-contrast junctions are detected. Figure 6.9 shows a sample image (a) and the junctions found in it (b), (d).

The idea of the method is to let these junctions survive the segmentation process; we assume that these junctions indicate region boundaries. Therefore, the set $\mathcal{L}$ of gray-scale values should satisfy the following condition:

For each pair (e, e') of gray-scale values assigned to neighboring sectors at

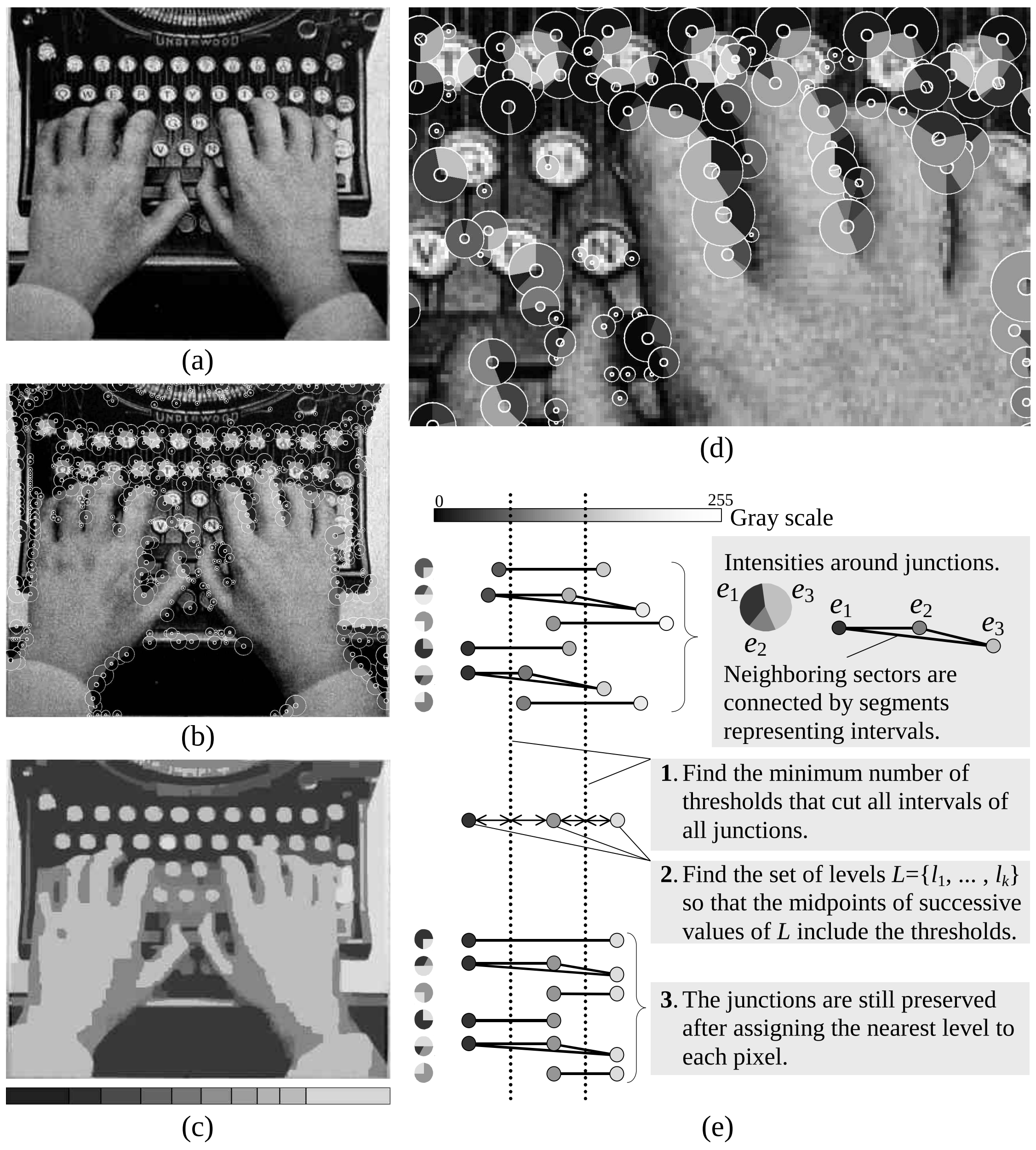

(a)
(b)
(c)
(d)
0
255
Gray scale
Intensities around junctions.
e_1 e_3 e_2
e_1 e_2 e_3
Neighboring sectors are connected by segments representing intervals.
1. Find the minimum number of thresholds that cut all intervals of all junctions.
2. Find the set of levels $L=\{l_1, \dots, l_k\}$ so that the midpoints of successive values of L include the thresholds.
3. The junctions are still preserved after assigning the nearest level to each pixel.
(e)

Figure 6.9: The segmentation process for the sample image "Typing" (a). After junctions are detected (b) (d), gray-scale values of all sectors of the junctions are used to determine those for the levels (e). The map from pixels to levels is computed using the maximum-flow algorithm. The segmentation result and 10 selected gray values are shown in (c).

a junction, the nearest values in $\mathcal{L}$ to e and e' are always different.

We look for the minimum set of threshold values that separates any two neighboring sectors in the detected junctions by gray-scale values. Suppose a junction has four sectors a, b, c, and d with gray-scale values $e_\mathrm{a}, e_\mathrm{b}, e_\mathrm{c}$, and e_d in that order, and that the relations $e_\mathrm{c} < e_\mathrm{d} < e_\mathrm{a} < e_\mathrm{b}$ hold. Thus for each pair of the neighboring sectors, the relations $e_\mathrm{a} < e_\mathrm{b}$, $e_\mathrm{b} > e_\mathrm{c}$, $e_\mathrm{c} < e_\mathrm{d}$, and $e_\mathrm{a} > e_\mathrm{d}$ hold. We call $(e_\mathrm{a}, e_\mathrm{b})$, $(e_\mathrm{c}, e_\mathrm{b})$, $(e_\mathrm{c}, e_\mathrm{d})$, and $(e_\mathrm{d}, e_\mathrm{a})$ the intervals associated with the junction. Note that if we define thresholds so that each interval contains at least one threshold, all four gray-scale values are divided by these thresholds. (See Figure 6.9 (e).)

Let $\mathcal{I}$ be the set of all intervals associated with the detected junctions. We define T to be the smallest among the sets of gray-scale values satisfying the following condition:

For any $\iota \in \mathcal{I}$, there exists $t \in T$ such that $t \in \iota$.

Remember that each interval $\iota = (e, e')$ represents the gray-scale values of a pair of neighboring sectors at a junction, and we want these two values mapped to different levels in the segmentation process, lest the boundary inside the junction would be lost. The condition states that there always is at least one threshold t that comes between the two numbers e and e'. Thus T is the set of thresholds that separates all pairs of gray-scale values that we wish to separate. Given such a set T, we define the set of gray-scale values $\mathcal{L} = \{l_1, l_2, \ldots, l_k\}$ for the levels to be the smallest set whose Voronoi diagram includes T. In other words, every threshold t in T is actually a boundary that decides which element of L is the nearest. Then, for each interval $(e, e') \in \mathcal{I}$, the nearest values in $\mathcal{L}$ to e and e' are always different.

