A Perceptual Organization Approach for Figure Completion, Binocular and Multiple-View Stereo and Machine Learning using Tensor Voting

by

Philippos Mordohai

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

August 2005

Dedication

For my parents, Nikos and Rena.

For the memory of my grandparents, Eleni, Nikitas and Hanania, who passed on during my years at USC.

Acknowledgements

I would like to thank my parents who early on instilled in me the value of education. They taught me and my brother, that, unlike your possessions and wealth, you can always rely on your education, your knowledge and your skills. Their continuous encouragement and support, their genuine happiness at our successes and their sadness, but not disappointment, at our failures provide the motivation to pursue and achieve my goals. I do not thank my father and mother separately, since they have always been one in my mind.

I am very grateful to my advisor, Professor Gérard Medioni for giving me a chance when no one else would, as well as for his guidance throughout the years. He sets the example for his students and peers with the high quality of his research and his excellent knowledge of the field. I learned a lot from his approach that does not accept shortcuts and halfhearted efforts. We never sacrificed the integrity of our philosophy and methods to achieve a small improvement in performance. Despite his busy schedule, Prof. Medioni cares about and remains involved in all projects. His style of leadership, which is founded upon his enthusiasm for the work, is something that I have caught myself imitating. I would consider myself very successful if I could run a research group at a level as high as he does.

I would also like to thank Prof. Ramakant Nevatia for broadening my perspective on computer vision and research, and for showing me a different approach to computer vision. His indirect contribution to the work presented in this dissertation is significant. I would also like

to thank him along with Prof. Bosco Tjan for serving on my qualifying examination and dissertation committees. I thank Profs. Ulrich Neumann and Antonio Ortega for serving on my qualifying examination committee.

The faculty, staff and students of the Institute for Robotics and Intelligent Systems welcomed me and supported me throughout the years. Prof. Isaac Cohen has given me with continuous help and support ranging from trivial administrative details to high-level scientific issues. Prof. Keith Price has always been prompt to assist with all kinds of requests I brought up. Andres Huertas helped tremendously in my introduction to the group and the concept of working in a group. He set the tone that made the IRIS group a family and separated it from any other that I have seen. Delsa Tan helped me solve all kinds of problems and fight bureaucracy, which is high on the list of things I strongly dislike. I especially want to thank Prof. Alexandre François for his friendship during his transition from student to postdoctoral researcher to research professor. Since he has been my friend in all these capacities without changing his attitude, I debated hard whether he should be included with the students. I learned a lot from his philosophy for research and life, as well as from the example he set on how a graduate student should be. I have also enjoyed working with Dr. Douglass Fidaleo in the last few months. I really appreciate his friendship and generosity.

My research would have been impossible, or taken a lot longer, had there not been the pioneers of tensor voting. I would like to thank Dr. Gideon Guy, Dr. Mi-Suen Lee and Prof. Chi-Keung Tang for inventing and developing the main concepts upon which my work was founded. The work of Dr. Qian Chen on face reconstruction contributed indirectly to my research and gave me the opportunity to interact with all kinds of people. I also want to thank Drs. ZuWhan Kim and Somboon Hongeng, who were the veteran students when I joined the group, for their help and friendship.

At this point, I would like to thank the people whose stay at IRIS overlapped the most with mine. Profs. Mircea Nicolescu and Eun-Young Kang and I have spent countless hours working

in the lab downstairs and endless nights trying to meet deadlines. I think they agree with me that these are bonding experiences. I hold both Mircea's work ethic and Eun-Young's open mind in very high esteem. At the same time, Drs. Tao Zhao, Sung Chun Lee and Jinman Kang were our upstairs peers. The friendship and camaraderie of all these people contributed in making the environment at IRIS pleasant and motivated me to try to maintain the friendly atmosphere. I also enjoyed the times downstairs with Sung Uk Lee, Hongxia Li, Cheng Zhu, Maurya Shah, Dr. Hosub Yoon and Dr. Jonghyun Park.

I have also spent a lot of time with the current PhD students of the group. I have worked more closely with Changki Min, Elena Dotsenko and Matheen Siddiqui, whose company I have really enjoyed. They provided some of the lighter moments during the days of hard work. In addition, Matheen contributed the feature detectors presented in Chapter 4. I have also spent between one and four years with Fengjun Lv, Mun Wai Lee, Xuefeng Song, Tae Eun Choe, Pradeep Natarajan, Chi-Wei Chu, Wei-Kai Liao, Bo Wu, Chang Yuan, Paul Hsiung, Qian Yu, ShengYang Dai and Kwang Su Kim. They have all helped me in various projects and errants. I have recently started working with Adit Sahasrabudhe who will inherit my code and extend some of my work. I wish them all good luck in their journey towards the PhD.

The presence of visitors from all over the world enhanced the IRIS population both scientifically and culturally. Amin Massad, Dr. René Dencker Eriksen and Dr. Wai-Shun Tong also contributed to the tensor voting framework during their stay at USC. I greatly appreciated the friendship and company of Dr. Christophe Vestri, Dr. Haisong Gu, Prof. Soon Ki Jung, Prof. Vittorio Murino, Riccardo Corsi, Alessandro Marianantoni, Dr. Leo Reyes and Federico Pernici. I would especially like to thank Marco Cristani, one of the nicest people I ever met, for his friendship.

I would like to thank the students that I supervised for allowing me to practice my teaching skills on them. I did my best to make this a positive experience for them and me. Ammar Chinoy, who was the first among them, made my life easier with his skills, knowledge, work

ethic and reliability. He made significant contributions to the algorithms and results of Chapter 6. I would also like to thank Jae Guyn Lim for his hard work and the endless hours he devoted to the project. Finally, I would like to thank Lily Cheng, my last student at USC. I am genuinely impressed by her ability to do research and tackle concepts that are arguably beyond her years. The latest results on binocular stereo were made possible by her exhaustive evaluation of methods and settings. It was a pleasure working with all of them and I wish them a bright future.

Presenting the 3-D face reconstruction and recognition demo was an integral part of my career at USC. I enjoyed working with the Integrated Media Systems Center and demonstrating our research to diverse audiences ranging from world renowned professors to little children. I am thankful to Lisette Garcia, Dr. Isaac Maya, Nichole Phillips, Ann Spurgeon, Cheryl Weinberger and Linda Wright.

I am grateful to Profs. Stergios Roumeliotis and Panayiotis Georgiou for their friendship and ongoing support since my first days at USC. The former has been instrumental in providing more exposure for my work, while the latter has been a priceless resource for all kinds of scientific and practical issues. They are among the few people, besides my teachers, whose help in academic matters I may never be able to repay.

Finally, I am most grateful to all my friends in Greece, the United States and Europe. Unfortunately, they are too numerous to mention. This paragraph is an acknowledgement to all of them for their presence in good and bad times. If you are reading this and think you should be included, then you most probably are. If you have doubts, then maybe there is a reason for that as well.

Contents

List of Figures

List of Tables

Abstract

This works extends the tensor voting framework and addresses a wide range of problems from a perceptual organization perspective. The most important contributions are the addition of boundary inference capabilities, a novel re-formulation of the framework applicable to high-dimensional spaces and the development of algorithms for computer vision and machine learning problems. In all cases, the problem is formulated as the organization of the inputs into salient perceptual structures.

For single image analysis, we address the inference of integrated descriptions in terms of edges and keypoints in way that can be useful for higher level processes. We also address a higher level problem: figure completion. We propose a computational framework which implements both modal and amodal completion and provides a fully automatic decision making mechanism for selecting between them. We illustrate the approach on several inputs producing interpretations consistent with those of human observers.

We propose an approach for stereovision that considers both binocular and monocular cues. It allows the integration of matching candidates generated by different operators, combining their strengths. Then, perceptual organization is performed in 3-D under the assumption that

correct matches, unlike wrong ones, form salient coherent surfaces. Disparity hypotheses for unmatched pixels are generated considering both geometric and photometric criteria. We also address dense, multiple-view stereo under the same assumption. Unlike other approaches, ours can process all data simultaneously, can be applied to more general camera configurations and does not require foreground/background segmentation. We were able to reconstruct scenes that provide serious challenges to state-of-the-art methods.

Finally, we present a new implementation of tensor voting that can be generalized to spaces with hundreds of dimensions, since it achieves significant reductions in storage and computational requirements. Its advantages include its applicability to a far wider range of datasets, noise robustness, absence of global computations and capability to process very large numbers of points. We present results in dimensionality and manifold orientation estimation, geodesic distance measurement, nonlinear interpolation and function approximation. This work opens the door for applications such as unsupervised classification, forward and inverse kinematics.

Chapter 1

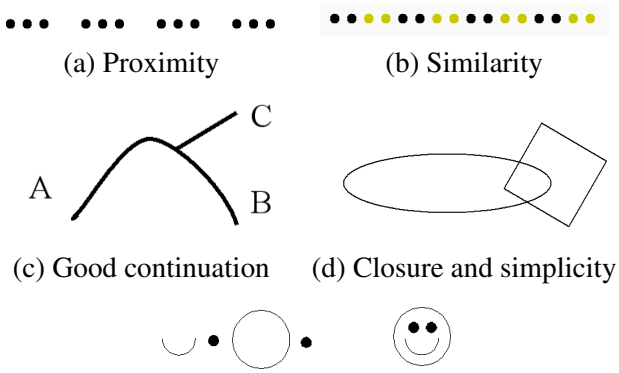
Introduction

This research is an attempt towards the development of a general, unsupervised, data-driven methodology to address problems ranging from computer vision to machine learning. It is founded upon the work of former members of the Computer Vision Laboratory of the Institute for Robotics and Intelligent Systems at the University of Southern California under the guidance of Gérard Medioni. In its preliminary form the tensor voting framework was proposed by Guy in [41] and was further developed by Lee [77] and Tang [151]. It is a computational framework for perceptual organization based on the Gestalt principles, which was mainly applied for organizing generic points (tokens) into coherent groups and for computer vision problems that were formulated as perceptual organization tasks of simple tokens. Our research extends the framework in many ways. First, by augmenting it with first order properties that allow the inference of boundaries and terminations; second, by applying tensor voting directly to images for core computer vision problems, taking into account the inherent difficulties associated with them; and, finally, by proposing a new N-D implementation that opens the door for many applications

in instance-based learning. Throughout this work, we strive to maintain the balance between problem-specific aspects and the general applicability of the methodology being developed.

1.1 Motivation

The tensor voting framework attempts to implement the often conflicting Gestalt principles for perceptual organization. These principles were proposed in the first half of the twentieth century by psychologists in Central Europe. Some of the most representative research can be found in the texts of Köhler [67], Wertheimer [169] and Koffka [66]. At the core of Gestalt psychology is the axiom that "the whole is greater than the sum of the parts", or in other words, that configurations of simple elements give rise to the perception of more complex structures. Figure 1.1 shows a few of the numerous factors discussed in [169].



(a) Proximity (b) Similarity

(c) Good continuation (d) Closure and simplicity

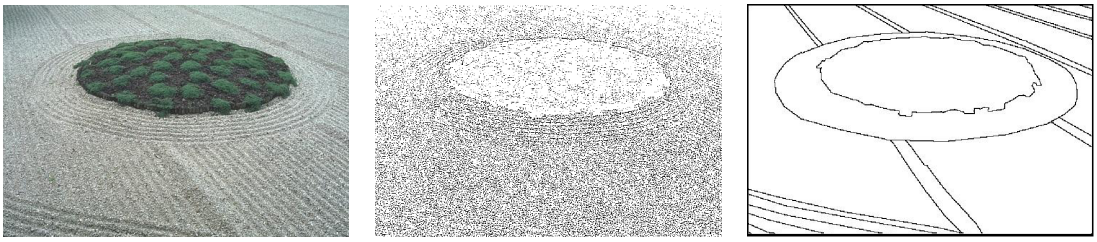
(e) The whole is greater than the sum of the parts

Figure 1.1: Some examples of the Gestalt principles. In (a) the dots are grouped in four groups according to proximity. In (b) the darker dots are grouped in pairs, as do the lighter ones. In (c) the most likely grouping is A to B, and not A to C, due to the smooth continuation of curve tangent from A to B. In (d), the factors of closure and simplicity generate the perception of an ellipse and a diamond. Finally, (e) illustrates that the whole is greater than the sum of the parts.

Even though Gestalt psychologists mainly addressed grouping problems in 2-D, the generalization to 3-D is straightforward, since salient groupings in 3-D can be detected by the human visual system based on the same principles. This is the basis of our approach to binocular and multiple view stereo, where the main premise is that correct pixel correspondences reconstructed in 3-D form *salient* surfaces, while wrong correspondences are not well aligned and do not form any coherent structures. The term *saliency* is widely used in this document to indicate the quality of features to be important, stand out conspicuously, be prominent and attract our attention. Our definition of saliency is that of Shashua and Ullman's [144] *"structural saliency"*, which is a product of proximity and good continuation, and not that of Itti and Baldi [55], where saliency is defined as the property to attract attention due to reasons that include novelty and disagreement with surrounding elements. Note that the term *"alignment"* is used here to indicate good continuation and not any configuration.

In our research, we are interested in inferring salient groups that adhere to the "matter is cohesive" principle of Marr [89] and are more likely to carry information about the configuration of the scene. For instance, given an image, taken from the Berkeley Segmentation Dataset (http://www.cs.berkeley.edu/projects/vision/grouping/) that contains texture, one can perform high-pass filtering and keep high responses of the filter as "edges" (Fig. 1.2). On the other hand, a human observer selects the most salient edges due to their good alignment that forms either familiar or coherent shapes, as in Fig. 1.2(c). One cannot ignore the effect of familiarity in the ability of people to detect meaningful groupings, but a machine-based perceptual organization system should be able to improve upon the performance of the high-pass filter and move towards human performance. Significant edges are not only characterized by high responses of the

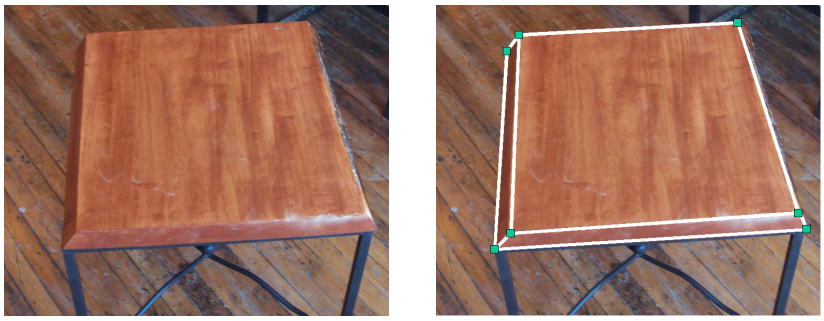
filter, but, more importantly, they are aligned with other edgels to form typically smooth, closed contours that encompass regions that are consistent in color or texture. Edges that are not well aligned, as those in the interior of unstructured texture, are usually less important for the understanding of the scene.



(a) Input image (b) Edge responces (c) Edges marked by human observer

Figure 1.2: An image with texture, outputs of a simple edge detector and human-marked edges

We are also interested in the simultaneous inference of all types of structure that may exist in a scene, as well as their boundaries and intersections. Given two images of a table from different viewpoints such as the one in Fig. 1.3(a), we would like to be able to group pixel correspondences and infer the surfaces, but we would also like to infer the intersections of these surfaces, which are the edges of the table. Furthermore, the intersections of these intersections are the corners of the table, which, despite their infinitesimal size, carry significant information about the configuration of the objects in the scene. The inference of integrated descriptions has been addressed in [77, 151] and is a major advantage of tensor voting over competing approaches. The fact that different feature types are not independent of each other, but rather certain types occur at special configurations of other types, remains an important consideration in this work.

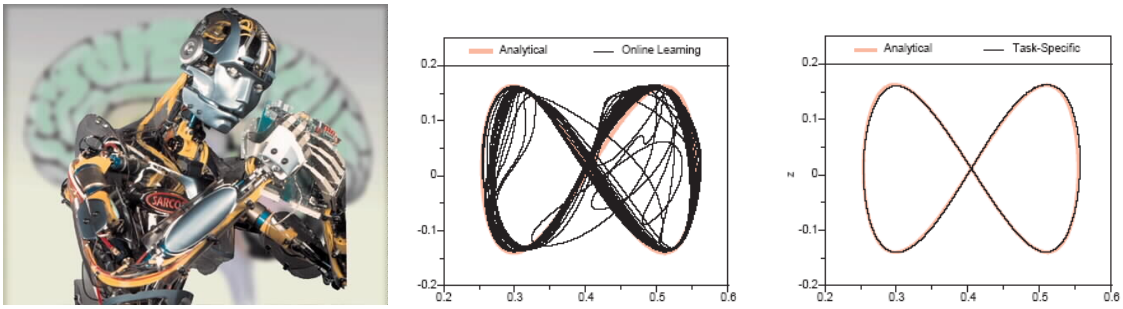


(a) Input image (b) Surface intersections and corners

Figure 1.3: An image (potentially from a stereo pair), surface intersections and corners

The notion of structural saliency extends to spaces of higher dimensionality, even though it is hard to be confirmed by the human visual system. Samples in high-dimensional spaces that are produced by a consistent system, or are somehow meaningfully related, form sooth structures in the same way that point samples from a smooth surface measured with a range finder provide an explicit representation of the surface the surface. The detection of important structures and the extraction of information about the underlying process from them is the object of machine learning. An example from the field of kinematics taken from [163] can be seen in Fig. 1.4 where a humanoid robot tries to lean how to draw a figure "8". The observables in this case are 30 positions, velocities and accelerations of the joints of the arm of the robot. Therefore, each state can be represented as a point in a 90-D space. Even though the analytical computation of the appropriate commands to perform the task is possible, a machine learning approach based on the premise that points on the trajectory form a low-dimensional manifold in the high-dimensional space, proves to be very effective, as seen in Fig. 1.4(c) where the two solutions almost coincide. A significant contribution of this dissertation is a new efficient

implementation of the tensor voting framework that is applicable for salient structure inference in very high-dimensional spaces and can be a powerful tool for instance-based learning.