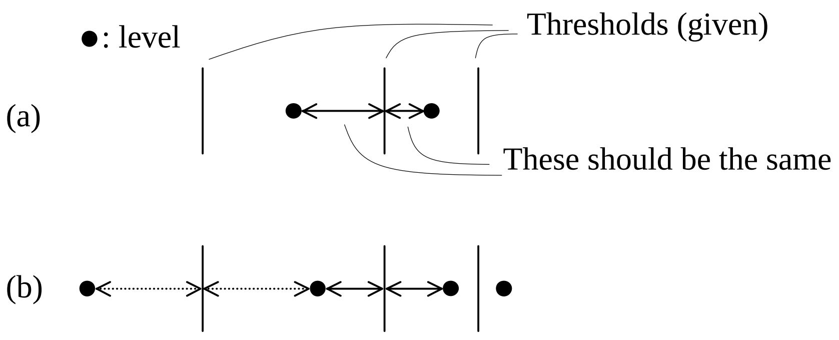


Figure 6.10: Given the set T of thresholds, the set $\mathcal{L}$ of levels is not the set of the midpoints of successive values in T (a). Rather, it is a set such that every threshold t in T is the midpoint of successive values in $\mathcal{L}$ (b).

Note that $\mathcal{L}$ is *not* simply the set of midpoints of successive values in T (Figure 6.10.)

We assume that the gray-scale values are ordered in a natural manner:

$$l_i < l_{i+1}, \quad i = 1, ..., k-1. \tag{6.2}$$

6.3.2 Assignment by the MRF

After deciding the set $\mathcal{L}$ of levels, we use the MRF model to assign a level to each pixel. The graph $G = (V, E)$ is the standard four-connected neighborhood structure; the set V of vertices consists of pixels, and each pixel is connected to its four neighbors. The energy is defined by (4.1), i.e.,

$$E(X) = \sum_{(u,v)\in E} g(X_u - X_v) + \sum_{v\in V} h(v, X_v)$$

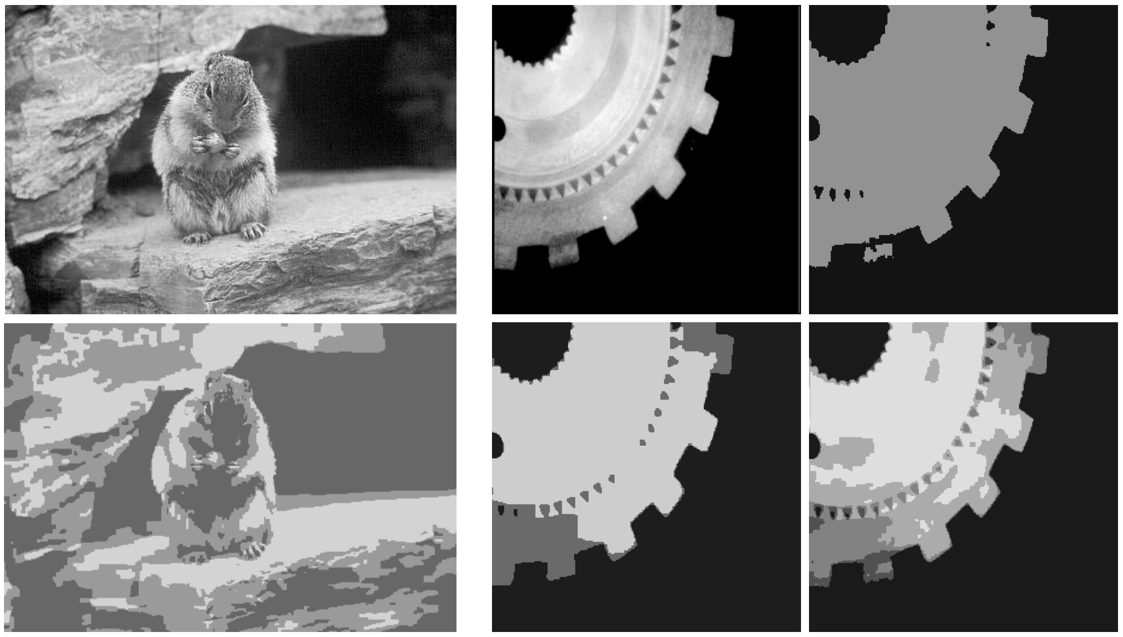


Figure 6.11: The results of MRF segmentation. Left: a segmentation of sample image "Squirrel" into three levels. Right: segmentations of sample image "Gear" (top left) into two (top right), three (bottom left), and four (bottom right) levels.

and

$$g(x) = |x|,$$
$$h(v,x) = (I(v) - l_x)^2.$$

The smoothing cost $g(x)$ is the simplest one under the convexity requirement. This corresponds to the penalty edges that have positive capacities only when they are horizontal, i.e., just as shown in Figure 4.1. Note that the level set $\mathcal{L}$ does not correspond to the set L of labels in the MRF model, as $\mathcal{L}$ is not evenly spaced. What corresponds to L in this case is rather the set $\{1, \ldots, k\}$ of level indices.

In Figure 6.11, two additional instances are shown. The "Squirrel" sample image (400×274 pixel 8-bit grayscale) is segmented to only three levels. The

"Gear" sample image (256 $\times$ 256 pixel 8-bit grayscale) is shown segmented to 2, 3 and 4 levels. The number of levels can be indirectly controlled by changing various parameters for the junction detector.

6.4 Contour and Region Detection

The minimum ratio cycle method that is explained in chapter 5 finds a closed contour, which defines a region in the image plane. The method can also incorporate region information that can be expressed as an integrable function on the image plane. In this section, I describe an application of the method for simple closed contour extraction from a single image. In the subsequent sections, I discuss examples of segmentation processes that use not only intensity information but also correspondence between multiple images.

6.4.1 Gradient-based Energy

Using the region and boundary extraction technique described in Chapter 5, we can naturally separate a figure from ground in a single image. The most simple and intuitive criterion for the boundary local energy would be using the intensity gradient $\vec{\nabla} I$ for the vector field $\vec{v}$ in (5.5):

$$E(I, \partial \mathcal{R}) = \frac{\int_{\partial \mathcal{R}} \vec{t} \times \vec{v}\, ds}{\int_{\partial \mathcal{R}} f\, ds}.$$

This encourages the contour to go through high gradient points perpendicular to the gradient vector field.