(a) Humanoid robot (b) Time course of learning (c) Trajectory after learning and analytical solution

Figure 1.4: Training a humanoid robot to draw a figure "8" using an unsupervised machine learning approach [163]

1.2 Approach

Heeding the principles discussed in the previous section, we aim at the development of an approach that is both effective at each problem and also general and flexible. Tensor voting serves as the core of all the algorithms we developed, since it meets the requirements we consider necessary. It is:

- local,
- data-driven,
- unsupervised,
- able to process large amounts of data,

- able to represent all structure types and their interactions simultaneously,
- robust to noise
- and amenable to a least-commitment strategy, postponing the need for hard decisions.

.

The strength of tensor voting is due to two factors: data representation with second order, symmetric, non-negative definite tensors and first order, polarity vectors; and local information propagation in the form of tensor and vector fields. The core of the representation is the second order tensor, which encodes a saliency value for each possible type of structure along with its normal and tangent orientations. The eigenvalues and eigenvectors of the tensor, which can be conveniently expressed in matrix form, provide all the information. For instance, the eigenvalues of a 3-D second order tensor encode the saliency of a token as a surface inlier, a curve inlier or surface intersection, or as a curve intersection or volume inlier. The eigenvectors, on the other hand, correspond to the normal and tangent orientations, depending on the structure type. If the token belongs to a curve, the eigenvectors that correspond to the two largest eigenvalues are normal to the curve, while the eigenvector that corresponds to the minimum eigenvalue is tangent to it. Perceptual organization occurs by combining the information contained in the arrangement of these tokens by tensor voting. During the voting process, each token communicates its preferences for structure type and orientation to its neighbors in the form of votes, which are also tensors that are cast from token to token. Each vote has the orientation the receiver would have if the voter and receiver were part of the same smooth perceptual structure.

More detailed information on tensor voting can be found in the next two chapters, the book by Medioni *et al.* [92] and the more recent chapters in [93, 95].

The major difference between tensor voting and other methodologies is the absence of an explicit objective function. In tensor voting, the solution emerges from the data and, is not enforced upon them. If the tokens align to form a curve, then the accumulation of votes will produce high curve saliency and an estimate for the tangent at each point. On the other hand, if one poses the problem as the inference of the most salient surface from the data under an optimization approach, then a surface that optimizes the selected criteria will be produced, even if it is not the most salient structure in the dataset. Furthermore, tensor voting allows the interaction of different types of structures, such as the intersection of a surface and a curve. To our knowledge this is not possible with any other method.

Additional advantages brought about by the representation and voting schemes are noise robustness and the ability to employ a least commitment approach. As shown in previous research on tensor voting and in the following chapters of this dissertation, tensor voting is robust to very large percentages of outliers in the data. This is due to the fact that random outliers cast inconsistent votes, which do not affect the solution significantly. This does not hold when there is a systematic source of errors, as is the case in many computer vision problems. Such problems are addressed in the appropriate chapters. Regarding the avoidance of premature decisions, the capability of the second order tensor to encompass saliencies for all structure types, allows us to not have to decide whether a token is an inlier or an outlier, for instance, before all the necessary information has been accumulated.

Finally, we believe that a model-free, data-driven design is more appropriate for a general approach since it is easier to generalize to new domains and more flexible in terms of the types of solutions it can infer. The main assumption in the algorithms described here is smoothness, which is a very weak and general model. Moreover, the absence of global computations increases the amount of data that can be processed since computational and storage complexity scale linearly with the number of tokens, if their density does not increase.

1.3 Contributions of this Research

The contribution of our research are threefold:

- enhancements to the original tensor voting framework
- the development of algorithms for computer vision problems and
- the development of algorithms for machine learning problems.

A significant contribution to the framework is the addition of first order representation and voting that complement the strictly second order previous formulation of [41, 61, 77, 92, 110, 151]. The augmented framework of Chapter 3 makes the inference of the terminations of perceptual structures possible. Polarity vectors are now associated with each token and encode the support the token receives for being a termination of a perceptual structure. The new representation exploits the essential property of boundaries to have all their neighbors, at least locally, on the same side of a half-space. The voting scheme is based on that of the second order case, and the first order vector voting fields can be easily derived from the same second order fundamental stick voting field. Besides the capability to detect terminations in itself, the work presented in

this chapter can serve as the foundation for more complex perceptual organization problems. The second major contribution to the framework is a new, efficient N-D implementation, which is described in the last part of the document.

Research on computer vision is described in Chapters 4 through 7. In Chapter 4, the framework, augmented with first order properties, is applied to the problem of integrated feature inference from images. By integrated features we mean step edges, ridges, endpoints and junctions inferred in a way that can be useful for higher level processes. Standard feature detectors do not produce results of sufficient quality and suffer from spurious responses. On the other hand, contour and surface completion approaches require their inputs to come from nearly perfect feature detectors. We convolve the images with a set of oriented filters and extract extensive information by considering, besides the filter responses, properties such as the direction of contrast and a measure of "stepness". Then, we use tensor voting to infer the most salient features based on their consistency with their neighbors. Fragmented curves are connected and junctions are formed during the final completion stage.

In Chapter 5, we move farther along the single-image case, and address the issues associated with figure completion. Endpoints and junctions play a critical role in contour completion by the human visual system, and should be an integral part of a computational process that attempts to emulate human perception. We present an algorithm which implements both modal and amodal completion and integrates a fully automatic decision making mechanism for selecting between them. It proceeds directly from the outputs of the feature extraction module, infers descriptions in terms of overlapping layers and labels junctions as T, L, X and Y. The addition of first order information to the original framework is crucial, since it makes the inference of endpoints and

the labeling of junctions possible. We illustrate the approach on several challenging inputs, producing interpretations consistent with those of the human visual system.

A large part of our research efforts is devoted to the development of a stereo reconstruction approach, which is presented in Chapter 6. Stereovision can be cast as a perceptual organization problem under the premise that solutions must comprise coherent structures that are salient due to the alignment of potential pixel correspondences, reconstructed in a 3-D space. A novel pixel matching framework that allows the integration of multiple diverse matching modules has been developed to generate the initial matching candidates. Tensor voting is performed to infer the correct matches that are generated by the true scene surfaces as inliers of smooth perceptual structures. The retained matches are grouped into smooth surfaces, whose projections on both images can be used to obtain estimates of the color properties of the scene surfaces and reject inconsistent matches. Disparity hypotheses for pixels that remain unmatched are generated based on the color information of nearby surfaces and validated by ensuring the good continuation of the surfaces via tensor voting. Thus, information is propagated from more to less reliable pixels considering both geometric and color information. The use of segmentation based on geometric cues to infer the color distributions of scene surfaces is arguably the most significant contribution of our research on binocular stereo.

In Chapter 7, we present an approach for the inference of dense surfaces from multiple view stereo with general camera placement. Thus far, research on dense multiple view stereo has evolved along three axes: computation of scene approximations in the form of visual hulls; merging of depth maps derived from simple configurations, such as binocular or trinocular;

and multiple view stereo with restricted camera placement. These approaches are either sub-optimal, since they do not maximize the use of available information, or cannot be applied to general camera configurations. Our approach does not involve binocular processing other than the detection of tentative pixel correspondences. The inference of scene surfaces is based again on the premise that correct pixel correspondences, reconstructed in 3-D, form salient, coherent surfaces, while wrong correspondences form less coherent structures. A major advantage over other approaches is that we do not need the "blue sky" assumption, which is equivalent to a priori foreground segmentation.

In the final part of the dissertation, we introduce a new implementation of tensor voting that significantly reduces computational time and storage requirements, especially in high-dimensional spaces, and thus can be applied to machine learning problems, as well as a variety of new domains. This work is based on the observation that the Gestalt principles still apply in spaces of higher dimensionality. The computational and storage requirements of the original implementation, prevent its wide application to problems in high dimensions. The new formulation that does not suffer from these limitations is described in Chapter 8.

Chapter 9 presents our approach to machine learning problems. We address unsupervised manifold learning from observations, in the form of points in high-dimensional spaces, from a perceptual organization standpoint, using the new efficient implementation. We are able to estimate local dimensionality and structure, measure geodesic distances and perform nonlinear interpolation. We first show that we can obtain reliable dimensionality estimates at each point. Then, we present a quantitative evaluation of our results in the estimation of local manifold structure using synthetic datasets with known ground truth. We also present results on datasets

with varying dimensionality and intersections under severe noise corruption, which would have been impossible to process with current state-of-the-art methods. Finally, we address function approximation from samples, which is an important problem with many applications in machine learning. We propose a non-iterative, local, nonparametric approach that can successfully approximate nonlinear functions in high-dimensional spaces in the presence of noise. We present quantitative results on data with varying density, outliers and perturbation, as well as real data.

1.4 Overview

This document is organized as follows:

- Chapter 2 is a brief overview of tensor voting under the light of this research, parts of which were published in [94].
- Chapter 3 presents the first order augmentation to the framework that enables us to infer terminations and label junctions. This work has been published in [93, 95, 161].
- Chapter 4 describes our approach to integrated image feature detection.
- Chapter 5 contains our experiments on figure completion, which were published in [104].
- Chapter 6 formulates the problem of binocular stereo as perceptual organization and presents a thorough experimental validation of all processing stages. The preliminary approach was published in [79], while the complete algorithm that uses monocular information to correct systematic errors that appear in stereo was presented in [105].

- Chapter 7 presents results our approach to dense multiple view stereo with general camera placement [103]. Preliminary versions of this work appear in [101, 102].
- Chapter 8 describes the new implementation of tensor voting for high-dimensional spaces,
- Chapter 9 contains results on core manifold learning problems, such as dimensionality estimation, structure inference, geodesic distance estimation and nonlinear function approximation. Results on dimensionality estimation have been accepted for publication in [106].
- Chapter 10 summarizes our contributions and identifies the directions of our future work.

Chapter 2

The Original Second-Order Formulation of Tensor Voting

In this chapter, we review the original tensor voting framework in 2- and 3-D as it is the foundation of our research. Our contributions in this chapter are limited to a more extensive investigation of the relationships with other perceptual organization approaches, and to the evaluation of the sensitivity to scale during orientation estimation. This chapter is necessary as an introduction in the interest of completeness, as well as for comparison with the novel N-D formulation of tensor voting of Chapter 8.

The tensor voting framework is an approach to perceptual organization that is able to infer salient structures based on the support the tokens, which comprise them, receive from their neighbors. It is based on the Gestalt principles of proximity and good continuation and can tolerate very large numbers of outliers. Data tokens are represented by tensors and the saliency of each token is computed based on information propagated among neighboring tokens via

tensor voting. The tensor voting framework enables us to cast problems in computer vision and other domains, as perceptual organization ones, the solutions for which are the most salient perceptual structures. It has been developed over the past several years beginning in the work of Guy [41] which was followed by the work of Lee [77] and Tang [151]. Applications to vision problems can be found in the dissertations of Nicolescu [110], E.Y. Kang [61] and J. Kang [62]. Parts of the work presented here that address enhancements to the tensor voting framework are a continuation of the work of the first three dissertations and were carried out concurrently with the latter three.

This chapter begins by presenting a broad range of approaches that address similar problems in the field of perceptual organization. The work presented here includes only general methodologies. Approaches to specific problems are presented in the corresponding chapters of this dissertation. This choice should enhance the readability of the document. After related work, we briefly review the representation and voting and analysis in 2-D, followed by the generalization of the framework in 3-D. Some results on synthetic data demonstrate the effectiveness of tensor voting and its low sensitivity to scale, which is the only free parameter.

2.1 Related Work

Perceptual organization has been an active research area since the beginning of the previous century and the work of the Gestalt psychologists [66, 67, 169]. Important issues include noise robustness, initialization requirements, handling of discontinuities, flexibility in the types that can be represented, and computational complexity. This section reviews related work which can be classified in the following categories:

- regularization
- relaxation labeling
- robust methods
- level set methods
- symbolic methods
- clustering
- methods based on local interactions
- methods inspired by psychophysiology and neuroscience.

Regularization Due to the projective nature of imaging, a single image can correspond to different scene configurations. This ambiguity in image formation makes the inverse problem, the inference of the 3-D world from 2-D images, ill-posed. To address ill-posed problems, constraints have to be imposed on the solution space. Within the regularization theory, this is achieved by selecting an appropriate objective function with a regularization term and optimizing it according to the constraints. In general, the goal of the regularization term is to make the solution smoother. Poggio *et al.* [117] present the application of regularization theory to computer vision problems. Terzopoulos [159], and Robert and Deriche [120] address the issue of preserving discontinuities while enforcing global smoothness in a regularization framework. A Bayesian formulation of the problem based on Minimum Description Length is proposed by Leclerc [75]. Variational techniques are used by Horn and Schunk [51] for the estimation of

optical flow, and by Morel and Solimini [107] for image segmentation. In both cases, the goal is to infer functions that optimize the selected criteria, while preserving discontinuities.

Relaxation labeling A different approach to vision problems is relaxation labeling. The problems are cast as the assignment of labels to the elements of the scene from a set of possible labels. Haralick and Shapiro define the consistent labeling problem in [44] and [45]. Labels that violate consistency according to predefined criteria are iteratively removed from the tokens until convergence. Faugeras and Berthod [27] describe a gradient optimization approach to relaxation labeling. A global criterion is defined that combines the concepts of ambiguity and consistency of the labeling process. Geman and Geman discuss how stochastic relaxation can be applied to the task of image restoration in [34]. MAP estimates are obtained by a Gibbs sampler and simulated annealing. Hummel and Zucker [52] develop an underlying theory for the continuous relaxation process. One result is the definition of an explicit function to maximize to guide the relaxation process, leading to a new relaxation operator. The second result is that finding a consistent labeling is equivalent to solving a variational inequality. This work was continued by Parent and Zucker [115] for the inference of trace points of curves in 2-D, and by Sander and Zucker [126] for surface inference in 3-D.

Robust methods In the presence of noise, robust techniques such as RANSAC (random sample consensus) [29] can be applied. Small random samples are selected from the noisy data and are used to derive model hypotheses which are tested using the remainder of the dataset. Hypotheses that are consistent with a large number of the data points are considered valid. Variants of RANSAC include RESC [176] and MIR [70], which are mainly used for segmentation of

surfaces from noisy 3-D point clouds. The extracted surfaces are limited to planar or quadratic, since the number of model parameters cannot be very large, except for the approach of Lee *et al.* [76] which can extract high-order polynomial surfaces. Chen *et al.* [17] introduce robust statistical methods to computer vision. They show how robust M-estimators with auxiliary scale are more general than the other robust estimators previously used in the field. They also point out the difficulties caused by multiple structures in the data and propose a way to detect them. In all cases, an a priori parametric representation of the unknown structure is necessary, thus limiting the applicability of these methods. In addition, the search for a particular type of structure typically results in the discovery of a solution that maximizes the criterion but is not necessarily the most salient property of the data. In other words, the search for the "best" plane will produce a plane even in a dataset that consists entirely of curves.

Level set methods In this case, the desired structures (typically curves in 2-D and surfaces in 3-D) are expressed implicitly as the level sets of a function. In [143], Sethian proposes a level set approach under which surfaces can be inferred as the zero-level iso-surface of a multivariate implicit function. The technique allows for topological changes, thus it can reconstruct surfaces of any genus as well as non-manifolds. Zhao, *et al.* [181] and Osher and Fedkiw [114] propose efficient ways of handling implicit surfaces as level sets of a function. A combination of points, elementary surfaces and curves can be provided as input to their technique, which can handle local changes locally, as well as global deformations and topological changes. The output however is limited to surfaces only. Lorigo *et al.* [85] extend the applicability of level sets in computer vision to manifolds of codimension two. Previously, level sets were limited

to codimension-one manifolds, i.e. surfaces in 3-D, while this generalization allows the inference of curves, which are lower dimensional manifolds, in 3-D. All the implicit surface-based approaches are iterative and require careful selection of the implicit function and initialization. Only one manifold type can be extracted at a time and, moreover, it has to be closed. Our criticism, as with the previous group of methods, is that the desired type may not be present, or may not be the most salient structure in the data.

Symbolic methods Following the paradigm set by Marr [89], many researchers developed methods for hierarchical grouping of symbolic data. Lowe [86] developed a system for 3-D object recognition based on perceptual organization of image edgels. Groupings are selected among the numerous possibilities according to the Gestalt principles, viewpoint invariance and low likelihood of being accidental formations. Later, Mohan and Nevatia [100] and Dolan and Riseman [23] also proposed perceptual organization approaches based on the Gestalt principles. Both are symbolic and operate in a hierarchical bottom-up fashion starting from edgels and increasing the level of abstraction at each iteration. The latter approach aims at inferring curvilinear structures, while the former aims at segmentation and extraction of 3-D scene descriptions from collations of features that have high likelihood of being projections of scene objects. Along the same lines is Jacobs' [56] technique for inferring salient convex groups among clutter since they most likely correspond to world objects. The criteria to determine the non-accidentalness of the potential structures are convexity, proximity and the contrast of the edgels.

Clustering A significant current trend in perceptual organization is clustering [57]. Data are represented as nodes of a graph and the edges between them encode the likelihood that two nodes belong in the same partition of the graph. Clustering is achieved by cutting some of these edges in a way that optimizes global criteria. A landmark approach in the field was the introduction of *normalized cuts* by Shi and Malik [145]. They aim at maximizing the degree of dissimilarity between the partitions normalized by essentially the size of each partition, in order to remove the bias for small clusters. Boykov, Veksler and Zabih [12] use graph-cut based algorithms to approximately optimize energy functions whose explicit optimization is NP-hard. They demonstrate the validity of their approach on a number of computer vision problems. Stochastic clustering algorithms are developed by Cho and Meer [18] and Gdalyahu, Weinshall and Werman [32]. A consensus of numerous clusterings of the data, achieved with randomly selected initializations, is used as a basis of the solution. Finally, Robles-Kelly and Hancock [121] present a perceptual grouping algorithm based on graph cuts and an iterative Expectation Maximization scheme, which improves the quality of results at the expense of increased computational complexity.

Methods based on local interactions We now turn our attention to perceptual organization techniques that are based on local interaction between primitives. Shashua and Ullman [144] first addressed the issue of structural saliency and how prominent curves are formed from tokens that are not salient in isolation. They define a locally connected network that assigns a saliency value to every image location according to the length and smoothness of curvature of curves going through that location. In [115], Parent and Zucker infer trace points and their

curvature based on spatial integration of local information. An important aspect of this method is its robustness to noise. This work was extended to surface inference in three dimensions by Sander and Zucker [126]. Sarkar and Boyer [130] employ a voting scheme to detect a hierarchy of tokens. Voting in parameter space has to be performed separately for each type of structure, thus making the computational complexity prohibitive for generalization to 3-D. The inability of previous techniques to simultaneously handle surfaces, curves and junctions was addressed in the precursor of our research in the work of Guy and Medioni [42, 43]. A unified framework where all types of perceptual structures can be represented is proposed along with a preliminary version of the voting scheme presented here. The major advantages of [42, 43] are noise robustness and computational efficiency, since it is not iterative. How this methodology evolved is presented in the remaining sections of this chapter.

Methods inspired by psychophysiology and neuroscience Finally, there is an important class of perceptual organization methods that are inspired by human perception and research in psychophysiology and neuroscience. Grossberg and Mingolla [39] and Grossberg and Todorovic [40] developed the *Boundary Contour System* and the *Feature Contour System* that can group fragmented and even illusory edges to form closed boundaries and regions by feature cooperation in a neural network. Heitger and von der Heydt [47], in a classic paper on neural contour processing, claim that elementary curves are grouped into contours via convolution with a set of orientation selective kernels, whose responses decay with distance and difference in orientation. Williams and Jacobs [170] introduce the *stochastic completion fields* for contour grouping. Their theory is probabilistic and models the contour from a source to a sink as the

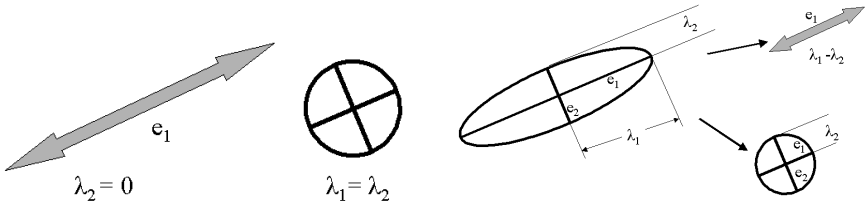
motion of a particle performing a random walk. Particles decay after every step, thus minimizing the likelihood of completions that are not supported by the data or between distant points. Li [82] presents a contour integration model based on excitatory and inhibitory cells and a top-down feedback loop. What is more relevant to our research, that focuses on the pre-attentive, bottom-up process of perceptual grouping, is that connection strength decreases with distance, and that zero or low curvature alternatives are preferred to high curvature ones. The model for contour extraction of Yen and Finkel [174] is based on psychophysical and physiological evidence that has many similarities to ours. It employs a voting mechanism where votes, whose strength decays as a Gaussian function of distance, are cast along the tangent of the osculating circle. An excellent review of perceptual grouping techniques based on cooperation and inhibition fields can be found in [109]. Even though we do not attempt to present a biologically plausible system, the similarities between our framework and the ones presented in this paragraph are nevertheless encouraging.

Comparison with our approach Our methodology offers numerous advantages over previous work. These include the fact that the inputs can be oriented, unoriented or a combination of both, while many of the above methods require oriented inputs to proceed. Our model-free approach allows us to handle arbitrary perceptual structures that adhere to Marr's "matter is cohesive" principle [89] only, and do not require predefined models that restrict the admissible solutions. Our representation is symbolic (in the sense defined in [133]) which brings about advantages that include the ability to attach attributes to each token, and a greater flexibility in assigning meaningful interpretations to tokens. An important feature of our approach is that we

are able to infer all possible types of perceptual structures, such as volumes, surfaces, curves and junctions in 3-D simultaneously, without having to specify the type of structure we are interested in. Instead, analysis of the results of voting indicates the most likely type of structure at each position along with its normal and tangent orientations without having to specify in advance the desired type. To our knowledge, the tensor voting framework is the only methodology capable of this. Our voting function has many similarities with other voting-based methods, such as the decay with distance and curvature [47, 82, 174], and the use of constant curvature paths [115, 130, 134, 174] that result in an eight-shaped voting field (in 2-D) [47, 174]. The major difference is that in our case, the votes cast are tensors and not scalars, therefore they are a lot richer in information. Each tensor simultaneously encodes *all* structure types allowing for a least commitment strategy until all information for a decision has been accumulated. Furthermore, our results degrade much more gracefully in the presence of noise (see for example [42] and [92]).

2.2 Tensor Voting in 2-D

This section introduces the tensor voting framework in 2-D. It begins by describing the original second order representation and voting of Medioni *et al.* [92]. It has been augmented with first order properties as part of this research, which is presented in detail in Chapter 3. To avoid confusion we will refer to the representation and voting of this chapter as "second order", even though their first order counterparts have not been introduced yet. In the following sections we demonstrate how oriented and unoriented inputs can be encoded, and how they propagate their information to their neighbors in the form of votes. The orientation and magnitude of



(a) Geometric illustration of saliency (b) 2-D tensor decomposition

Figure 2.1: Illustration of 2-D second order symmetric tensors and decomposition of a tensor into its *stick* and *ball* components

a second order vote cast from a unit oriented voter are chosen as in [42]. Based on them, the orientation and magnitude of the vote cast by an unoriented token can be derived. The appropriate information for all possible votes is contained in the *stick* and *ball* voting fields. Finally, the present the way perceptual structures are inferred after analysis of the accumulated votes.

2.2.1 Second Order Representation and Voting in 2-D

The second order representation is in the form of a second order, symmetric, non-negative definite tensor which essentially indicates the saliency of each type of perceptual structure (curve, junction or region in 2-D) the token may belong to and its preferred normal and tangent orientations. Tokens cast second order votes to their neighbors according to the tensors they are associated with.

Second order representation A second order, symmetric, non-negative definite tensor is equivalent to a 2×2 matrix, or an ellipse, whose axes are the eigenvectors of the tensor, and

whose aspect ratio is the ratio of the eigenvalues. The major axis is the preferred **normal** orientation of a potential curve going through the location. The *shape* of the ellipse indicates the certainty of the preferred orientation. That is, an elongated ellipse represents a token with high certainty of orientation. Even further, a degenerate ellipse with only one non-zero eigenvalue represents a perfectly oriented point (a curvel). On the other hand, an ellipse with two equal eigenvalues represents a token with no preference for any orientation (Fig. 2.1(a)). The tensor's *size* encodes the saliency of the information encoded. Larger tensors convey more salient information than smaller ones. An arbitrary second order, symmetric, non-negative definite tensor can be decomposed as in the following equation:

$$T = \lambda_1 \hat{e}_1 \hat{e}_1^T + \lambda_2 \hat{e}_2 \hat{e}_2^T = (\lambda_1 - \lambda_2)\hat{e}_1 \hat{e}_1^T + \lambda_2(\hat{e}_1 \hat{e}_1^T + \hat{e}_2 \hat{e}_2^T) \tag{2.1}$$

where λ_i are the eigenvalues in decreasing order and $\hat{e}_i$ are the corresponding eigenvectors (see also Fig. 2.1(b)). Note that the eigenvalues are non-negative since the tensor is non-negative definite. The first term in Eq. 2.1 corresponds to a degenerate elongated ellipsoid, termed hereafter the *stick tensor*, that indicates an elementary curve token with $\hat{e}_1$ as its curve *normal*. The second term corresponds to a circular disk, termed the *ball tensor*, that corresponds to a perceptual structure which has no preference of orientation or to a location where multiple orientations coexist. The size of the tensor indicates the certainty of the information represented by it. For instance, the size of the stick component $(\lambda_1 - \lambda_2)$ indicates curve saliency.

Based on the above, an elementary curve with normal $\vec{n}$ is represented by a stick tensor parallel to the normal, while an unoriented token is represented by a ball tensor. Note that

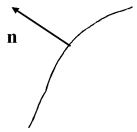
Input	Second order tensor	Eigenvalues	Quadratic form
 oriented		$\lambda_1 = 1, \lambda_2 = 0$	$\begin{bmatrix} n_1^2 & n_1 n_2 \\ n_1 n_2 & n_2^2 \end{bmatrix}$
 unoriented		$\lambda_1 = \lambda_2 = 1$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Table 2.1: Encoding oriented and unoriented 2-D inputs as 2-D second order symmetric tensors

curves are represented by their normals and not their tangents, for reasons that become apparent in higher dimensions. See Table 2.1 for how oriented and unoriented inputs are encoded and the equivalent ellipsoids and quadratic forms.

Second order voting After the inputs, oriented or unoriented, have been encoded with tensors, we examine how the information they contain is propagated to their neighbors. Given a token at O with normal $\vec{N}$ and a token at P that belong to the same smooth perceptual structure, the vote the token at O (the *voter*) casts at P (the *receiver*) has the orientation the receiver would have, if both the voter and receiver belonged to the same perceptual structure. The magnitude of the vote is a function of the confidence we have that the voter and receiver indeed belong to the same perceptual structure. We first examine the case of a voter associated with a *stick tensor* and show how all other cases can be derived from it. We claim that, in the absence of other information, the arc of the *osculating circle* (the circle that shares the same normal as a curve at the given point) at O that goes through P is the most likely smooth path, since it maintains

constant curvature. The center of the circle is denoted by C in Fig. 2.2. In case of straight continuation from O to P, the osculating circle degenerates to a straight line. Similar use of primitive circular arcs can also be found in [115, 130, 134, 174].

As shown in Fig. 2.2, the second order vote is also a stick tensor and has a normal lying along the radius of the osculating circle at P. What remains to be defined is the magnitude of the vote. According to the Gestalt principles it should be a function of proximity and smooth continuation. The influence from a token to another should attenuate with distance, to minimize interference from unrelated tokens, and curvature, to favor straight continuation over curved alternatives when both exist. Moreover, no votes are cast if the receiver is at an angle larger than 45^o with respect to the tangent of the osculating circle at the voter. Similar restrictions on the fields appear also in [47, 82, 174]. The *saliency decay function* has the following form:

$$DF(s, \kappa, \sigma) = e^{-(\frac{s^2 + c\kappa^2}{\sigma^2})} \tag{2.2}$$

where s is the arc length OP, κ is the curvature, c controls the degree of decay with curvature, and σ is the *scale of voting*, which determines the effective neighborhood size. The parameter c

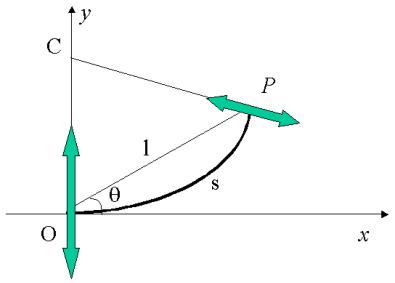


Figure 2.2: Second order vote cast by a stick tensor located at the origin

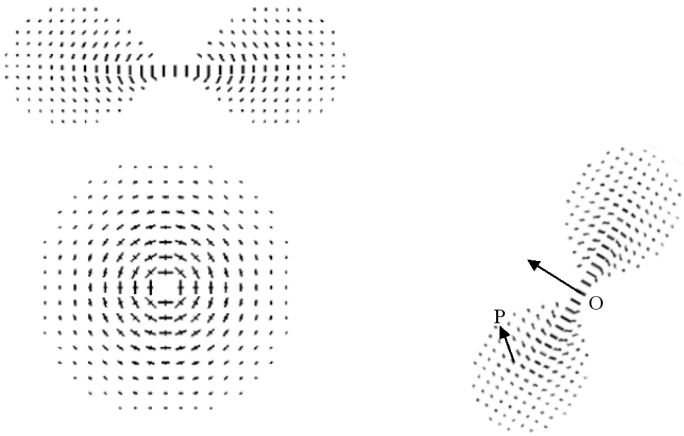
is a function of the scale and is optimized to make the extension of two orthogonal line segments to from a right angle equally likely to the completion of the contour with a rounded corner [42]. Its value is given by: $c = \frac{-16log(0.1)\times(\sigma-1)}{\pi^2}$. Scale essentially controls the range within which tokens can influence other tokens. It can also be viewed as a measure of smoothness. A large scale favors long range interactions and enforces a higher degree of smoothness, aiding noise removal, while a small scale makes the voting process more local, thus preserving details. Note that σ is the only free parameter in the system.

The 2-D second order stick voting field for a unit stick voter located at the origin and aligned with the y-axis can be defined as follows as a function of the distance l between the voter and receiver and the angle θ, which is the angle between the tangent of the osculating circle at the voter and the line going through the voter and receiver (see Fig. 2.2).

$$\mathbf{S}_{SO}(l,\theta,\sigma) = DF(s,\kappa,\sigma) \begin{bmatrix} -sin(2\theta) \\ cos(2\theta) \end{bmatrix} [-sin(2\theta) \quad cos(2\theta)]$$

$$s = \frac{\theta l}{sin(\theta)}, \quad \kappa = \frac{2sin(\theta)}{l} \tag{2.3}$$

The votes are also stick tensors. For stick tensors of arbitrary size the magnitude of the vote is given by 2.2 multiplied by the size of the stick $\lambda_1 - \lambda_2$. In Section 2.2.2 the second order ball voting field is derived from the second order stick voting field. It is used for voting tensors that have non zero ball components and contains a tensor at every position that expresses the orientation preference and saliency of the receiver of a vote cast by a ball tensor. For arbitrary ball tensors, the magnitude of the votes has to be multiplied by λ_2.



(a) The 2-D stick and ball fields (b) Stick vote cast from O to P

Figure 2.3: Voting fields in 2-D and alignment of the stick field with the data for vote generation

The voting process is identical whether the receiver contains a token or not, but we use the term **sparse voting** to describe a pass of voting where votes are cast to locations that contain tokens only, and the term **dense voting** for a pass of voting from the tokens to all locations within the neighborhood regardless of the presence of tokens. Receivers accumulate the votes cast to them by tensor addition.

2.2.2 Voting Fields

In this paragraph we show how votes from ball tensors can be derived with the same saliency decay function and how voting fields can be computed to reduce the computational cost of calculating each vote according to Eq. 2.2. Finally, we show how the votes cast by an arbitrary tensor can be computed given the voting fields.

The second order stick voting field contains at every position a tensor that is the vote cast there by a unit stick tensor located at the origin and aligned with the y axis. The shape of the

field in 2-D can be seen in the upper part of Fig. 2.3(a). Depicted at every position is the eigenvector corresponding to the largest eigenvalue of the second order tensor contained there. Its size is proportional to the magnitude of the vote. To compute a vote cast by an arbitrary stick tensor, we need to align the field with the orientation of the voter, and multiply the saliency of the vote that coincides with the receiver by the saliency of the arbitrary stick tensor, as in Fig. 2.3(b).

The *ball voting field* can be seen in the lower part of Fig. 2.3(a). The ball tensor has no preference of orientation, but still can cast meaningful information to other locations. The presence of two proximate unoriented tokens, the voter and the receiver, indicates a potential curve going through the two tokens. The ball voting field allows us to infer preferred orientations from unoriented tokens, thus minimizing initialization requirements. For simplicity we introduce the notation $\mathbf{B}_{so}(P)$ for the tensor which is the vote cast by a unit ball tensor at the origin to the receiver P. The derivation of the ball voting field $\mathbf{B}_{so}(P)$ from the stick voting field can be visualized as follows: the vote at P from a unit ball tensor at the origin O is the integration of the votes of stick tensors that span the space of all possible orientations. In 2-D, this is equivalent to a rotating stick tensor that spans the unit circle at O. The 2-D ball field can be derived from the stick field $\mathbf{S}_{so}(P)$, according to:

$$\mathbf{B}_{so}(P) = \int_0^{2\pi} R_\theta^{-1} \mathbf{S}_{so}(R_\theta P) R_\theta^{-T} d\theta \tag{2.4}$$

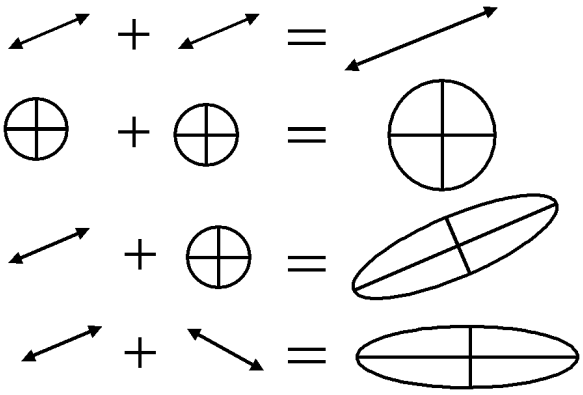


Figure 2.4: Tensor addition in 2-D. Purely stick tensors result only from the addition of parallel stick tensors and purely ball tensors from the addition of ball tensors or orthogonal stick tensors.

where R_θ is the rotation matrix to align $\mathbf{S}$ with $\hat{e}_1$, the eigenvector corresponding to the maximum eigenvalue (the stick component), of the rotating tensor at P. In practice, the integration is approximated by tensor addition (see also Fig. 2.4):

$$V = \sum_{i=1}^{K} \vec{v}_i \vec{v}_i^T \tag{2.5}$$

where V is the accumulated vote and $\vec{v}_i$ are the stick votes, in vector form (which is equivalent since a stick tensor has only one non-zero eigenvalue and can be expressed as the outer product of its only significant eigenvector), from O to P cast by K stick tensors at angle intervals of $2\pi/K$ that span the unit circle. Normalization has to be performed in order to make the energy emitted by a unit ball equal to that of a unit stick. The sum of the maximum eigenvalues of each vote is used as the measure of energy. As a result of the integration, the second order ball field does not contain purely stick or purely ball tensors, but arbitrary second order symmetric tensors. The field is radially symmetric, as expected, since the voter has no preferred orientation.