As discussed in section 5.2.2, we can freely convert between a boundary-based energy and the region-based one; therefore we can use, in addition to the gradient energy above, any function e that measures the merit of the region $\mathcal{R}$ surrounded by the closed contour as illustrated in the energy (5.4):

$$E(I,\mathcal{R}) = \frac{\int_{\mathcal{R}} e(x,y)\,dx\,dy}{\int_{\partial\mathcal{R}} f\,ds},$$

The function e can be any integrable function. In particular, it can be the convolution of the image with any filter F: $e(p) = I \star F(p)$. In this case, equation (5.5) also takes the form of a convolution:

$$E(I,\mathcal{R}) = \frac{\int_{\mathcal{R}}(I \star F)\,dx\,dy}{\int_{\partial\mathcal{R}} f\,ds}$$
$$E(I,\partial\mathcal{R}) = \frac{\int_{\partial\mathcal{R}}\vec{t} \times (I \star \vec{v})\,ds}{\int_{\partial\mathcal{R}} f\,ds}$$

where $\vec{\nabla} \cdot \vec{v} = F$.

Note that the result of this region and boundary extraction technique can be used in combination with the MRF method. For instance, we can modify the smoothing behavior of the MRF—strengthen the statistical correlation among pixels in the region and weaken it between the pixels inside and outside the region. This amounts to incorporating non-local information (that a pixel is part of a larger structure) in the MRF model.

6.4.2 Minimum Ratio Cycle Algorithm

The particular instance of the minimum ratio cycle algorithm used here is as follows.

Remember (section 5.2.1), on a graph $G = (V,E)$, we are given an integral cost $c : E \to \mathbf{Z}$ and a nonnegative integral length $t : E \to \mathbf{Z}^{>0}$. The problem is to find a cycle C with a minimum ratio weight

$$w(C) = \frac{c(C)}{t(C)} = \frac{\sum_{e \in C} c(e)}{\sum_{e \in C} t(e)}.$$

We define a new weight function $w_\alpha : E \to \mathbf{Q}$, parametrized by $\alpha \in \mathbf{Q}$, by

$$w_\alpha(e) = c(e) - \alpha t(e).$$

The algorithm searches for a rational number α such that i) there is no negative cycle with respect to the weight function w_α, and ii) $\min_C w_\alpha(C) = 0$. Suppose we have such an α and a cycle C such that $w_\alpha(C) = 0$. Then C is a minimum ratio cycle and α is its ratio weight. To see why, suppose this were not the case and that we have another cycle C' that gives a smaller ratio weight:

$$w(C') = \frac{c(C')}{t(C')} < \frac{c(C)}{t(C)} = \alpha,$$

where the second equality follows from $w_\alpha(C) = c(C) - \alpha t(C) = 0$. It follows that $c(C') - \alpha t(C') = w_\alpha(C') < 0$, which means C' is a negative cycle with respect to w_α. This contradicts the manner in which α was chosen.

The algorithm searches the pairs of rational numbers α and cycles C as follows. We start with a known upper bound α_0 of α and maintain the property $\alpha_k \geq \alpha$. If $\alpha_k > \alpha = \frac{c(C)}{t(C)}$, then $w_{\alpha_k}(C) = c(C) - \alpha_k t(C) < 0$. Hence, there exists at least one negative cycle. We use a negative cycle detection algorithm to find a negative cycle C_k with respect to the weight function w_{α_k}. If we do not find a negative cycle, it means $\alpha_k = \alpha$; we find a zero cycle C and we are done. Otherwise, we set $\alpha_{k+1} = w(C_k) = \frac{c(C_k)}{t(C_k)} \geq \alpha$. It follows from $w_{\alpha_k}(C_k) = c(C_k) - \alpha_k t(C_k) < 0$ that $\alpha_k > \alpha_{k+1}$. We repeat this until we find α.

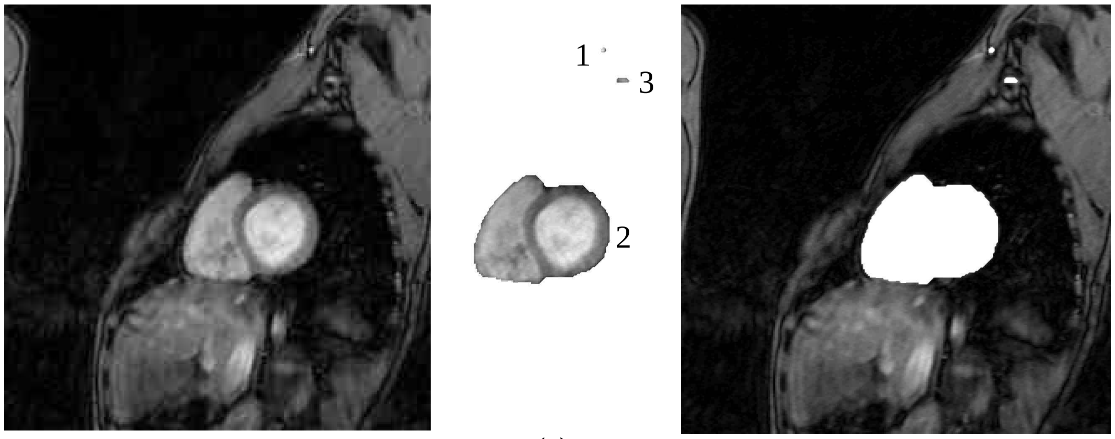
There are other approaches to the search, including binary search [53] and a more sophisticated approach using parametric edge weights [60]. In practice we find that the fastest method is simply linear search explained above. Note that the weights c and t are integral. This means that w is rational, as already stated. This enables us to place a pseudo-polynomial on the search time [53]. In practice, it can take anywhere between a few seconds to ten minutes to run for the size of the problem with which we are dealing here.

6.4.3 Results

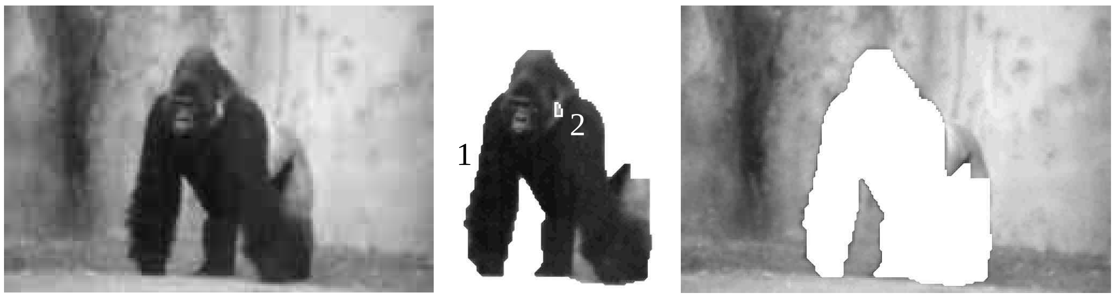
Figure 6.12 shows three results. The energy used is a combination of simple boundary and region energies. The boundary energy is the gradient; the local term $\vec{t} \times \vec{\nabla} I$ is integrated around the closed contour. This encourages the contour to i) go through high-gradient points, and ii) go perpendicular to the gradient direction. In addition to this, a constant function is added as the region energy $e(x,y)$. This is to compensate the remaining tendency to favor small cycles, which was discussed in section 5.1.2. For the integral in the denominator, simple arc length is used. The sizes of the images are (a) 256×256, (b) 200×134, and (c) 124×166 pixels. The algorithm was applied repeatedly so as to find multiple regions. Each time a boundary was found, those vertices in the graph that lay on the boundary were removed. The numbers indicate the order in which the regions were found. Note that both small and large regions were found.

6.5 Stereo and Motion Segmentation

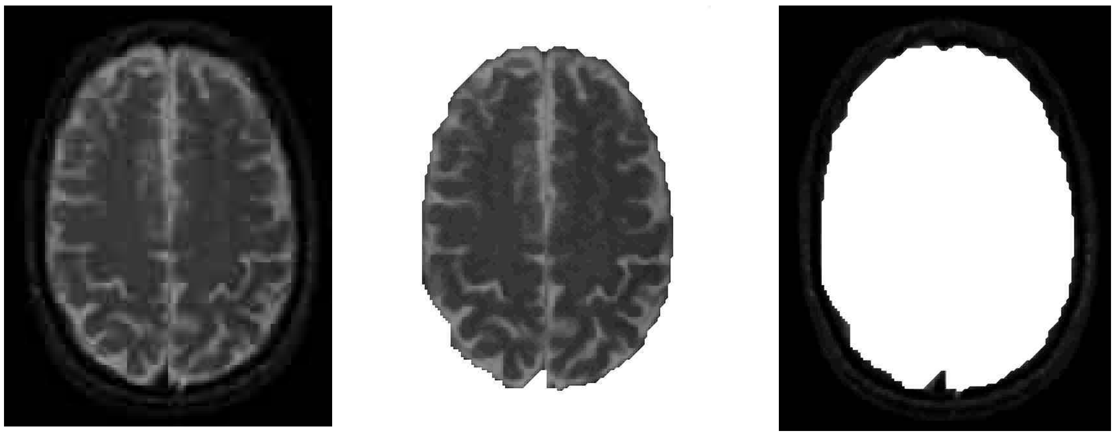
Sometimes multiple images of a scene can be used for segmentation. For instance, a stereo pair gives depth information for a scene once we solve the cor-



(a)



(b)



(c)

respondence problem. The depth map can then be used to help determine the object boundaries based on, for instance, a hypothesis that large depth changes tend to correspond to object boundaries. In an image sequence, or a video, different objects may move differently, thereby revealing the grouping of the points in the images into larger objects, if we can extract such a local motion map. The vector field that represents the motion of each point in the image is called the optical flow. Given an optical flow, we can use the information to determine the object boundaries just as we can use a depth map.

However, there is another possibility—in this section, I describe a method for multiple-image segmentation that uses the correspondence information between the images on the fly, without first extracting such information in the form of a depth map or an optical flow.

6.5.1 Finding Discontinuities in a Match Space

Here I use the notation of section 5.3. Let us suppose that we have n images $I_1, \dots, I_n$. As before, we denote by $\check{I}_i$ the domain on which the image function I_i is defined, i.e., $I_i(x)$ gives the color in the image at point x on the image plane $\check{I}_i$. We define the match space M by the direct product of all images:

$$M = \check{I}_1 \times \cdots \times \check{I}_n.$$

There is a natural projection from the match space M to each image:

$$p_i : M \to \check{I}_i.$$

Each point x in M stands for a match of the points $p_1(x), \ldots, p_n(x)$. From the assumption that each image is a rendition of a scene, it follows that many points in the scene appear in all the images; such a point q in the scene defines a point x in M by requiring $p_i(x)$ to be the position where q appears in the image I_i. In this way, the whole scene surface defines a two dimensional surface in M.

Any function (or differential form) on the image planes can be pulled back to the match space M by the projection; for a function f on $\check{I}_i$, the pull-back $p_i^* f$ of f by p_i is a function on M defined by $(p_i^* f)(x) = f(p_i(x))$ for $x \in M$.

The energy functional we use here is, as defined in (5.8),

$$E(s) = \frac{\int_{S^1} s^* A}{\sum_i \int_{S^1} |f_i|_{s,1}^2 / |g_i|_{s,0}^2}, \tag{6.3}$$

where $s : S^1 \to M$ is a closed contour in the match space M, A is a one-form on M, and f_i and g_i are each either functions or one-forms on M. See section 5.3 for the notation.

Numerator Let us consider the simplest example: we look for a closed contour in each image exactly in the same way as in the previous section, but simultaneously in all the images. Thus, on each image plane $\check{I}_i$, we define a one-form A_i that corresponds to the sum of the gradient vector field $*\mathbf{d}I_i$ (i.e., $\vec{\nabla} I_i$) and an integration of the constant function. We pull all these one-forms back to M and take a linear combination (usually just a sum) to obtain a one-form on M:

$$A = \sum_{i=1}^{n} a_i p_i^* A_i, \quad a_i \in \mathbf{R}. \tag{6.4}$$

For now, let us assume that we use, in the energy (6.3), a denominator that simply measures the length of the contour:

$$\int_{S^1} |1|^2_{s,1}.$$

When we find the minimum, we find a closed contour in M that would tend to be a good contour in each of projections to the images.

As in this example, in general we use the numerator of the energy as the measure of the desirability of each projected contour and the region it encloses, by combining one-forms in the images by (6.4). This alone would tend to match similar regions, especially if the single-image energy uses a region term, such as a filter response, that seeks a particular kind of region.

Denominator We use the denominator of the energy functional (6.3) to evaluate how well the points that are projected to the image planes match to each other. For instance, a simple measure would be the following: we define a vector-valued function on M:

$$f_{\mathrm{N}} = (p_1^* I_1 - p_2^* I_2, p_2^* I_2 - p_3^* I_3, \dots, p_{n-1}^* I_{n-1} - p_n^* I_n),$$

which is suitable for the case where there is a linear order on the images, as in the case of a video sequence; we can also compare all possible pairs of the images, if that is preferable. Then we use the denominator

$$\int_{S^1} |f_{\mathrm{N}}|^2_{s,1}.$$

The integrand in this case is

$$|f_\mathrm{N}|^2_{s,1} = \{\sum_{i=1}^{n-1}(p_i^* I_i - p_{i+1}^* I_{i+1})^2\}\omega_s,$$

which is always nonnegative and becomes smaller the better the points match each other.

Beyond this simplest instance, we note that we wish to find not only a match but also a discontinuity in the match surface at the contour.