The voting fields are functions of the position of the receiver relative to the voter, the tensor at the voter and a single parameter, the scale of the saliency decay function. After they have been pre-computed at the desired resolution, computing the votes cast by any second order tensor is reduced to a few look-up operations and linear interpolation. Voting takes place in a finite neighborhood within which the magnitude of the votes cast remains significant. For example, we can find the maximum distance s_{max} from the voter at which the vote cast will have 1% of the voter's saliency, as follows:

$$e^{-(\frac{s_{max}^2}{\sigma^2})} = 0.01 \tag{2.6}$$

The size of this neighborhood is obviously a function of the scale σ. As described in section 2.2.1, any tensor can be decomposed into the basis components (stick and ball in 2-D) according to its eigensystem. Then, the corresponding fields can be aligned with each component. Votes are retrieved by simple look-up operations, and their magnitude is multiplied by the corresponding saliency. The votes cast by the stick component are multiplied by $\lambda_1 - \lambda_2$ and those of the ball component by λ_2.

2.2.3 Vote analysis

Votes are cast from token to token and accumulated by tensor addition. Analysis of the second order votes can be performed once the eigensystem of the accumulated second order 2×2 tensor has been computed. Then the tensor can be decomposed into the stick and ball components:

$$T = (\lambda_1 - \lambda_2)\hat{e}_1\hat{e}_1^T + \lambda_2(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T) \tag{2.7}$$

where $\hat{e}_1\hat{e}_1^T$ is a *stick tensor*, and $\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T$ is a *ball tensor*. The following cases have to be considered:

- If $\lambda_1 - \lambda_2 > \lambda_2$, the saliency of the stick component is larger than that of the ball component and this indicates certainty of one normal orientation, therefore the token most likely belongs on a curve whose estimated normal is $\hat{e}_1$.

- If $\lambda_1 \approx \lambda_2 > 0$, the dominant component is the ball and there is no preference of orientation, either because all orientations are equally likely, or because multiple orientations coexist at the location. This indicates either a token that belongs to a region, which is surrounded by neighbors from the same regions at all directions, or a junction where two or more curves intersect and multiple curve orientations are present simultaneously (see Fig. 2.5). Junctions can be discriminated from region inliers since their saliency is a distinct peak of λ_2, whereas the saliency of region inliers is more evenly distributed.

- Finally, outliers receive only inconsistent, contradictory votes, so both eigenvalues are small.

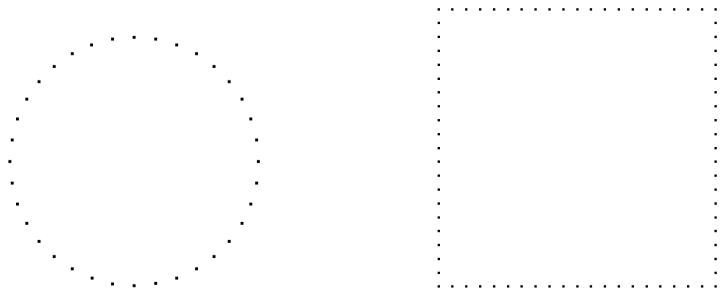
2.2.4 Sensitivity to Scale

The scale of the voting field is the only free parameter of the framework. Nevertheless, the sensitivity to it is low. Similar results should be expected for similar values of scale and small changes in the output are associated with small changes of scale. We begin analyzing the effects of scale with simple synthetic data for which ground truth can be easily computed. The first example is a set of unoriented tokens evenly spaced on the periphery of a circle of radius 100

(a) Junction input (b) Ball saliency map (c) Region input (d) Ball saliency map

Figure 2.5: Ball saliency maps at regions and junctions. Darker pixels in the saliency map correspond to higher saliency than lighter ones. The latter are characterized by a sharp peak of ball saliency.

(see Fig. 2.6(a)). The tokens are encoded as ball tensors and tensor voting is performed to infer the most salient types of structures they form and their preferred orientations. All points are labeled as curvels, while the estimated tangent orientations can be compared to the analytical values, which are given by $[-sin(\theta) \;\; cos(\theta)]^T$.



(a) 72 unoriented tokens on a circle (b) 76 unoriented tokens on a square

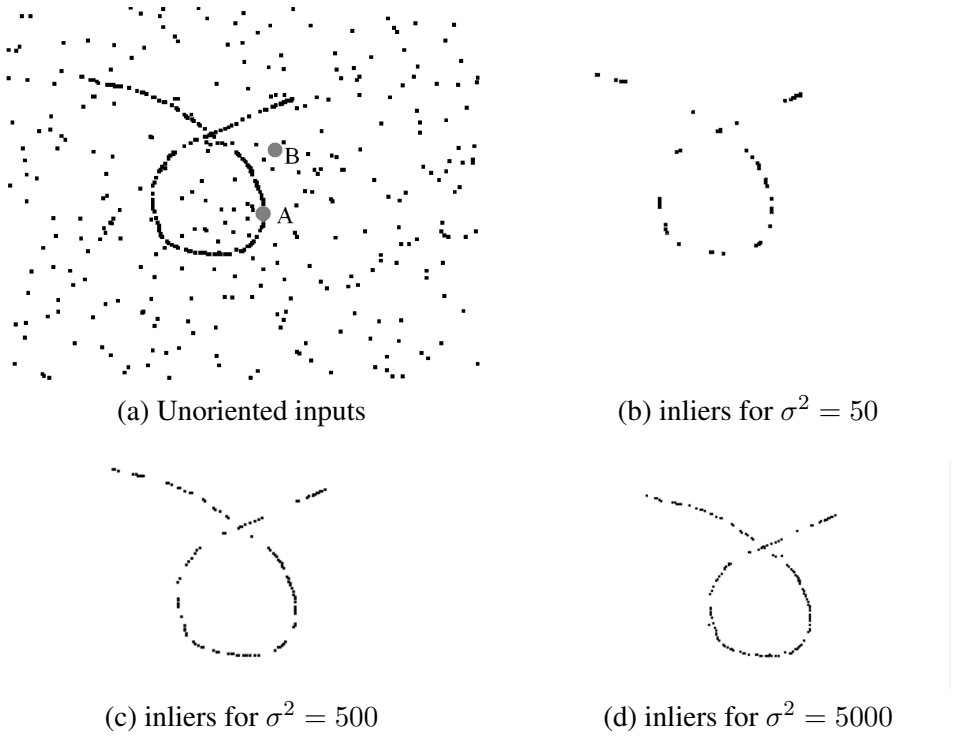
Figure 2.6: Inputs for quantitative estimation of orientation errors as scale varies

Table 2.2 reports the angular error in degrees between the ground truth tangent orientation at each of the 72 locations and the orientation estimated by tensor voting. (Note that the values on the table are for σ^2.) The second example in Fig. 2.6(b) is a set of unoriented tokens lying at equal distances on a square. After voting, the junctions are detected due to their high ball

σ^2	Angular error for circle (degrees)	Angular error for square (degrees)
50	1.01453	1.11601e-007
100	1.14193	0.138981
200	1.11666	0.381272
300	1.04043	0.548581
400	0.974826	0.646754
500	0.915529	0.722238
750	0.813692	0.8893
1000	0.742419	1.0408
2000	0.611834	1.75827
3000	0.550823	2.3231
4000	0.510098	2.7244
5000	0.480286	2.98635

Table 2.2: Errors in curve orientation estimation as functions of scale

saliency and the angular error between the ground truth and the estimated tangent at each curve inlier is reported in Table 2.2. The extreme scale values, especially the larger ones, used here are beyond the range that is frequently used in the other experiments presented in this chapter. A σ^2 value of 50 corresponds to a voting neighborhood with a radius of 16 units of distance, while a σ^2 of 5000 corresponds to a neighborhood of 152 units. Given that the radius of the circle is 100 units and the side of the square is 200 units, the smaller scales are rather small while the larger ones result in voting neighborhoods that include a very large percentage of all tokens in each of them. An observation from these simple experiments is that, as scale increases, each token receives more votes and this leads to a better approximation of the circle by the set of points, while, on the other hand, too much influence from the other sides of the square rounds its corners causing an increase in the reported errors.



(a) Unoriented inputs

(b) inliers for $\sigma^2 = 50$

(c) inliers for $\sigma^2 = 500$

(d) inliers for $\sigma^2 = 5000$

Figure 2.7: Noisy input for inlier/outlier saliency comparison

A different simple experiment demonstrating the stability of the results with respect to large scale variations is shown in Fig. 2.7. The input consists of a set of unoriented tokens from a hand drawn curve. Approximately half the points of the curve have been randomly removed from the data set and the remaining points are spaced apart by 3 units of distance. Random outliers whose coordinates follow a uniform distribution have been added to the data. The 166 inliers and the 300 outliers are encoded as ball tensors and tensor voting is performed with a wide range of scales. The detected inliers at a few scales are shown in Fig. 2.7(b)-(d). At the smallest scales, isolated inliers do not receive enough support to be selected among the most salient tokens, but as scale increases the output becomes stable. In all cases, outliers are

σ^2	Saliency of inlier A	Saliency of outlier B
50	0.140	0
100	0.578	0
200	1.533	0
300	3.369	0
400	4.973	0.031
500	6.590	0.061
750	10.610	0.163
1000	14.580	0.297
2000	30.017	1.118
3000	45.499	2.228
4000	60.973	3.569
5000	76.365	5.100

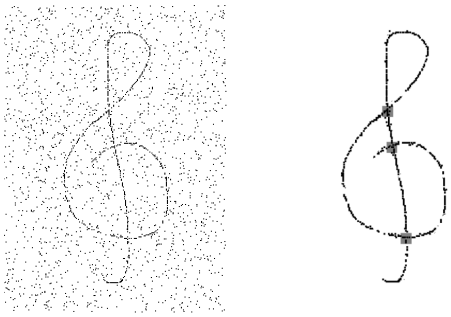
Table 2.3: Errors in curve orientation estimation as functions of scale

successfully removed. Table 2.3 shows the curve saliency of an inlier A and that of an outlier B as the scale varies. (Note that the values on the table again are for σ^2.) Regardless of the scale, the saliency of A is always significantly larger than that of B, making the selection of a threshold that separates inliers from outliers trivial.

2.2.5 Results in 2-D

An experiment on synthetic data can be seen in Fig. 2.8. The input is a set of points which are encoded as ball tensors before voting. After analysis of the eigensystem of the resulting tensors, we can infer the most salient curve inliers and junctions and, at the same time, remove the outliers due to their low saliency.

The example in Fig. 2.9 illustrates the simultaneous detection of regions and curves. The input consists of a cardioid, represented by a set of unoriented points, and two quadratic curves, also represented as sets of unoriented points, contaminated by random uniformly distributed



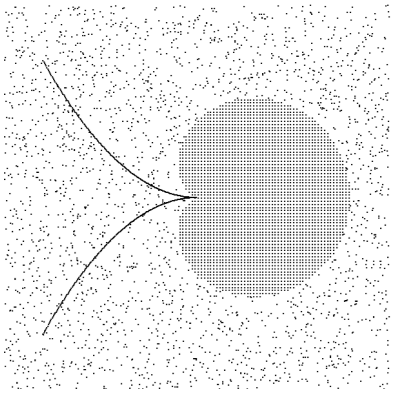
(a) Input (b) Curves and junctions

Figure 2.8: Curves and junctions from a noisy point set. Junctions have been enlarged and marked as squares

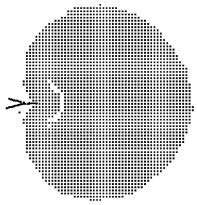
noise (Fig. 2.9(a)). Figures 2.9(b)-(d) show the detected region inliers, region boundaries and curve inliers respectively. Note that some points, where the curves and the cardioid overlap, have been detected as both curve and region inliers.

2.3 Tensor Voting in 3-D

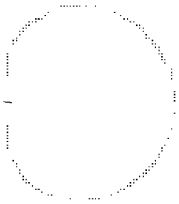
We proceed to the generalization of the framework in 3-D. The remainder of this section shows that no significant modifications need to be made, apart from taking into account that more types of perceptual structure exist in 3-D than in 2-D. In fact, the 2-D framework is a subset of the 3-D framework, which in turn is a subset of the general N-D framework. The second order tensors are now 3-D, while the voting fields are derived from the same *fundamental 2-D second order stick voting field*. In 3-D, the types of perceptual structure that have to be represented are regions (which are now volumes), surfaces, curves and junctions. The inputs can be either unoriented or oriented, in which case there are two types: elementary surfaces (*surfels*) or elementary curves (*curvels*).



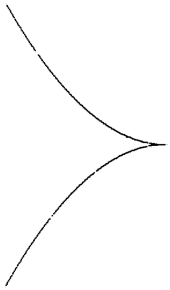
(a) Noisy unoriented data set



(b) Region inliers



(c) Region boundaries



(d) Curve inliers

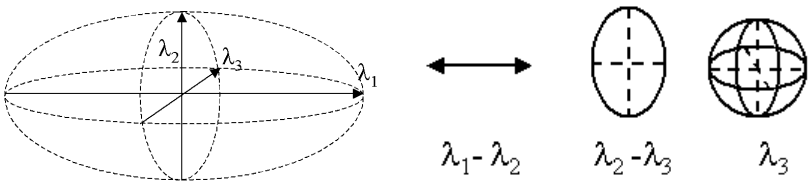
Figure 2.9: Region and curve extraction from noisy data

2.3.1 Representation in 3-D

The representation of a token consists of a 3-D, second order, symmetric, non-negative definite tensor that encodes *saliency* as before. It is equivalent to a 3×3 matrix and a 3-D ellipsoid. The eigenvectors of the tensor are the axes of the ellipsoid and the corresponding eigenvalues are their lengths. The tensor can be decomposed as in the following equation:

$$T = \lambda_1 \hat{e}_1 \hat{e}_1^T + \lambda_2 \hat{e}_2 \hat{e}_2^T + \lambda_3 \hat{e}_3 \hat{e}_3^T =$$

$$= (\lambda_1 - \lambda_2)\hat{e}_1 \hat{e}_1^T + (\lambda_2 - \lambda_3)(\hat{e}_1 \hat{e}_1^T + \hat{e}_2 \hat{e}_2^T) + \lambda_3(\hat{e}_1 \hat{e}_1^T + \hat{e}_2 \hat{e}_2^T + \hat{e}_3 \hat{e}_3^T) \quad (2.8)$$



(a) A 3-D generic tensor (λ_i are its eigenvalues in descending order)

(b) Decomposition into the *stick*, *plate* and *ball* components

Figure 2.10: A second order generic tensor and its decomposition in 3-D

where λ_i are the eigenvalues in decreasing order and $\hat{e}_i$ are the corresponding eigenvectors (see also Fig. 2.10). The first term in Eq. 2.8 corresponds to a *3-D stick tensor*, that indicates an elementary surface token with $\hat{e}_1$ as its surface normal. The second term corresponds to a degenerate disk-shaped ellipsoid, termed hereafter the *plate tensor*, that indicates a curve or a surface intersection with $\hat{e}_3$ as its tangent, or, equivalently with $\hat{e}_1$ and $\hat{e}_2$ spanning the plane normal to the curve. Finally, the third term corresponds to a *3-D ball tensor*, that corresponds to a structure which has no preference of orientation. Table 2.4 shows how oriented and unoriented inputs are encoded and the equivalent ellipsoids and quadratic forms.

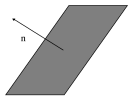
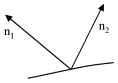
The representation using normals instead of tangents can be justified more easily in 3-D, where surfaces are arguably the most frequent type of structure. In 2-D, normal or tangent representations are equivalent. A surface patch in 3-D is represented by a stick tensor parallel to the patch's normal. A curve, which can also be viewed as a surface intersection, is represented by a plate tensor that is normal to the curve. All orientations orthogonal to the curve belong in the 2-D subspace defined by the plate tensor. Any two of these orientations that are orthogonal to each other can be used to initialize the plate tensor (see also Table 2.4). Adopting this representation allows a structure with $N - 1$ degrees of freedom in N-D (a curve in 2-D, a

surface in 3-D) to be represented by a single orientation, while a tangent representation would require the definition of $N - 1$ vectors that form a basis for an (N-1)-D subspace. Assuming that this is the most frequent structure in the N-D space, our choice of representation makes the *stick voting field*, which corresponds to the elementary (N-1)-D variety, the basis from which all other voting fields are derived. In addition, this choice makes the handling of intersections considerably easier. Using a representation based on normals, intersections are represented as the union of the normal spaces of each surface, for instance, which can be computed with the Gramm-Schmidt algorithm. On the other hand, using a representation based on tangents, the same operation would require the more cumbersome computation of the intersection of the tangent spaces.

2.3.2 Voting in 3-D

Identically to the 2-D case, voting begins with a set of oriented and unoriented tokens. We begin by showing how a voter with a purely stick tensor generates and casts votes, and then, derive the voting fields for the plate and ball cases. We chose to keep voting a function of only the position of the receiver relative to the voter and of the voter's preference of orientation. Therefore, we again address the problem of finding the smoothest path between the voter and receiver by fitting arcs of the osculating circle, as described in Section 2.2.1.

Note that the voter, the receiver and the stick tensor at the voter define a plane. The voting procedure is restricted on this plane, thus making it identical to the 2-D case. The second order vote, which is the surface normal at the receiver under the assumption that the voter and receiver belong to the same smooth surface, is also a purely stick tensor on the plane (see also Fig. 2.2).