For a point (match) x in M, consider the following affine subspace of the match space M:

$$\Delta(x) = \{y \in M | p_1(y-x) = p_2(y-x) = \cdots = p_n(y-x)\}.$$

This is the unique two-dimensional affine subspace of M that goes through x and is parallel to the diagonal (i.e., the surface we find when all the images are the same.) Intuitively, $\Delta(x)$ represents the direction at x in which the match does not change. This corresponds to constant disparity in the case of stereo, and zero optical flow in the case of motion. Now, let us assume that $x \in M$ represents a good match. This means that x is on the locus of local minima of the modulus of the vector-valued function f_N. Imagine the function restricted to the two-dimensional subspace $\Delta(x)$. Near x, the function is small. Near a discontinuity in the match surface, the function should have a large gradient; and the contour should be perpendicular to the gradient of the function. Thus, we take a vector-valued one-form on M:

$$f_\mathrm{D} = *_2 \mathbf{d}_2 f_\mathrm{N},$$

where $\mathbf{d}_2$ and $*_2$ stand for the gradient and the Hodge star on $\Delta(x)$. When the

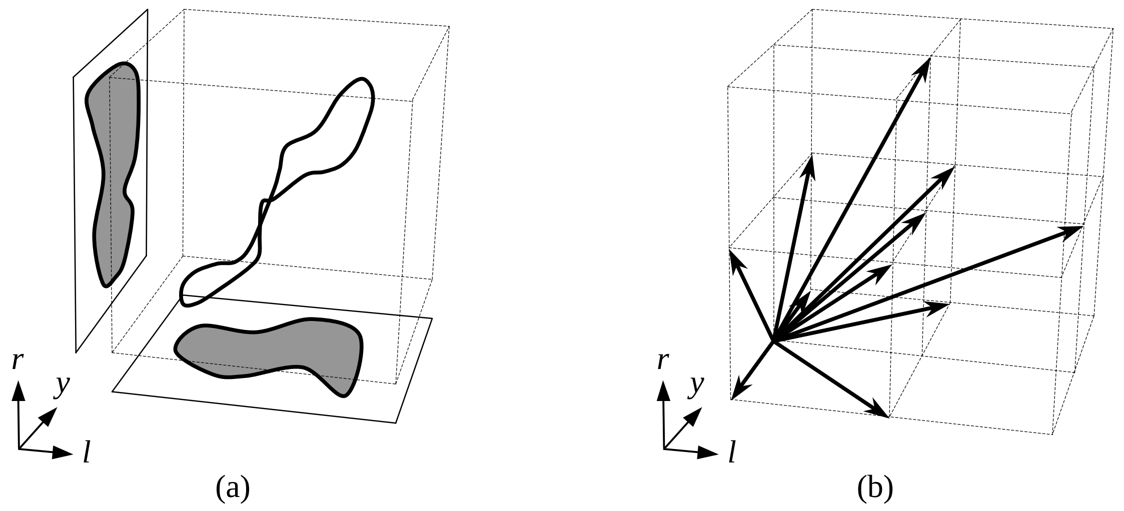


Figure 6.13: (a) A closed contour in the *l*-*r*-*y* space and (b) Neighborhood structure for an embedded graph.

contour runs near a discontinuity perpendicular to the gradient, the function $|f_{\mathrm{D}}|^2_{s,0}$ becomes large (see (5.7) for the definition.) Hence, we use the following integral as the denominator in (5.8):

$$\int_{S^1} \frac{|f_{\mathrm{N}}|^2_{s,1}}{|f_{\mathrm{D}}|^2_{s,0}}.$$

This encourages s to go through a locus of good matches (numerator) and near a discontinuity of the match surface (denominator.)

6.5.2 Stereo Segmentation

In the case of stereo, the epipolar constraint reduces the dimension of the match space to three (see section 3.2.1.) Since this is simply a subspace of the match space M that was discussed in the previous section, the argument there applies without modification. Figure 6.13 (a) shows a closed contour in the *l*-*r*-*y* space.

According to the method described in chapter 5, we embed a graph in this

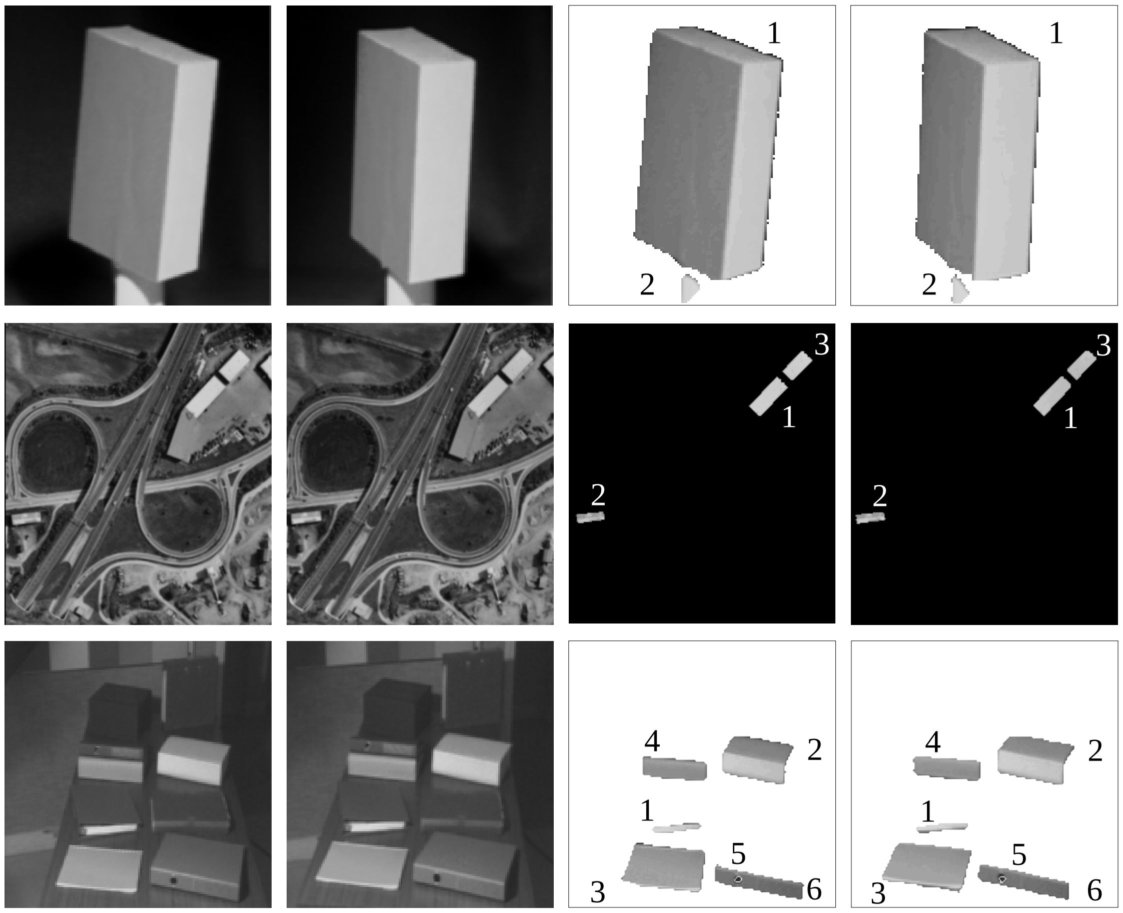


Figure 6.14: Stereo segmentation.

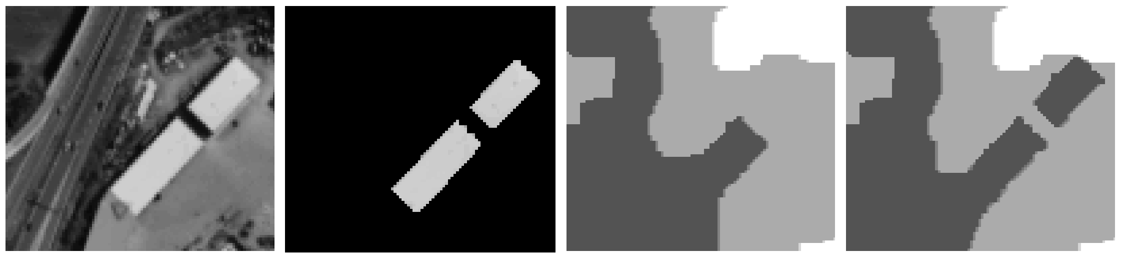


Figure 6.15: Disparity map obtained by the method explained in section 6.2 with and without contour information.

space. Let us assume the graph has a vertex for each integral triple (l, r, y) in a certain range. There are some factors to be considered in deciding which vertices should be connected by an edge. By the monotonicity constraint, every edge must have a nonnegative δl - δr ratio, where we are assuming a coordinate change $(\delta l, \delta r, \delta y)$ for the edge. Also, we would like to avoid an edge that projects to a point in one image, such as $(\delta l, \delta r, \delta y) = (1, 0, 0)$, which projects to $(0, 0)$ in the right image. However, if we take the first neighbors that satisfy these requirements only, no edge would allow for a change in the disparity. Therefore, we include some second neighbor edges such as $(\delta l, \delta r, \delta y) = (1, 2, 0)$. Figure 6.13 (b) shows half of the edges coming out from a vertex. The edges in the other half have the opposite directions to the ones shown.

Figure 6.14 shows three results of stereo segmentation. From the top, the dimensions of the pairs are 128 $\times$ 144, 230 $\times$ 260, and 215 $\times$ 172. The numbers indicate the order in which the contours were found.

Figure 6.15 shows a full disparity map obtained by the method explained in section 6.2 with and without contour information. In the case with contour information (rightmost image,) the smoothing term in the MRF was weakened at points on the contours.

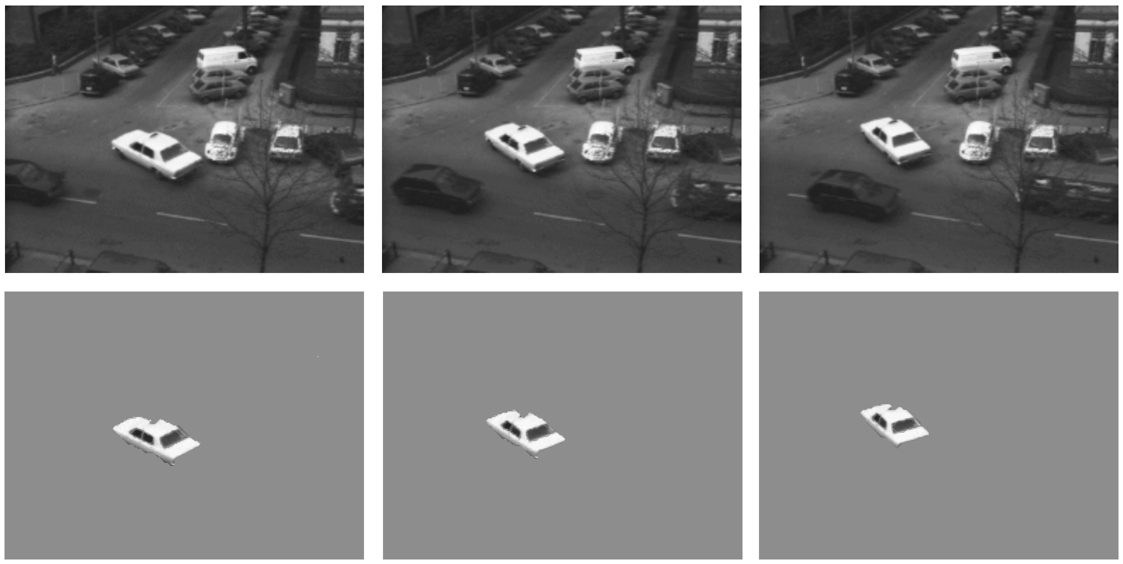


Figure 6.16: Motion segmentation. The bottom row shows the regions that were found in the frames shown in the top row.

6.5.3 Motion Segmentation

Similar considerations apply to motion segmentation as in the case of stereo. For a sequence $I_1, \ldots, I_n$, the match space M is a $2n$ dimensional space. Of course, since the problem size grows exponentially with the number of frames, n cannot be too big. We can restrict the maximum amount of motion between frames to reduce the complexity.

We ran the algorithm on each triple of consecutive frames in a video sequence. Figure 6.16 shows the second frame of each of three triples picked from the beginning, the middle and the end of the sequence.

seven

Summary and Conclusion

In this thesis, I have introduced two new methods that map search problems in computer vision into graph problems that have efficient solutions. Both methods find globally optimal objects that embody information sought in vision problems.

1. Finding codimension 1 objects that minimize an energy functional by embedding a graph with regularly arranged vertices in a Euclidean space and using a minimum cut algorithm. A cut approximates a hypersurface in the space and the total cost of the cut corresponds to the energy. Because of the use of the minimum cut algorithm, a globally optimal solution is found in polynomial time. In particular, the method can globally solve first order Markov Random Field problems in more generality than was previously possible. The convexity of smoothing function in the energy is shown to be both necessary and sufficient for the applicability of the method.

2. Finding an optimal cycle in a Euclidean space, typically a closed contour

in the image plane, by using a minimum ratio cycle algorithm. In the case of two dimensions, the energy can depend not only on the cycle itself but also on the region defined by the cycle. Because of this, the method unifies the two competing views of boundary and region segmentation.

I have shown the results of experiments in the areas of binocular stereo, image restoration, and image segmentation using these methods. Image segmentation in various situations using different forms of information, such as motion, stereo, and intensity, was demonstrated.

Bottom-up vs. Top-down In the introduction, I stated that a search process is essential to the solution of any vision problem; so-called bottom-up processes, or simple data-driven transformations, can only be part of the solution. This is also a matter of where the information resides, in addition to being a question of method (searching or converting.) To perform a search, you have to know what you are looking for.

Take contour extraction. The space of contours embodies the idea of what a contour is. Think of the following two ways of representing a closed contour in a rectangular array of pixels.

1. A closed contour is a subset of the array, where each pixel belonging to the subset has exactly two neighbors that also are in the subset.

2. A closed contour is a sequence of pixels such that each pixel is a neighbor of the previous one in the sequence, and the first and the last pixels in the sequence are neighbors.

We can consider these the implicit and explicit definitions. The implicit definition characterizes a contour using only the concepts of subset and neighbor, whereas in addition to these the explicit definition uses the concept of sequence, essentially the concept of 1D object. Note that the additional concept used in the explicit version characterizes the idea of a contour.