Input	Tensor	Eigenvalues	Quadratic form
 surfel		$\lambda_1 = 1, \lambda_2 = \lambda_3 = 0$	$\begin{bmatrix} n_1^2 & n_1n_2 & n_1n_3 \\ n_1n_2 & n_2^2 & n_2n_3 \\ n_1n_3 & n_2n_3 & n_3^2 \end{bmatrix}$
 curvel		$\lambda_1 = \lambda_2 = 1, \lambda_3 = 0$	**P** (see below)
unoriented		$\lambda_1 = \lambda_2 = \lambda_3 = 1$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$\textbf{P=} \begin{bmatrix} n_{11}^2 + n_{21}^2 & n_{11}n_{12} + n_{21}n_{22} & n_{11}n_{13} + n_{21}n_{23} \\ n_{11}n_{12} + n_{21}n_{22} & n_{12}^2 + n_{22}^2 & n_{12}n_{13} + n_{22}n_{23} \\ n_{11}n_{13} + n_{21}n_{23} & n_{12}n_{13} + n_{22}n_{23} & n_{13}^2 + n_{23}^2 \end{bmatrix}$$

Table 2.4: Encoding oriented and unoriented 2-D inputs as 2-D second order symmetric tensors

The magnitude of the vote is defined by the same *saliency decay function*, duplicated here for completeness.

$$DF(s, \kappa, \sigma) = e^{-(\frac{s^2 + c\kappa^2}{\sigma^2})} \tag{2.9}$$

Sparse, token to token voting is performed to estimate the preferred orientation of tokens, followed by dense voting during which saliency is computed at every grid position.

Voting by any 3-D tensor takes place by decomposing the tensor into its three components: the stick, the plate and the ball. Votes are retrieved from the appropriate voting field by look-up operations and are multiplied by the saliency of each component. Stick votes are weighted by

$\lambda_1 - \lambda_2$, plate votes by $\lambda_2 - \lambda_3$ and ball votes by λ_3. The 3-D stick voting field can be derived from the fundamental 2-D stick field by rotation about the voting stick, which is the axis of symmetry of the 3-D field. The visualization of the 2-D second order stick field in Fig. 2.3(a) is also a cut of the 3-D field that contains the stick tensor at the origin.

To show the derivation of the ball voting field $\mathbf{B}_{so}(P)$ from the stick voting field, we can visualize the vote at P from a unit ball tensor at the origin O as the integration of the votes of stick tensors that span the space of all possible orientations. In 2-D, this is equivalent to a rotating stick tensor that spans the unit circle at O, while in 3-D the stick tensor spans the unit sphere. The 3-D ball field can be derived from the second order stick field $\mathbf{S}_{so}(P)$, as follows:

$$\mathbf{B}(P)_{so} = \int_0^{2\pi} \int_0^{2\pi} R_{\theta\phi\psi}^{-1} \mathbf{S}_{so}(R_{\theta\phi\psi} P) R_{\theta\phi\psi}^{-T} d\phi d\psi|_{\theta=0} \tag{2.10}$$

where $R_{\theta\phi\psi}$ is the rotation matrix to align $\mathbf{S}$ with $\hat{e}_1$, the eigenvector corresponding to the maximum eigenvalue (the stick component), of the rotating tensor at P, and θ, ϕ, ψ are rotation angles about the x, y, z axis respectively. The integration is approximated by tensor addition, $T = \sum \vec{v}_i \vec{v}_i^T$. Note that normalization has to be performed in order to make the energy emitted by a unit ball equal to that of a unit stick. The field is radially symmetric, as expected, since the voter has no preferred orientation.

To complete the description of the voting fields for the 3-D case, we need to describe the second order *plate voting field* $\mathbf{P}_{so}(P)$. Since the plate tensor encodes uncertainty of orientation around one axis, it can be derived by integrating the votes of a rotating stick tensor that spans

the unit circle, in other words the plate tensor. The formal derivation is analogous to that of the ball voting fields and can be written as follows:

$$\mathbf{P}_{so}(P) = \int_0^{2\pi} R_{\theta\phi\psi}^{-1} \mathbf{S}_{so}(R_{\theta\phi\psi} P) R_{\theta\phi\psi}^{-T} d\psi|_{\theta=\phi=0} \tag{2.11}$$

where θ, ϕ, ψ, and $R_{\theta\phi\psi}$ have the same meaning as in the previous equation. Normalization has to be performed in order to make the total energy of the ball and plate voting fields equal to that of the stick voting fields. The sum of the maximum eigenvalues of each vote is used as the measure of energy.

2.3.3 Vote Analysis

Analysis of the second order votes can be performed once the eigensystem of the accumulated second order 3×3 tensor has been computed. Then the tensor can be decomposed into the stick, plate and ball components:

$$T = (\lambda_1 - \lambda_2)\hat{e}_1\hat{e}_1^T + (\lambda_2 - \lambda_3)(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T) + \lambda_3(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T + \hat{e}_3\hat{e}_3^T). \tag{2.12}$$

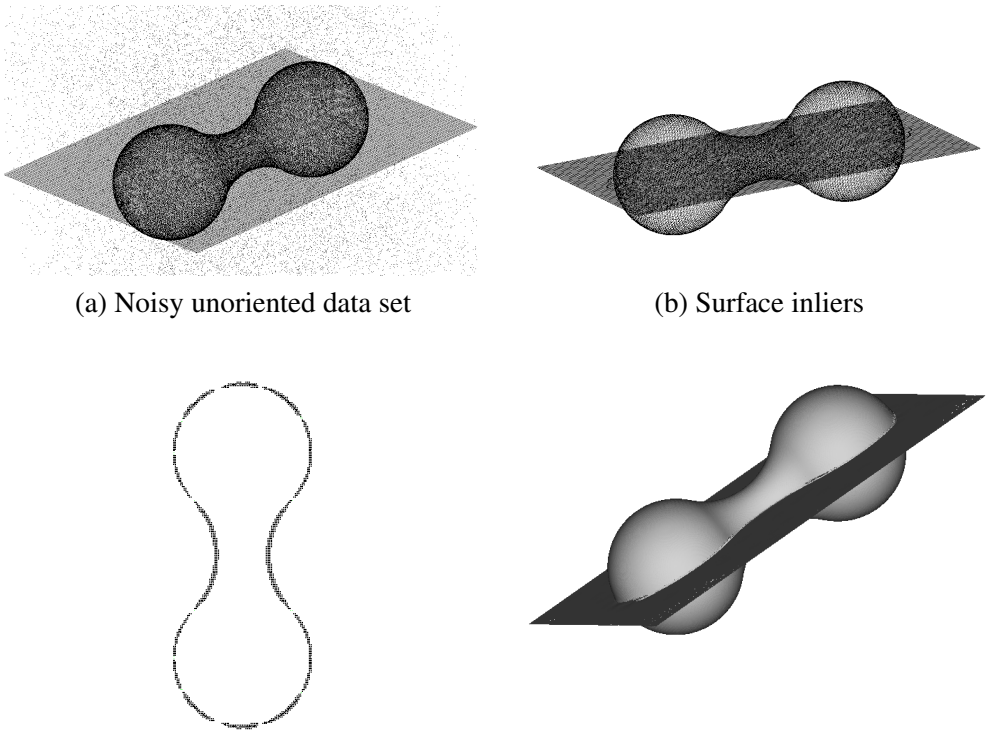
where $\hat{e}_1\hat{e}_1^T$ is a *stick tensor*, $\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T$ is a *plate tensor*, $\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T + \hat{e}_3\hat{e}_3^T$ is a *ball tensor.* The following cases have to be considered:

- If $\lambda_1 - \lambda_2 > \lambda_2 - \lambda_3$ and $\lambda_1 - \lambda_2 > \lambda_3$, the stick component is dominant. Thus the token most likely belongs on a surface whose normal is $\hat{e}_1$.

- If $\lambda_2 - \lambda_3 > \lambda_1 - \lambda_2$ and $\lambda_2 - \lambda_3 > \lambda_3$, the plate component is dominant. In this case the token belongs on a curve or a surface intersection. The normal plane to the curve or the surface orientation is spanned by $\hat{e}_1$ and $\hat{e}_2$. Equivalently, $\hat{e}_3$ is the tangent.

- If $\lambda_3 > \lambda_1 - \lambda_2$ and $\lambda_3 > \lambda_2 - \lambda_3$, the ball component is dominant and the token has no preference of orientation. It is either a junction or it belongs in a volume. Junctions can be discriminated from volume inliers since they are distinct peaks of λ_3.

- Outliers receive only inconsistent, contradictory votes, so both eigenvalues are small.



(a) Noisy unoriented data set

(b) Surface inliers

(c) Curve inliers

(d) Dense surfaces

Figure 2.11: Inference of surfaces and surface intersections from noisy data

2.3.4 Results in 3-D

Due to space considerations and to the fact that more challenging experiments are presented in Chapter 3, we present results on just one synthetic 3-D dataset. The example in Fig. 2.11 illustrates the simultaneous inference of surfaces and curves. The input consists of a "peanut" and a plane, encoded as unoriented points contaminated by random uniformly distributed noise (Fig. 2.11(a)). The "peanut" is empty inside, except for the presence of noise, which has an equal probability of being anywhere in space. Figure 2.11(b) shows the detected surface inliers, after tokens with low saliency have been removed. Figure 2.11(c) shows the curve inliers, that is the tokens that lie at the intersection of the two surfaces. Finally, Fig. 2.11(d) shows the extracted dense surfaces.

Chapter 3

Integration of First Order Information

This chapter presents a significant contribution of our work to the tensor voting framework: the integration of first order representation and voting that enable us to infer terminations in perceptual structures. Many computer vision and machine learning applications require the reliable detection of boundaries, which is a particularly challenging problem in the presence of outliers and missing data. Here, we propose to address it by complementing the original tensor voting framework, which is limited to second order properties, with first order information. First order voting fields and a mechanism to vote for the terminations of perceptual structures, such as the endpoints of curves, are defined in the remainder of this chapter. We take great care to ensure that the integration of first order properties is in accordance with the philosophy of the framework. Namely, we maintain the capability to represent *all* structure types *simultaneously* and adhere to the principle of least commitment. What should be noted is that the first order representation, even though it is complementary to the second order one, can represent all boundary types. The work of this chapter has been published in [93] and also in [161] in conjunction with

concurrent research by W.S. Tong and C.K. Tang in the Hong Kong University of Science and Technology.

3.1 Motivation

The strictly second order formulation of the original tensor voting framework, as shown in [92] and the previous chapters, is very effective for many perceptual organization problems, but it is unable to detect terminations of open structures such as the endpoints of curves. It can be viewed as an excitatory process that facilitates grouping of the input data, and is able to extrapolate and infer dense salient structures. The integration of boundary inference, via first order voting, provides a mechanism to inhibit the growth of the extracted structures. *Polarity vectors* are now associated with each token and encode the support the token receives for being a termination of a perceptual structure. The term *polarity* refers to the magnitude of the polarity vector and is large when the majority of the token's neighbors lie on the same side. The direction of the polarity vector indicates the direction of the inliers of the perceptual structure whose potential boundary is the token under consideration. The new representation exploits the essential property of boundaries to have all their neighbors, at least locally, on the same side of a half-space. As described in the remainder of the chapter, the voting scheme is identical to that of the second order case, and the first order vector voting fields can be easily derived from the second order fundamental tensor voting field.

Figure 3.1 illustrates the motivation behind the first order augmentation to the framework. The contour consists of a number of "smooth inliers", such as A and B, which can be inferred from their neighbors under the constraint of good continuation. Consider point C, which is an

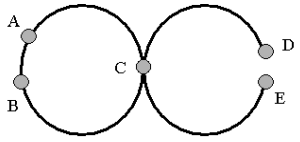
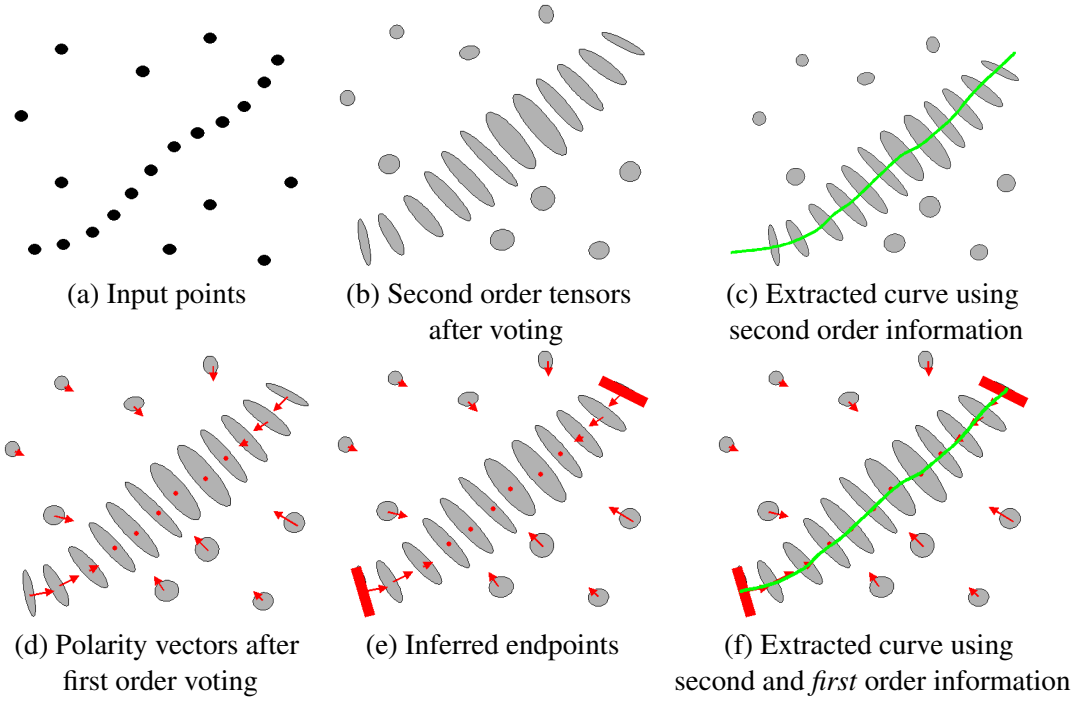


Figure 3.1: A contour with a junction and two endpoints

X-junction where two smooth segments of the contour intersect. It can be represented as having both curve normals simultaneously. This is a second order discontinuity that is described by the presence of a salient ball tensor. On the other hand, the endpoints D and E of the contour are also smooth continuations of their neighbors and are represented as salient stick tensors. Therefore, they cannot be qualitatively discriminated from points A and B using second order properties only. The property that makes these pairs of points very different is that D and E are terminations of the curve while A and B are not. This is captured by the polarity vector, whose magnitude is very close to zero for A and B, which are surrounded by other curvels, and locally maximum for D and E, which receive stronger support from one side of the contour only. Polarity allows us to infer richer descriptions that make the distinction between interior points and endpoints, that was missing before.

A simple illustration of first order voting can be seen in Fig. 3.2. The inputs are a few unoriented points that form a curve and some outliers (Fig. 3.2(a)), which are encoded as ball tensors. After second order voting takes place, the inliers of the curves develop high stick saliency and the major axes of their tensors align with the normal orientation at each position (Fig. 3.2(b)). If we try to extract a continuous curve based on the strictly second order formulation of [92], we begin marching from the most salient token and stop when stick saliency drops below a threshold. The resulting curve is shown in Fig. 3.2(c), where clearly it has grown



(a) Input points (b) Second order tensors after voting (c) Extracted curve using second order information

(d) Polarity vectors after first order voting (e) Inferred endpoints (f) Extracted curve using second and *first* order information

Figure 3.2: Illustration of curve extraction with and without first order voting. In the former case, even though the curve normals have been estimated correctly, there is no way other than heuristic thresholding to detect the endpoints and the curve extends beyond them, as seen in (c). On the other hand, when first order information is available as in (d), the endpoints can be inferred as in (e) and the curve is terminated correctly as shown in (f).

beyond the endpoints, and is not consistent with the human interpretation of the input. If we perform,instead, first order voting before curve extraction, we obtain a polarity vector at each token position (Fig. 3.2(d)). Non-maximum suppression of polarity values can, then, be performed along the direction of the polarity vectors to detect the endpoints (Fig. 3.2(e)). Given the endpoints, the correct open curve can be extracted and is shown in Fig. 3.2(f).

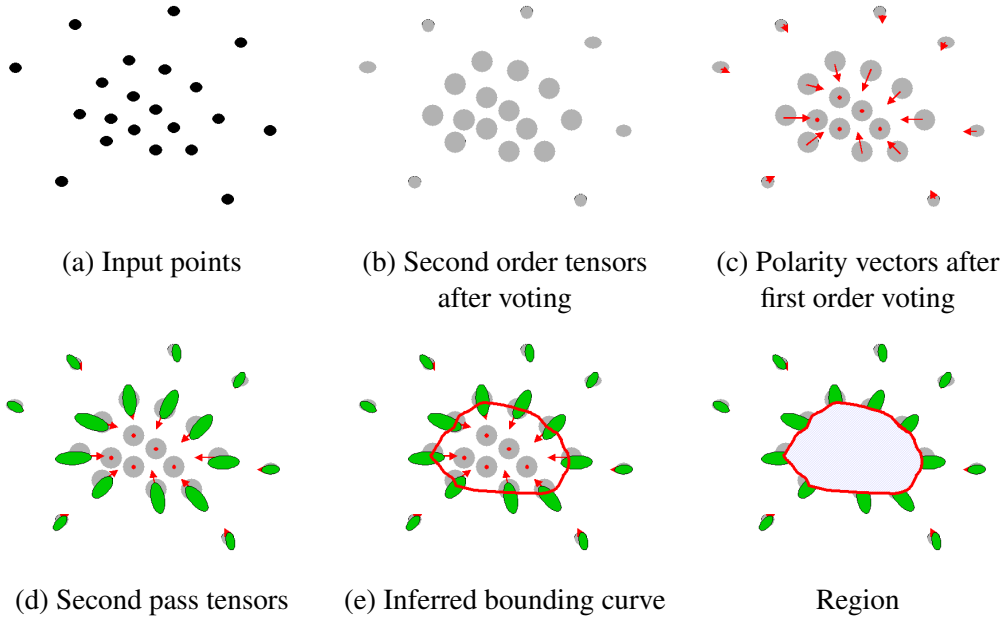
The same principle can be applied to region inference. We propose to infer regions via their boundaries, which in 2-D are curves consisting of tokens with high polarity. Figure 3.3 illustrates the process.

This chapter is organized as follows:

- Section 3.2 presents the first order representation, voting and voting fields and shows how they are naturally derived from their second order counterparts.
- Section 3.3 shows how the accumulated first and second order votes are analyzed to infer salient perceptual structures.
- Section 3.4 contains results on 2-D and 3-D synthetic, but challenging, datasets.
- Section 3.5 concludes the chapter with a discussion on the contributions presented here and possible extensions.

3.2 First Order Representation, Voting and Voting Fields

The first order information is conveyed by the polarity vector that encodes the likelihood of the token being on the boundary of a perceptual structure. Such boundaries in 2-D are the endpoints of curves and the bounding curves of regions. In 3-D, the possible types of boundaries are the bounding surfaces of volumes, the bounding curves of surfaces and the endpoints of curves. The direction of the polarity vector, in all cases, indicates the direction of the inliers of the perceptual structure whose potential boundary is the token under consideration. As before, the second order tensor indicates the saliency of each type of perceptual structure the token belongs to and



(a) Input points (b) Second order tensors after voting (c) Polarity vectors after first order voting

(d) Second pass tensors (e) Inferred bounding curve Region

Figure 3.3: Illustration of region extraction. The inputs, in (a), are unoriented points encoded as ball tensors. The region inliers accumulate salient ball tensors after second order voting, while the saliency of the outliers is smaller, as shown in (b). The boundaries accumulate consistent first order votes and thus develop large polarity vectors shown in (c). These can be used as inputs for another pass of voting where points on salient boundary curves receive support from their neighbors, (d). Then a continuous curve can be extracted as in (e) and the desired region is the area enclosed by the bounding curve (f).

its preferred normal and tangent orientations. The two parts of the representation combined provide a much richer description of the tokens.

Now, we turn our attention to the generation of first order votes. As shown in Fig. 3.4, the first order vote cast by a unit stick tensor at the origin is *tangent* to the osculating circle, the smoothest path between the voter and receiver. Its magnitude, since nothing suggests otherwise,

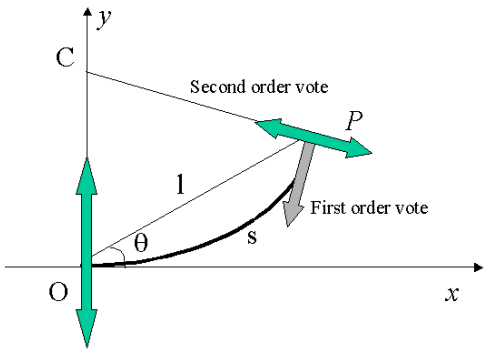


Figure 3.4: Second and first order votes cast by a stick tensor located at the origin

is equal to that of the second order vote according to Eq. 2.2. The first order voting field for a unit stick voter aligned with the z-axis is:

$$\mathbf{S}_{FO}(l,\theta,\sigma) = DF(s,\kappa,\sigma)\begin{bmatrix} -cos(2\theta) \\ -sin(2\theta) \end{bmatrix} \tag{3.1}$$

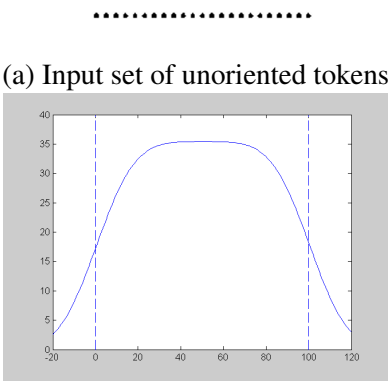
What should be noted is that *tokens cast first and second order votes based on their second order information only*. This occurs because polarity vectors have to be initialized to zero since no assumption about structure terminations is available. Therefore, first order votes are computed based on the second order representation which can be initialized (in the form of ball tensors) even with no information other than the presence of a token. However, if first order information was available, as a result of an endpoint detector for instance, it can be incorporated naturally in the representation.

A simple illustration of how first and second order votes can be combined to infer curves and their endpoints in 2-D appears in Fig. 3.5. The input consists of a set of colinear unoriented tokens which are encoded as ball tensors. The tokens cast votes to their neighbors and collect

the votes cast to them. The accumulated curve saliency can be seen in Fig. 3.5(b), where the dashed lines mark the limits of the input. The interior points of the curve receive more support and are more salient than those close to the endpoints. Since second order voting is an excitatory process, locations beyond the endpoints are also compatible with the inferred line and receive consistent votes from the input tokens. Detection of the endpoints based on the second order votes is virtually impossible since there is no systematic way of selecting a threshold that guarantees that the curve will not extend beyond the leftmost and rightmost input tokens. The accumulated polarity can be seen in Fig. 3.5(c). The endpoints appear clearly as maxima of polarity. The combination of saliency and polarity allows us to infer the curve and terminate it at the correct points. See the following section for a complete analysis of structure inference.

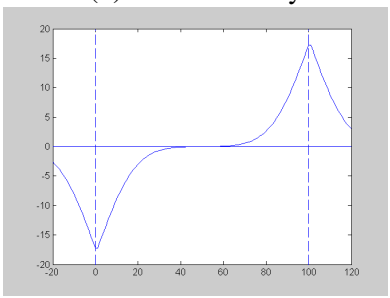
The *2-D first order stick voting field* $\mathbf{S_{fo}}$ is a vector field, which at every position holds a vector that is equal in magnitude to the stick vote that exists at the same position in the fundamental second order stick voting field, but is tangent to the smooth path between the voter and receiver instead of normal to it. The *first order ball voting field* $\mathbf{B_{fo}}$ can be derived, as the second order one, by integrating the contributions of a rotating stick tensor that casts first order votes. The integration this time is approximated by vector instead of tensor addition. In 2-D this is accomplished as follows:

$$\mathbf{B_{fo}}(P) = \int_0^{2\pi} R_\theta^{-1} \mathbf{S_{fo}}(R_\theta P) R_\theta^{-T} d\theta \tag{3.2}$$



(a) Input set of unoriented tokens

(b) Curve saliency



(c) Polarity

Figure 3.5: Accumulated saliency and polarity for a simple input

where R_θ is the rotation matrix to align $\mathbf{S_{fo}}$ with $\hat{e}_1$, the eigenvector corresponding to the maximum eigenvalue (the stick component), of the rotating tensor at P. The 3-D case is similar, but now the rotating stick tensor spans the unit sphere:

$$\mathbf{B_{fo}}(P) = \int_0^\pi \int_0^\pi R_{\theta\phi\psi}^{-1} \mathbf{S_{fo}}(R_{\theta\phi\psi} P) R_{\theta\phi\psi}^{-T} d\phi d\psi|_{\theta=0} \tag{3.3}$$

where $R_{\theta\phi\psi}$ is the rotation matrix to align $\mathbf{S_{fo}}$ with $\hat{e}_1$, the eigenvector corresponding to the maximum eigenvalue (the stick component), of the rotating tensor at P, and θ, ϕ, ψ are rotation angles about the x, y, z axis respectively.

To complete the description of the voting fields for the 3-D case, we need to describe the *first order plate voting field* $\mathbf{P_{fo}}$. Since the plate tensor encodes uncertainty of orientation around one axis, it can be derived by integrating the votes of a rotating stick tensor that spans the unit circle, simulating a plate tensor. The derivation is analogous to that of the ball voting fields and can be written as follows:

$$\mathbf{P_{fo}}(P) = \int_0^{\pi} R_{\theta\phi\psi}^{-1} \mathbf{S_{fo}}(R_{\theta\phi\psi} P) R_{\theta\phi\psi}^{-T} d\psi|_{\theta=\phi=0} \tag{3.4}$$

where θ, ϕ, ψ, and $R_{\theta\phi\psi}$ have the same meaning as in the previous equation. As in the second order case, the voting fields are normalized so that the total energy is equal to that of the stick voting field. The norm of the vector votes is used as the measure of energy.

As in the second order case, arbitrary tensors are decomposed into elementary tensors as in Eq. 2.1 and Eq. 2.8, which cast votes using the appropriate fields. The vote of each component is weighted by the appropriate saliency value.

3.3 Vote Analysis

Vote collection for the first order case is performed by vector addition. The accumulated result is a vector whose direction points to a weighted center of mass from which votes are cast, and whose magnitude encodes polarity. Since the first order votes are also weighted by the saliency of the voters and attenuate with distance and curvature, their vector sum points to the direction from which the most salient contributions were received. The accumulated first order votes provide information that complements the accumulated second order saliency information.

2-D Feature	Saliency	Second order tensor orientation	Polarity	Polarity vector
curve interior	locally max $\lambda_1 - \lambda_2$ along $\hat{e}_1$	normal: $\hat{e}_1$	low	-
curve endpoint	locally max $\lambda_1 - \lambda_2$ along $\hat{e}_1$	normal: $\hat{e}_1$	locally max along pol. vec.	parallel to $\hat{e}_2$
region interior	high λ_2	-	locally non-max	-
region boundary	high λ_2	-	locally max along pol. vec.	normal to boundary
junction	local peak of λ_2	-	depends on type	-
outlier	low	-	low	-

Table 3.1: Summary of first and second order tensor structure for each feature type in 2-D

In 2-D, a relatively low polarity value indicates that a token is in the interior of a curve or region, therefore surrounded by neighbors whose votes cancel each other out. On the other hand, high polarity indicates a token that is on or close to a boundary, thus receiving votes from only one side with respect to the boundary, at least locally. The correct boundaries can be extracted as local maxima of polarity along the direction of the polarity vector. Table 3.1 illustrates how tokens can be characterized using the collected first and second order information. In more detail, the cases that have to be considered are the following:

- *Interior points of curves* can be found as local maxima of $\lambda_1 - \lambda_2$ along $\hat{e}_1$. In other words, they form the path of maximal curve saliency as one is marching along the curve's tangent starting from any token on the curve. Interior points receive first order votes from

both sides, which virtually cancel each other out. Their polarity is not a local maximum along the direction of the polarity vector.

- *Curve endpoints* have the same second order properties, but they can be detected as local maxima of polarity along the direction of the polarity vector.

- *Interior points of regions* have high λ_2 values, as a result of the higher density of surrounding points. In general, for datasets that contain regions with different densities, λ_2 is a measure of density. For instance, for datasets with two distinct densities, λ_2 has two modes. In this case, the largest mode and its surrounding values can be classified as region inliers. This can be extended for datasets with more than two modes, as long as a threshold that determines the density required to classify a token as a region inlier is set.

- *Region boundaries* are region tokens that also have locally maximum polarity along the direction of the polarity vector. The latter is also orthogonal to the region boundaries.

- *Junctions* are isolated peaks of λ_2 as shown in Fig. 2.5. Their polarity depends on the type of junction. It is very small for X and W-junctions, while it is high for L and T-junctions.

- *Outliers* receive very little consistent support and have low saliency values compared to those of the inliers. Since they are in regions of low data density, their polarity values are also small due to the fact that votes attenuate with distance.

In 3-D, tokens can be classified according to the accumulated first and second order information according to Table 3.2. The accumulated second order tensor is decomposed as in Section 2.3.3 and stick, plate and ball saliencies are computed based on the eigenvalues.

The tokens can be classified as follows:

- *Interior points of surfaces* have locally maximal $\lambda_1 - \lambda_2$ along the direction of the surface normal, $\hat{e}_1$. The surface in dense form can be extracted by the marching cubes algorithm [84] as the zero-level of the first derivative of surface saliency. Unlike levels sets and marching cubes, in our case, the surface does not have to be closed. Interior points have non-maximal polarity values.

- *Surface boundaries* have the same second order properties as the above, but also have locally maximal polarity along the direction of the polarity vector. The latter is orthogonal to the surface end-curve at the token's location.

- *Interior points of curves* have the same properties as in the 2-D case, but now the appropriate saliency is given by $\lambda_2 - \lambda_3$. For a token to be on a curve its curve saliency has to be locally maximal on a plane normal to the curve spanned by $\hat{e}_1$ and $\hat{e}_2$, which are the two normals to the curve.

- *Curve endpoints* have the same properties as in the 2-D case. The polarity vector in both cases is parallel to the tangent of the curve, the endpoints of which can be detected by their locally maximal polarity.

- *Interior points of regions* are characterized by the same properties as in the 2-D case. The differences are that, in 3-D, ball saliency is given by λ_3 and that regions are volumes.

- *Region boundaries* are volume boundaries in 3-D and the analysis is the same as in the 2-D case. The polarity vector is orthogonal to the bounding surface of the volume.

3-D Feature	Saliency	Second order tensor orientation	Polarity	Polarity vector
surface interior	locally max $\lambda_1 - \lambda_2$ along $\hat{e}_1$	normal: $\hat{e}_1$	low	-
surface end-curve	locally max $\lambda_1 - \lambda_2$ along $\hat{e}_1$	normal: $\hat{e}_1$	locally max $\lambda_1 - \lambda_2$ along pol. vec.	orthogonal to $\hat{e}_1$ and end-curve
curve interior	locally max $\lambda_2 - \lambda_3$ on $\hat{e}_1$, $\hat{e}_2$ plane	tangent: $\hat{e}_3$	low	-
curve endpoint	locally max $\lambda_2 - \lambda_3$ on $\hat{e}_1$, $\hat{e}_2$ plane	tangent: $\hat{e}_3$	locally max along pol. vec.	parallel to $\hat{e}_3$
region interior	high λ_3	-	low	-
region boundary	high λ_3	-	locally max along pol. vec	normal to bounding surface
junction	local peak of λ_3	-	depends on type	-
outlier	low	-	low	-

Table 3.2: Summary of first and second order tensor structure for each feature type in 3-D

- *Junctions* are local peaks of λ_3. Their polarity depends on the type of junction as in the 2-D case. Convex vertices of polyhedra have polarity vectors that point to the interior of the polyhedron, while the polarity vector points to the outside at concave vertices.

- *Outliers*, as before, receive very little consistent support and have low saliency and polarity values.

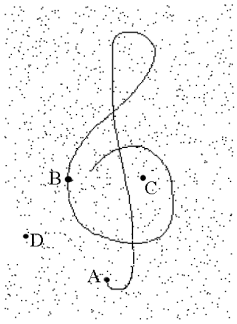
The dense structure extraction algorithms presented by Tang in [151] can be modified to take first order information into account. Both the curve and surface extraction algorithms depended on a "low saliency threshold" which the tokens had to exceed for the curve or surface

to continue to grow. This is no longer necessary, since curve extraction stops once an endpoint is reached and surface extraction stops at surface boundaries. Even though the selection of this threshold is not critical, the integration of first order information reduces the number of free parameters and makes the extraction of dense structures more systematic.

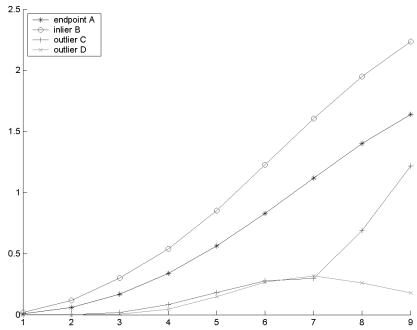
3.3.1 Sensitivity to Scale

Before proceeding to challenging experiments, we conducted a simple evaluation of the stability of both first and second results with respect to scale. The points of Fig. 3.6(a) are encoded as ball tensors and used as inputs for first and second order voting with a large range of scales. In fact the radius of the voting field was 9 pixels for the smallest scale used, and 138 pixels for the largest one, while the input dimensions are 612×846. We identified an endpoint (A), a curve inlier (B) and two outliers, one close to the inliers (C) and one isolated (D), for which we measured the curve saliency and polarity. The logarithms of these two quantities are plotted on Fig. 3.6(b) and (c).

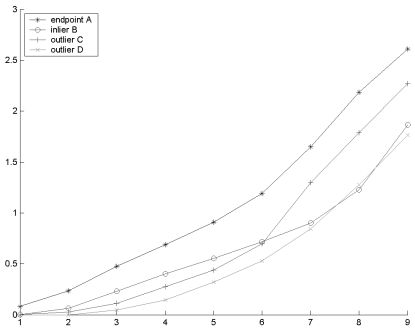
Regardless of the scale of the voting fields, the curve inlier B always has the highest curve saliency value and the endpoint A always has the second highest value. The outliers have clearly smaller values, which are zero at small scales. In terms of polarity, the endpoint A has clearly larger values than all other points. The curve inlier B does not have zero polarity, since it is on the left side of the inliers. If one considers, however, that it can be easily classified as a curve inlier, and projects the polarity on the curve tangent $\hat{e}_2$, then the magnitude of the projected polarity is very close to zero. On the contrary, the polarity vector and the curve tangent are



(a) Noisy unoriented data set



(b) Curve saliency



(c) Polarity

Figure 3.6: Noisy input where the four points that we have selected are marked by large black circles. The curve saliency and polarity plots are logarithmic. The 9 measurements on the x-axis, correspond to $sigma^2 = 20, 40, 80, 160, 320, 640, 1280, 2560$, and 5120.