Corresponding to these two definitions, there are two different ways of representing the closed contour:

1. An array of boolean variables corresponding to pixels.

2. A sequence of pointers to pixels.

This difference in representation has very important implications. First, the explicit representation is much more compact than the implicit one. In the contour case, the implicit version needs storage proportional to the number of pixels, while the explicit one only needs storage proportional to the length of the contour. Since they are representing the same objects, the first representation is more wasteful; in the space of possible values, almost no values represent a closed contour, because of the strong condition on what makes a subset of an array a closed contour. Thus it becomes convenient to weaken the restriction and obtain a subset that consists of pixels that are likely to be on a contour—or an edge map. We can imagine a similar, implicit representation of the stereo match surface, represented as the subset of the match space consisting of matches that are likely to be on the match surface. And because an image is represented in a manner similar to the implicit representation, it is easy to make such an edge map from an image or a pair of images. The traditional distinction between bottom-up and top-down processes largely coincides with these two ways of representing the resulting information.

I think that the information we wish to obtain is always explicit and, to obtain explicitly defined information we always need a search. That is the reason for the claim that a search is always necessary for vision. Search techniques in vision, including the two I have introduced in this thesis, search in the implicit representation space for the best object, while preserving the constraints that are needed to extract the object in an explicit form.

Designing an Ontology—and Space of Ontologies Moreover, what we ultimately would like to obtain is even more complex. In segmentation, what exactly is it that we are trying to find—the object boundaries? Remember that what we really want to find is the object boundaries in the scene, not in the image. To segment an image into objects as reliably as a human being can, I think the system must at least have models of shape and illumination in a 3D space, in addition to such basic notions as colors and textures. Illumination is simple enough, but shape is not. What is the space of all possible shapes?

A vision system must have an *ontology*—what I have been calling the parameter space. For the system, what "exists" is that which can be represented. It can only "see" entities in its ontology. As we see in the case of segmentation, defining an appropriate ontology is not a simple matter. Moreover, it is not enough to have a space of entities as a mathematical definition. As explained in the introduction, any probability distribution on the space must be drastically compressed to be useful. This means that the probability of an entity should be given by an algorithm that takes a representation of the entity as input. For this, it is essential that the parameter space be represented in an explicit form. The choice of an ontology is therefore very closely related to its representation and that of the probability distributions.

The design of such an ontology, representation, and probability distribution has been done case by case. What is needed, however, is a space of ontologies and their explicit representations. It would resemble the set of all computer programs and its semantics. An explicit representation corresponds to a computer program; its meaning, or semantics, is in some space of entities, or an ontology. I see some interesting possibilities here—but that is another story.

Bibliography

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms and Applications*. 133–165, Prentice Hall. NJ, U.S.A. 1993.

[2] T. D. Alter and R. Basri. Extracting salient curves from images: an analysis of the saliency network. *International Journal of Computer Vision*, **27**(1):51–69, 1997.

[3] A. Amini, S. Tehrani, and T. Weymouth. Using dynamical programming for minimizing the energy of active contours in the presence of hard constraints. In *Proceedings of the International Converence on Computer Vision*, 95–99, 1988.

[4] B. Anderson. The role of partial occlusion in stereopsis. *Nature*, **367**:365–368, 1994.

[5] N. Ayache. *Artificial Vision for Mobile Robots*. MIT Press. Cambridge, MA, U.S.A. 1991.

[6] H. H. Baker and T. O. Binford. Depth from Edge and Intensity-based Stereo. In *Proceedings of 7th International Joint Conferences on Artificial Intelligence*, 631–636, 1981.

[7] P. N. Belhumeur. A Bayesian Approach to Binocular Stereopsis. *International Journal of Computer Vision*, **19**(3):237–262, 1996.

[8] P. N. Belhumeur and D. Mumford. A bayesian treatment of the stereo correspondence problem using half-occluded regions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Champaign, IL, U.S.A. 506–512, 1992.

[9] J. Besag. Spatial interaction and the statistical analysis of lattice systems (with discussion). *Journal of the Royal Statistical Society, Series B*, **36**:192–326, 1974.

[10] S. Birchfield and C. Tomasi. Multiway Cut for Stereo and Motion with Slanted Surfaces. In *Proceedings of the International Converence on Computer Vision*, 489–495, Kerkyra, Greece. 1999.

[11] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press. Cambridge, MA, U.S.A. 1987.

[12] Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with efficient approximations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, U.S.A. 648–655, 1998.

[13] P. Burt and B. Julesz. A disparity gradient limit for binocular fusion. *Science*, **208**:615–617, 1980.

[14] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8**(6):679–698, 1986.

[15] B. Cernushi-Frias, D. B. Cooper, Y. P. Hung, and P. Belhumeur. Towards a model-based bayesian theory for estimating and recognizing parameterized 3d objects using two or more images taken from different positions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **11**:1028–1052, 1989.

[16] A. Champolle, D. Geiger, and S. Mallat. Un algorithme multi-echelle de mise encorrespondance stereo base sur les champs markoviens. In *13th GRETSI Conference on Signal and Image Processing*, Juan-les-Pins, France. 1991.

[17] B. V. Cherkassky and A. V. Goldberg. On implementing push-relabel method for the maximum flow problem. In *Proceedings of 4th International Programming and Combinatorial Optimization Conference*, 157–171, 1995.

[18] Y. Choquet-Bruhat, C. DeWitt-Morette, and M. Dillard-Bleick. *Analysis, Manifolds and Physics*. Elsevier Science. Amsterdam, The Netherlands. 1996.

[19] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to algorithms*. McGrow-Hill. New York. 1990.

[20] I. J. Cox, S. B. Rao, and Y. Zhong. Ratio regions: a technique for image segmentation. In *Proceedings of International Conference on Pattern Recognition*, **2**:557–564, 1996.

[21] G. B. Dantzig, W. O. Blatner, and M. R. Rao. Finding a Cycle in a Graph with Minimum Cost to Time Ratio with Application to a Ship Routing Problem. In *Theory of Graphs: International Symposium*, 77–84, Gordon and Breach. New York. 1966.

[22] T. Darrell and A. Pentland. Robust estimation of a multilayered motion representation. In *Proceedings of IEEE Workshop on Visual Motion*, Princeton, NJ, U.S.A. 1991.

[23] R. L. Dobrushin. The description of a random field by means of conditional probabilities and conditions of its regularity. *Theory of Probability and Its Applications*, **13**:197–224, 1968.

[24] J. Elder and S. W. Zucker. The Effect of Contour Closure on the Rapid Discrimination of Two-Dimensional Shapes. *Vision Research*, **33**:981–991, 1993.

[25] J. Elder and S. W. Zucker. A Measure of Closure. *Vision Research*, **34**(24): 3361–3369, 1994.

[26] O. Faugeras. *Three-Dimensional Computer Vision*. MIT Press. Cambridge, Mass. 1993.

[27] O. Faugeras and R. Keriven. Complete Dense Stereovision Using Level Set Methods. In *Fifth European Conference on Computer Vision*, Freiburg, Germany. 379–393, Springer-Verlag. 1998.

[28] P. A. Ferrari, A. Frigessi, and P. de Sá. Fast approximate maximum a posteriori restoration of multicolour images. *Journal of the Royal Statistical Society, Series B*, **57**:485–500, 1995.

[29] D. Geiger, A. Gupta, L. Costa, and J. Vlontzos. Dynamic Programming for Detecting, Tracking, and Matching Deformable Contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**(3):294–302, 1993.

[30] D. Geiger, B. Ladendorf, and A. Yuille. Occlusions and binocular stereo. *International Journal of Computer Vision*, **14**:211–226, 1995.

[31] D. Geiger and F. Girosi. Parallel and deterministic algorithms for MRFs: surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **13**:401–412, 1991.

[32] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6**:721–741, 1984.

[33] B. Gillam and E. Borsting. The role of monocular regions in stereoscopic displays. *Perception*, **17**:603–608, 1988.

[34] A. V. Goldberg and R. E. Tarjan. A New Approach to the Maximum-Flow Problem. *Journal of the ACM*, **35**(4):921–940, 1988.

[35] D. M. Greig, B. T. Porteous, and A. H. Seheult. Discussion of: On the statistical analysis of dirty pictures (by J. E. Besag.). *Journal of the Royal Statistical Society, Series B*, **48**:282–284, 1986.

[36] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society, Series B*, **51**:271–279, 1989.

[37] W. E. L. Grimson. *From Images to Surfaces*. MIT Press. Cambridge, MA, U.S.A. 1981.

[38] G. Guy and G. Medioni. Inferring Global Perceptual Contours from Local Features. *International Journal of Computer Vision*, **20** 1996.

[39] R. A. Hummel and S. W. Zucker. On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **5**:267–287, 1983.

[40] H. Ishikawa. Multi-scale Feature Selection in Stereo. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Fort Collins, CO, U.S.A. 132–137, 1999.

[41] H. Ishikawa and D. Geiger. Occlusions, discontinuities, and epipolar lines in stereo. In *Fifth European Conference on Computer Vision*, Freiburg, Germany. 232–248, Springer-Verlag. 1998.

[42] H. Ishikawa and D. Geiger. Segmentation by Grouping Junctions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, U.S.A. 125–131, 1998.

[43] H. Ishikawa and D. Geiger. Mapping Image Restoration to a Graph Problem. In *IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing*, Antalya, Turkey. 1999.

[44] A. K. Jain and R. C. Dubes. *Algorithms for clustering data.*. Prentice Hall. 1988.

[45] I. H. Jermyn and H. Ishikawa. Globally Optimal Regions and Boundaries. In *Proceedings of the Seventh International Conference on Computer Vision*, Kerkyra, Greece. 904–910, 1999.

[46] B. Julesz. *Foundations of Cyclopean Perception*. The University of Chicago Press. Chicago, IL, U.S.A. 1971.

[47] G. Kanizsa. Contours Without Gradients or Cognitive Contours. *Italian Journal of Psychology*, **1**:93–112, 1971.

[48] G. Kanizsa. *Organization in Vision: Essays on Gestalt Perception*. Praeger. New York. 1979.

[49] R. Karp. A Characterization of the Minimum Cycle Mean in a Digraph. *Discrete Mathematics*, **23**:309–311, 1978.

[50] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, **1**(4):321–331, 1988.

[51] R. Kinderman and J. L. Snell. *Markov Random Fields and Their Applications*. American Mathematical Society. Providence, RI, U.S.A. 1980.

[52] I. Kovács and B. Julesz. A Closed Curve is Much More Than an Incomplete One: Effect of Closure in Figure-Ground Segmentation. *Proceedings of the National Academy of Sciences of the United States of America*, **90**:7495–7497, 1993.

[53] E. L. Lawler. Optimal Cycles in Doubly Weighted Linear Graphs. In *Theory of Graphs: International Symposium*, 209–213, Gordon and Breach. New York. 1966.

[54] T. Leung and J. Malik. Contour Continuity in Region Based Image Segmentation. Fifth European Conference on Computer Vision Freiburg, Germany. 1998.

[55] J. Malik. On Binocularly viewed occlusion Junctions. In *Fourth European Conference on Computer Vision*, 167–174, Cambridge, U.K. Springer-Verlag. 1996.

[56] S. Mallat. A Wavelet Tour of Signal Processing. Academic Press. 1998.

[57] S. B. Marapane and M. M. Trivedi. Multi-primitive hierarchical (MPH) stereo analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16**(3):227–240, 1994.

[58] D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, **194**:283–287, 1976.

[59] D. Marr and T. Poggio. A computational theory of human stereo vision. *Proceedings of Royal Society of London, Series B*, **204**:301–328, 1979.

[60] N. Meggido. Combinatorial Optimization with Rational Objective Functions. *Mathematics of Operations Research*, **4**:414-424, 1979.

[61] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, **21**:1087–1091, 1953.

[62] U. Montanari. On the Optimal Detection of Curves in Noisy Pictures. *Communications of the ACM*, **15**(5):335–345, 1971.

[63] D. Mumford. Elastica and Computer Vision. In *Algebraic Geometry and Its Applications*, Chandrajit L. Bajaj ed. 491–506, Springer-Verlag. New York. 1994.

[64] D. Mumford and J. Shah. Boundary detection by minimizing functionals, I. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, U.S.A. 1985.

[65] K. Nakayama and S. Shimojo. Da vinci stereopsis: depth and subjective occluding contours from unpaired image points. *Vision Research*, **30**:1811–1825, 1990.

[66] Y. Ohta and T. Kanade. Stereo by intra- and inter-scanline search using dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **7**:139–154, 1985.

[67] M. Okutomi and T. Kanade. A multiple-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**:353–363, 1993.

[68] S. Osher and J. A. Sethian. Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations. *Journal of Computational Physics*, **79**:12–49, 1988.

[69] P. Parent and S. W. Zucker. Trace Inference, Curvature Consistency, and Curve Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **11**(8) 1989.

[70] L. Parida, D. Geiger and R. Hummel. Junctions: Detection, Classification and Reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**:687–698, 1998.

[71] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby. Disparity gradients and stereo correspondences. *Perception*, 1987.

[72] A. Rosenfeld, R. A. Hummel, and S. W. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man, and Cybernetics*, **6**(6):420–433, 1976.

[73] S. Roy and I. Cox. A maximum-flow formulation of the N-camera stereo correspondence problem. In *Proceedings of the International Conference on Computer Vision*, Bombai, India. 492–499, 1998.

[74] S. Sarkar and K. L. Boyer. Quantitative measures of change based on feature organization: eigenvalues and eigenvectors. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, U.S.A. 1996.

[75] J. A. Sethian. *Level Set Methods*. Cambridge University Press. 1996.

[76] A. Shashua and S. Ullman. Structural Saliency: The Detection of Globally Salient Structures Using a Locally Connected Network. In *Proceedings of the International Converence on Computer Vision*, Florida, U.S.A. 1988.

[77] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 731–737, Puerto Rico. 1997.

[78] Y. Weiss. Smoothness in layers: motion segmentation using nonparametric mixture estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Puerto Rico. 1997.

[79] L. R. Williams and D. W. Jacobs. Stochastic Completion Fields: A Neural Model of Contour Shape and Salience. *Neural Computation*, **9**:849–870, 1997.