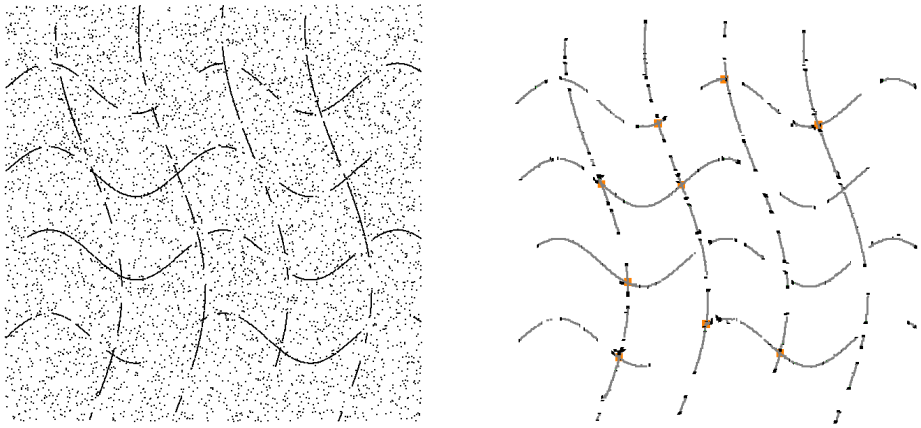
almost parallel at the endpoint. The outliers can accumulate polarity comparable to that of the inliers, but they can safely be rejected due to their low saliency.

3.4 Results using First Order Information

In this section, we present results on synthetic datasets corrupted by noise. In all cases, processing begins with unoriented tokens, which are encoded as ball tensors. The capability to proceed

with oriented or unoriented tokens, or a mixture of both, is a feature of tensor voting not shared by many other approaches.

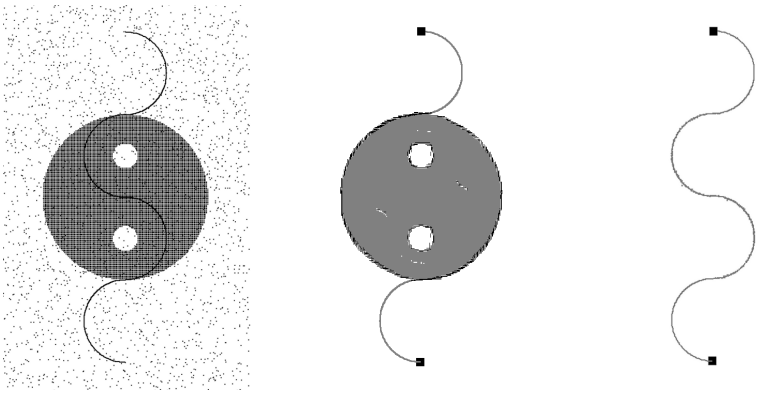
Results in 2-D Figure 3.7(a) shows a data set that contains a number of fragmented sinusoidal curves represented by unoriented points contaminated by a large number of outliers. All points are encoded as ball tensors. Figure 3.7(b) shows the output after tensor voting. Curve inliers are colored gray, endpoints black, while junctions appear as gray squares. The noise has been removed, and the curve segments have been correctly detected and their endpoints and junctions labeled.



(a) Noisy unoriented data set (b) Extracted curves, endpoints and junctions

Figure 3.7: Curve, endpoint and junction extraction on a noisy dataset with sinusoidal curves

More results are illustrated in Fig. 3.8, which demonstrates the simultaneous detection of a region and a curve, as well as their terminations. The curve is inferred even as it goes through the region, since curve saliency is still locally maximal due to the higher density of curve inliers compared to the region inliers. The curve by itself is shown in Fig. 3.8(c) for clarity.

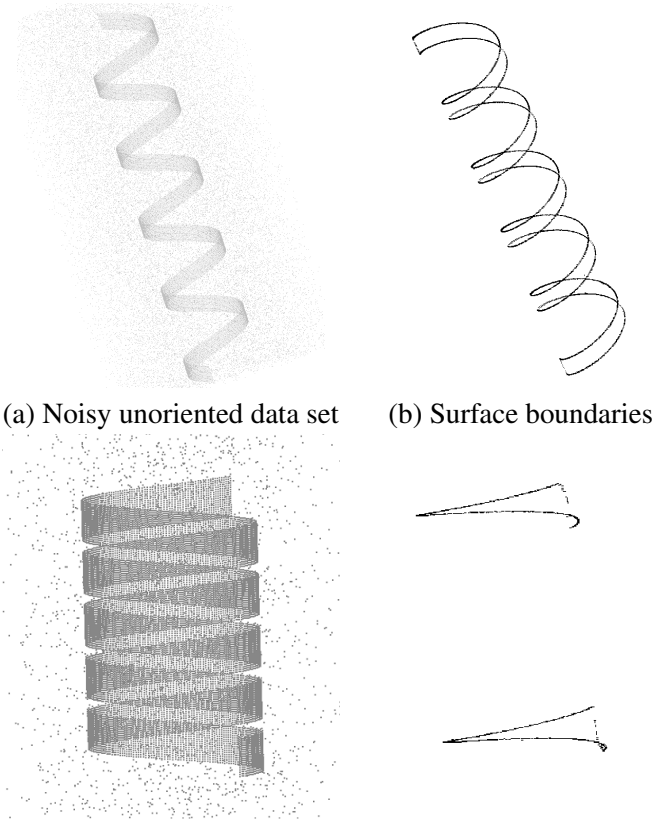


(a) Input (c) Regions, boundaries, curves and endpoints Curves and enpoints

Figure 3.8: Regions and curves from a noisy point set. Region boundaries and curve endpoints are marked in black. The curve is shown by itself on the right to aid visualization.

Results in 3-D The first example is on a dataset that contains a surface in the from of a spiral inside a cloud of noisy points (Fig. 3.9). Both the surface and the noise are encoded as unoriented points. The spiral consists of 19,000 points, while the outliers are 30,000. Figure 3.9(b) shows the surface boundaries detected after voting. As the spiral is tightened to resemble a cylinder (Fig. 3.9(c)), the surfaces merge and the inferred boundaries are those of a cylinder (Fig. 3.9(d)).

Given a noisy set of points that belong to a 3-D region, as in Fig. 3.10(a), we infer volume boundaries as local maxima of polarity along the direction of the polarity vector. In terms of second order tensors, volume inliers are characterized by a dominant ball component, since they collect second order votes from all directions in 3-D. The same holds for tokens close to the boundaries, since second order votes are function of orientation but not direction. The bounding surface of a 3-D region can be extracted by the modified surface marching algorithm [151, 152] as the maximal isosurface of polarity along the normal direction, indicated by the

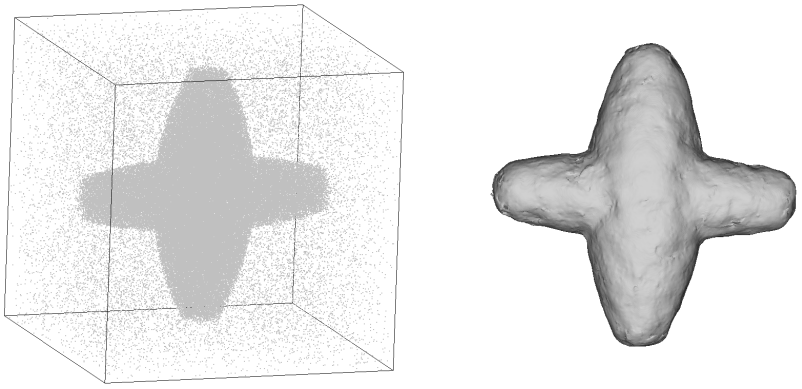


(a) Noisy unoriented data set (b) Surface boundaries

(c) Noisy unoriented data set (d) Surface boundaries

Figure 3.9: Surface boundary detection from noisy data

polarity vectors. Figure 3.10(a) depicts two solid generalized cylinders with different parabolic sweep functions. The cylinder are generated by a uniform random distribution of unoriented points in their interior, while the noise is also uniformly distributed but with a lower density. After sparse voting, volume inliers are detected due to their high ball saliency and low polarity, while volume boundaries are detected due to their high ball saliency and polarity. The polarity vectors are normal to the bounding surface of the cylinders. Using the detected boundaries as input we perform dense voting and extract the bounding surface in continuous form in Fig. 3.10(b).



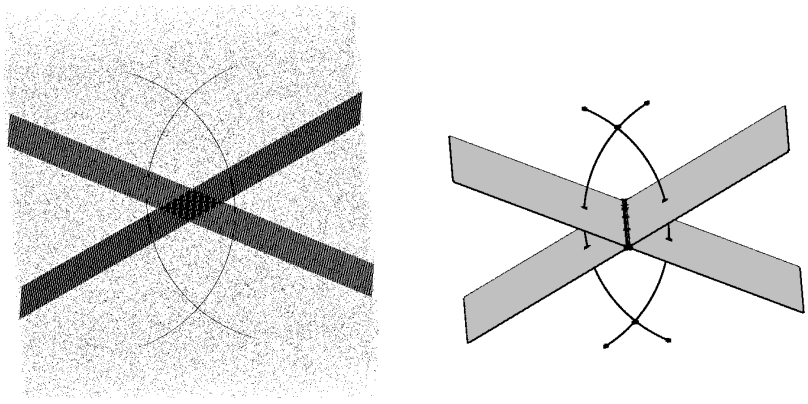
(a) Noisy unoriented data set (b) Dense bounding surface

Figure 3.10: Region inlier and boundary detection in 3-D

We close this section with an example of simultaneous inference of surfaces, curves, surface intersections, junctions, surface boundaries and curve endpoints, which is presented in Fig. 3.11. The simultaneous inference of *all* types of structure and the interaction among them is a feature that can only be found in the tensor voting framework. Methods based on optimization would have to be run once for each structure type while their output at the intersection of different types, such as a curve-surface intersection, would most likely be unclear.

3.5 Discussion

In this chapter, we have presented a critical augmentation to the tensor voting framework. It deals with the fundamental smoothness versus discontinuities dilemma that occurs in most non-trivial perceptual organization scenarios. Many perceptual organization approaches operate either as grouping or as segmentation processes. We believe that both grouping *and* segmentation must be performed in order to tackle challenging problems. In both cases, boundaries play a



(a) Input (b) Output

Figure 3.11: Results on simultaneous inference of multiple types of structures. (a)Unoriented data set that consists of two intersecting planes, two intersecting curves and random outliers. (b) Output after voting. Outliers have been rejected due to very low saliency. Surface inliers are marked in gray, curves and boundaries in black. Curve endpoints and junctions have been enlarged.

critical part. The strictly second order formalism [92] can be viewed as an excitatory process that facilitates grouping of the input data, and is able to extrapolate. On the other hand, the first order representation and voting proposed in this chapter is an inhibitory mechanism that does not allow the inferred structures to grow beyond their boundaries.

It is worth noting that the integration of first order information does not violate the principles on which the original framework is founded. The approach is model free and makes no assumptions about the input other than that salient perceptual structures are generated by the good alignment of tokens. All processing is local within the neighborhood of each token, and local changes of the input result to local changes of the output. Grouping and segmentation are still performed in a soft way, in the sense that the first and second order votes from a token to another express the degree of affinity between the tokens, but there is no hard labeling. Hard labeling can be the final stage of processing, if it is required by a specific application. In addition,

first and second order voting can be performed simultaneously, resulting in a small increase in computational complexity.

Besides the capability to detect terminations in itself, the work presented in this chapter serves as the foundation for more complex perceptual organization problems. In Chapter 5, we propose a novel approach for figure completion in which endpoints and labeled junctions play a critical role. The inference of these keypoints and the classification of junctions as T, L, X etc. is the first step in that direction. This is possible only through the use of the first order augmentation presented in this chapter. Then, the possible modal and amodal completions supported by each type of key point can be examined to produce hypotheses for completion.

Finally, the inferred terminations can serve as interfaces between structures in multi-scale processing scheme. This is especially useful for datasets where data density varies considerably. Even though it can be viewed as a case of amodal figure completion in 2-D, multi-scale processing, using the inferred terminations as inputs to the next scale, can be easily extended to higher dimensions. Encouraging results on medical imaging using this approach can be found in [161].

Chapter 4

Integrated Curve and Keypoint Inference

In this chapter, we address the inference of descriptions from an image in terms of integrated step edges, ridges, endpoints and junctions that can be useful for higher level processes. Standard feature detectors do not produce results of sufficient quality, mainly because edges and junctions are not integrated. In addition, detectors produce clutter, due to spurious responses. On the other hand, contour and surface completion modules generally assume nearly perfect feature detectors. We aim at bringing these ends closer together. We begin by convolving the image with a set of oriented filters and extract extensive information by considering, besides the filter responses, properties such as the direction of contrast and a measure of "stepness". Then, we use tensor voting to infer the most salient features based on their consistency with their neighbors. Fragmented curves are connected and junctions are formed during the final

completion stage. We first evaluate our approach on standardized benchmark datasets, then, proceed to difficult natural images that exhibit phenomena such as occlusion and texture.

4.1 Introduction

The inference of features, such as curves and keypoints, from images of natural scenes is an important component for image understanding. Even though a wide range of detectors have been put to this task, their outputs do not meet the requirements of higher level approaches. These approaches are typically applied to either synthetic data or rather simple real images. The gap between low and high-level modules is due to the fact that noise-free, non-fragmented edges with well localized junctions are necessary when one addresses more complex phenomena, such as modal and amodal surface completion, occlusion detection and relative depth ordering. Feature detectors, on the other hand, often fail to extract important features of the image or produce spurious responses due to intensity configurations that do not necessarily match the desired template. Meer and Georgescu [96] analyze this behavior in more detail.

In this chapter, we attempt to bridge the gap between low level feature detection and higher level scene interpretation by detecting salient features in such a way that they can be useful for subsequent processing modules. What makes these features salient is not the magnitude of filter responses, but rather their alignment with their neighbors that gives rise to the perception of connected contours and prominent junctions where the contours intersect. Such contours are usually associated with objects in the scene, while junctions are local indicators of potential occlusion. We do not address object segmentation or long-range completion here (for the latter see Chapter 5), but we show that our framework is effective in laying the groundwork for

them, even in difficult natural images. We show results on the benchmark datasets for contour completion proposed in [171], where we achieve good performance, as well as in real images. According to the principles set out in the introduction, we adopt a least commitment strategy and postpone hard decisions as long as possible.

The remainder of the chapter is organized as follows: related work is presented in the next section, an overview of the approach is presented in Section 4.3, feature detection in Section 4.4 and how tensor voting is applied for the inference of contours from edgels in Section 4.5. Integrated curve and keypoint inference is described in Section 4.6, experimental results in Section 4.7, and a summary and perspectives for future research in Section 4.8.

4.2 Related Work

In this section, we review research related to the main aspects of the chapter. Issues that are of particular interest to us are the integrated detection of edges and junctions (where the term is used to denote junctions of cardinality greater than two and not just corners), the capability to complete both ridge contours and surfaces, and applicability to real images.

We begin by reviewing research on feature detectors. We ignore isolated points as features and focus on regions, which can be represented by their bounding step edges, and ridges, which are 1-D manifolds with ridge-like intensity profiles. What should not be ignored are the junctions that are formed when regions overlap, and intersections of ridges. These junctions are not just locations of 2-D intensity variation, but they necessarily have to be associated with step edges or ridges, and cannot exist in isolation. Based on this observation, we do not consider edge and junction detectors that operate independently. Förstner [30] proposed an integrated

representation that captures 1- and 2-D image variations by local aggregation of intensity gradients. The degree of orientation or isotropy of each pixel can be estimated from the eigenvalues of a tensor. While the representation is integrated, it fails for ridges and suffers from blurring. These problems are not present in the subsequent work of Granlund and Knutsson [38], who derive tensors from the responses of a family of oriented quadrature filters. Along the same lines, Köthe [71] makes the observation that the integration of simple detectors for each type of feature is an almost impossible task. It is very difficult to adjust the weights appropriately, as well as to suppress the spurious responses of each detector at the features it has not been designed for. He proposes a filter bank that consists of polar harmonic separable filters and detects edges, corners and junctions in a unified way. The response at each image location is a "boundary tensor" that can simultaneously represent step and roof edges and junctions of different degrees, such as L (degree 2), T and Y (degree 3) and X and Ψ (degree 4). The boundary tensor meets our requirements as a representation since it simultaneously encodes all possible feature types without making premature hard decisions. It is not non-negative definite, as our tensors, but it could provide a good means of initializing the tokens. In fact, we adopt a very similar representation in Section 4.4.

Using the filter responses as inputs, much work has been done towards saliency estimation from edgels. Approaches, such as [115, 144, 170], which were presented in Section 2.1, attempt to measure edge saliency based on criteria such as good continuation. The drawback of these methods is that they do not consider junctions. In [42] Guy and Medioni address the simultaneous inference of curves and junctions based on the accumulation of local support. They employ

a voting scheme which is very similar to the one presented here, but their data representation is less effective than ours.

A similar class of methods that aim at the detection of closed contours, which signify scene objects include [26, 88, 166, 171]. In [26] the authors propose a Bayesian approach that infers closed contours as shortest-path cycles in a graph. Williams and Thornber [171] propose to use the probability that an edge is included in a closed contour as a measure of the saliency of a closed contour. They also provide a benchmark dataset and evaluate various grouping methods on the inference of closed contours. The inputs to both of these approaches are noise-free edge segments. Mahamud *et al.* [88] take as input the responses of an edge detector, and infer multiple salient closed contours from real images. Edgels are partitioned into sets of isolated strongly-connected components of the edge graph, which correspond to smooth salient contours. Wang *et al.* [166] find closed contours in real images by introducing a saliency measure that is normalized over the boundary length. The applicability of all the methods described in this paragraph is limited by the fact that they cannot handle junctions and thus cannot deal with overlapping objects. It should also be noted, that while closed contours are salient, open contours can also be very salient and thus closure should not be used as a necessary condition for saliency. Ren and Malik [119] presented an approach to contour completion that utilizes manually segmented images to derive a prior model for contours that are perceived as salient by human observers. Contours are represented as high-order Markov chains, since a first order model proved to be insufficient. An important contribution is a measure of local "textureness" that enables them to handle images with texture, which caused difficulties to

previous approaches. The authors do not explicitly require closure, but they also do not consider junctions.

A different approach is to address the problem from a region or object point of view. Mohan and Nevatia [100] aim at inferring curvilinear structures, while the former aims at segmentation and extraction of 3-D scene descriptions from collations of features that have high likelihood of being projections of scene objects. They operate in a hierarchical bottom-up fashion starting from edgels and increasing the level of abstraction at each iteration. In addition to local criteria, such as proximity and good continuation, the authors consider parallelism and symmetry as cues for grouping. Along the same lines is Jacobs' [56] technique for inferring salient convex groups among clutter since they most likely correspond to world objects. The criteria to determine the non-accidentalness of the potential structures are convexity, proximity and contrast of the edgels. Saund [135] proposed an approach for the completion of contours that bound opaque surfaces that examines the degree of genericness of properties of edges and junctions. Rules founded on ecological optics, which describe possible configurations, and ecological statistics, which prefer likely configurations, are used to achieve perceptual organization. These approaches are sensitive to the quality of the inputs, in terms of clutter and localization and orientation errors. In most cases, results are presented for images where feature detectors perform adequately.

It should be noted here that contour based systems prefer good continuation while surface based systems prefer convexity or closure as the dominant criterion for grouping. An approach that considers both was presented by Saund in [136]. The objective is to find nearly closed paths in sketches and line drawings in the presence of gaps and clutter. Both smooth continuation

paths and maximally turning paths are detected since both types can correspond to significant scene objects. Since the input consists by definition of ridges, contrast, which could discard some hypotheses, is not available. This work is significant since it provides a complete set of rules for perceptual organization that deal with the problem of competing criteria.

Complete systems that aim at completion include the work of Heitger and von der Heydt [47] who use quadrature filters to compute oriented energy of step edges and ridges and end-stopped operators that detect endpoints, as well as junctions. At later stages of processing, special fields that enable parallel and orthogonal grouping of contours are used to facilitate modal and amodal completion of surfaces. Their approach takes into account step edges, ridges, endpoints, junctions and contrast and is capable of producing interpretations of difficult natural images. In [97] Mingolla *et al.* combine the Boundary Contour System (BCS) and the Feature Contour System (FCS) that were developed separately to address contour and feature completion, respectively. The combined system is applied at multiple scales for the enhancement of images of very high dynamic range, achieving good results, even though it does not model junctions.

Finally, Massad *et al.* [91] were the first to apply tensor voting directly to gray-scale values rather than symbolic tokens such as edgels. The tensor at each pixel was initialized by the tensor addition of the responses of Gabor filters. An inhibitory voting mechanism was introduced in order to suppress the effect of the inevitably large filter responses close to the actual edge locations. Even though the inhibitory voting scheme was not very effective, the strength of tensor voting in inferring salient contours due to proximity and good continuation of correctly detected edgels, while suppressing isolated edgels, was demonstrated in real images.

Before proceeding to the description of the algorithm, two important aspects of our work need to be highlighted. First, junctions cannot be reliably detected at the pixel level since they occur because of the intersection of salient contours. Therefore, information must be aggregated in larger regions before a pixel can be reliably labeled as junction. Second, even though we do not make object segmentation decisions or attempt long-range completion, we still want to ensure that our outputs are general enough and suitable for these tasks. This is one of our main considerations when developing the algorithm presented in this chapter.

4.3 Algorithm Overview

The inputs to the algorithm are gray-scale images and the output is a description in terms of contours and keypoints. Processing entails three major steps:

- local feature detection,
- tensor voting for saliency estimation
- and short-range, integrated curve and junction completion.

These steps are described in the following sections.

Definition of terms To avoid confusion, we define the terms as they are used in the remainder of the chapter:

- **step edges** are region boundaries that have different intensities on each side and evoke large responses from *odd filters*.

- **ridges**, or roof edges, are curves in line drawings or textures. Image intensity is the same on both sides of ridges, and thus *even filters* are more sensitive to them.

- **junctions** are locations where surfaces overlap. L-junctions are corners of surface boundaries, while T-junctions typically occur at occlusion boundaries. Both types can also appear in areas with texture or intensity changes within a surface. Other types are possible, but less frequent since they result from non-generic configurations or transparency.

- **ridge intersections** are locations where multiple ridges intersect or points of high curvature of a single ridge. The most common types are L, T, X, Y and W intersections. Note that, unlike step edges that are associated with regions, which are necessarily closed, nearby ridges do not necessarily extend to form intersections.

- **curves** are connected chains of step edges or ridges. They start and end at an endpoint, junction, or ridge intersection. Using this definition the direction of contrast of a curve that consists of step edges remains constant. A junction has to be introduced for the contrast direction to change.

- **keypoints** according to [47] include endpoints, junctions and ridge intersections. Their cardinality is the number of curves that are associated with them. It is one for endpoints, two for L-junctions or L-intersections, and so on.

4.4 Local Feature Detection

At the first stage of the algorithm we detect 1- and 2-D intensity features by convolving the image with the second derivatives of an anisotropic Gaussian and its Hilbert transform, as in [119], to measure edge strength in eight orientations. The oriented energy at θ is then:

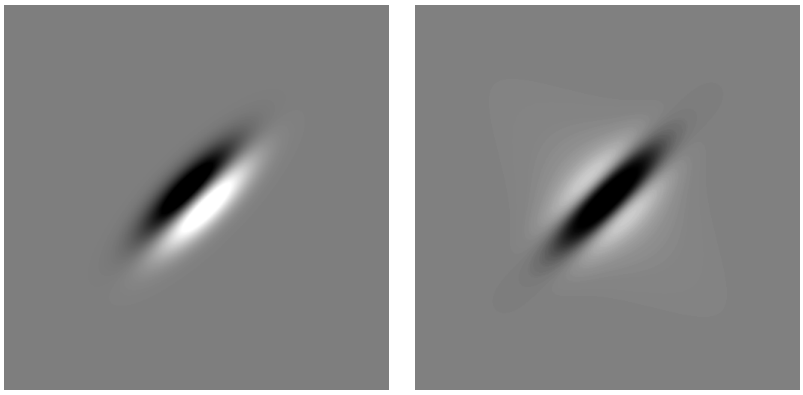
$$E_{\theta}^{2} = (I * G''_{\sigma_x,\sigma_y,\theta})^2 + (I * H_{\sigma_x,\sigma_y,\theta})^2 \tag{4.1}$$

where $G''_{\sigma_x,\sigma_y,\theta}$ is the second derivative of a Gaussian and $H_{\sigma_x,\sigma_y,\theta}$ is its Hilbert transform. These kernels are oriented at θ, and σ_y and σ_x are the standard deviations along and orthogonal to this direction, respectively. The shape of the odd and even filter, rotated at 45^o degrees and with an aspect ratio ($\frac{\sigma_x}{\sigma_y}$) of 3, can be in seen in Fig. 4.1. Using this measure of energy, local tensors can be computed via tensor addition:

$$\mathbf{t}(\theta) = \begin{bmatrix} cos(\theta) \\ sin(\theta) \end{bmatrix} \begin{bmatrix} cos(\theta) & sin(\theta) \end{bmatrix} \tag{4.2}$$

$$\mathbf{T}(x,y) = \sum_{i=0}^{N-1} E_{\theta}(x,y)\mathbf{t}(\frac{i\pi}{N}) \tag{4.3}$$

While tensors computed in this way capture the local structure, they do not indicate whether the underlying image features are step edges or ridges. This can be determined from the individual responses of the image to an anisotropic derivative of a Gaussian, which responds well to step edges, and its Hilbert transform which responds well to ridges. In particular we sum



(a) Odd filter (b) Even Filter

Figure 4.1: Second derivative of Gaussian filters. The aspect ratio is $\frac{\sigma_x}{\sigma_y} = 3$.

the magnitude of the responses in each orientation of the Hilbert pair separately. Our *stepness* measure is the ratio of the step response to the total response adjusted to be between -1 and 1, for perfect ridges and perfect step edges respectively.

For step edges, we also need the direction of contrast, i.e. a vector pointing from the dark side of a step edge to the bright side. This direction is parallel to the first eigenvector of the local tensor. However, in tensor voting, tensors represent orientation but not direction, and thus two opposite directions are indistinguishable in tensor form. In addition, the tensors themselves change considerably after voting. Therefore, we associate with each pixel an additional vector that indicates the direction of contrast, which remains unchanged throughout all the processing steps.

The output of the feature detector for each pixel, therefore, consists of:

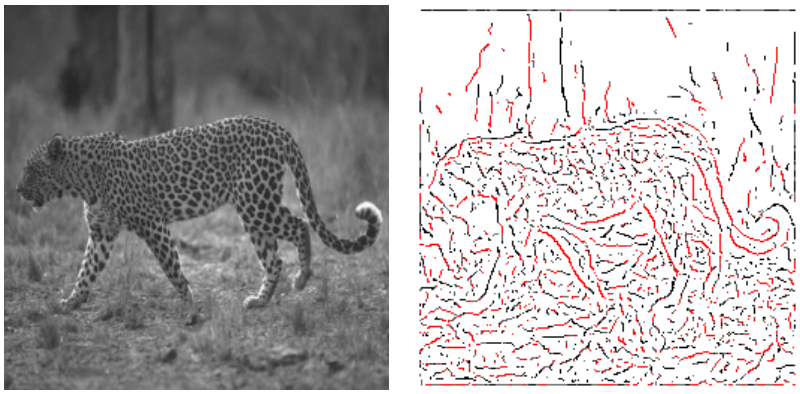
- a second order tensor, similar to the one in [71], whose eigenvalues (λ_1, λ_2) and eigenvectors $(\hat{e}_1, \hat{e}_2)$ describe the local structure,

- the stepness ratio
- and the direction of contrast.

4.4.1 Non-maximum Suppression

Since the filters are of a certain size, they suffer from localization uncertainty, and respond even when they are a few pixels away from a feature. These erroneous responses are weaker than the one at the exact feature location, but they may be stronger than the response at features of lower contrast. Their influence is detrimental to the following processing steps since they cause blurring. To remove them, in line with most edge detection schemes, we perform non-maximum suppression with respect to the edge strength ($\lambda_1 - \lambda_2$) of each pixel. We found that performing non-maximum suppression before tensor voting is a more effective approach than the one of [91], since it removes erroneous responses before they cause blurring.

Initially, all pixels are marked as potential edges. Then, the edge strength of the two neighboring pixels as indicated by the normal to the edge orientation of the current pixel is examined. If either is larger than that of the current pixel, the latter is unmarked as a potential edge. This 3×1 neighborhood proves to be sufficient in rejecting non-maximum responses. Initial keypoint candidates are also identified after non-maximum suppression with respect to λ_2 in 9×9 windows. Results on the "leopard" image, from the Berkeley Segmentation Dataset (http://www.cs.berkeley.edu/projects/ vision/grouping/), can be seen in Fig. 4.2, where keypoint candidates are omitted for simplicity.



(a) Input image

(b) Results of non-max suppression

Figure 4.2: Non maximum suppression for edge responses. Red lines are step edges, black lines are ridges.

4.5 Saliency Estimation via Tensor Voting

The most important part of the description produced by the local detectors is the tensor, which is used to initialize the tensor voting stage. The tensors of the previous stage are in a form suitable for tensor voting since they represent orientation information and uncertainty (isotropy) in the same way as in token-based voting. The chosen representation is a natural fit with the framework which, furthermore, allows us to initialize with tensors that are neither pure balls or pure sticks, but simultaneously encode both the oriented and unoriented properties of the pixels. The difference between [71] and our approach is that we use λ_2 as the strength of the unoriented response instead of $2\lambda_2$. In our notation, λ_1, and not the trace of the tensor, is the total energy of a pixel.

Tensor voting takes place in two stages. At the first stage, *sparse voting* is performed and only pixels that are marked as potential edges or keypoints after non-maximum suppression cast

and collect votes. This step refines the estimated orientations, reinforces the salient tokens and makes outliers easier to detect since they receive little support from their neighbors, and thus do not accumulate high saliency values. At the second stage, *dense voting* is performed and all pixels from the previous stage cast votes to all their neighbors. Pixels that were suppressed are represented by null tensors and only collect votes. During this stage information is propagated over the entire image and saliencies are computed for every pixel. Iterations of tensor voting, beyond the sparse and dense steps described here, are in general detrimental since they are effectively equivalent to increasing the scale of voting (since information propagation occurs over much larger ranges), which results in the over-smoothing of corners and junctions.

4.5.1 Quantitative Evaluation of Local Saliency Estimates

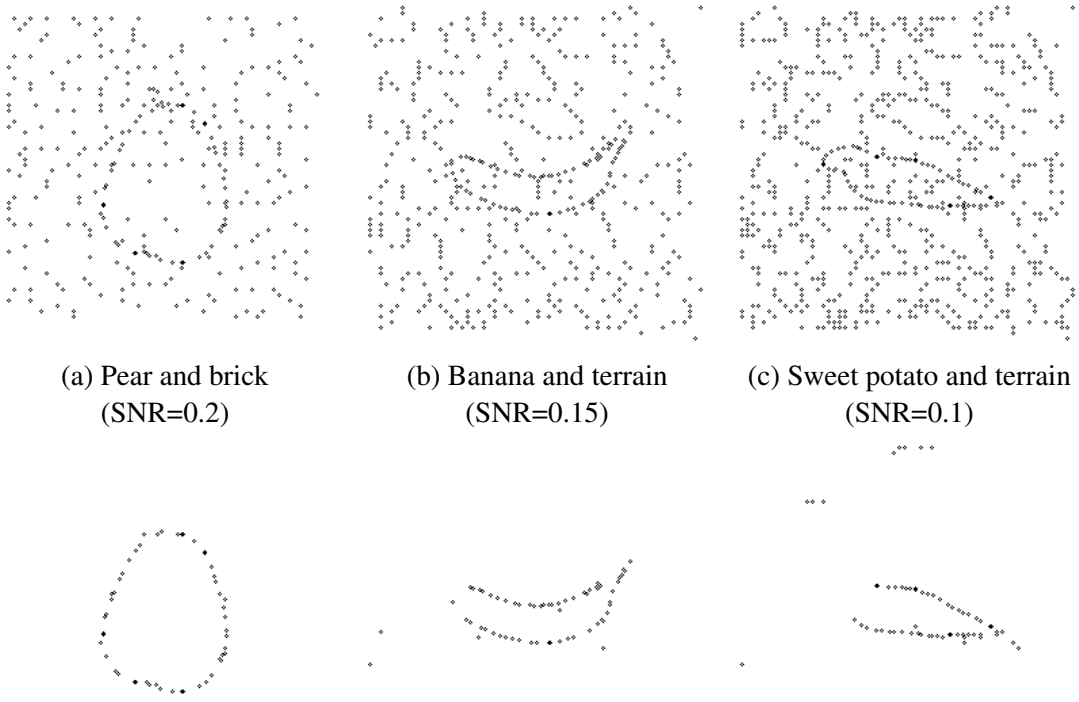
To evaluate the effectiveness of tensor voting in estimating the saliency of each input, we tested it with the datasets proposed in [171]. Each dataset consists of a foreground object represented by a sparse set of edgels super-imposed on a background texture, which is also represented as a sparse set of edgels. There are a total of nine foreground objects (fruit and vegetable contours), which are uniformly rescaled to fit 32×32 bounding boxes, and nine background textures which are rescaled to 64×64. The nine fruits and vegetables are: avocado, banana, lemon, peach, pear, red onion, sweet potato, tamarillo and yellow apple. The textures are taken from the MIT Media Lab texture database and are: bark, brick, fabric, leaves, sand, stone, terrain, water and wood.

The goal is to infer the edgels of the foreground object, which align to form the largest salient contour. The background edgels come from an image of texture and are, therefore, less

structured and do not produce alignments more salient than the foreground. The desired output is the N_f most salient edgels, where N_f is the number of edgels of the foreground. If they all belong to the foreground then performance is considered perfect. The reported error rates are the percentage of background edgels included in the N_f most salient. The difficulty comes from increasing the number of background edgels in each experiment. The SNR is defined as the ratio of foreground to background edgels in each dataset. Five SNRs ranging from 25% to 5% are used for each of the 81 combinations. All edgels are encoded as stick tensors of unit strength oriented at the given angles. After sparse voting, the given orientations are corrected and a second round of voting is performed. Since the error metric is based on the input positions, we only consider input locations in the second pass of voting. Figure 4.3 contains a few input and output pairs.

Figure 4.4 shows the performance obtained by the methods tested in [171] for each SNR level. The figure was manually replicated from the original paper. The methods tested and the corresponding initials in Fig. 4.4 are:

- Shashua and Ullman [144] (SU),
- Herault and Horaud [48] (HH),
- Sarkar and Boyer [131] (SB),
- Guy and Medioni [42] (GM),
- Williams and Jacobs [170] (WJ)
- and the one proposed in the paper by Williams and Thornber [171] (WT),



(a) Pear and brick (SNR=0.2)

(b) Banana and terrain (SNR=0.15)

(c) Sweet potato and terrain (SNR=0.1)

(d) Output (FPR=5.88%)

(e) Output (FPR=16.39%)

(f) Output (FPR=24.19%)

Figure 4.3: Most salient inputs and false positive rates on typical examples from [171] at various SNRs

Our algorithm achieves the following false positive rates (for $\sigma = 40$):

SNR	FPR
25	10.04%
20	12.36%
15	18.39%
10	35.81%
5	64.28%

As shown in Fig. 4.4, it outperforms all the methods in [171], even though we do not consider closure, which plays a significant role in this experiment. A subset of these datasets

was used to evaluate the algorithm of [166], but no numerical results are reported. Visually, their results, in the form of the most salient closed contour, are correct for high SNR values. The results we obtain are encouraging in our attempt to tackle real images, even though phenomena such as junctions and occlusion have to be ignored, since the fruit appear transparent when encoded as sets edgels from their outlines in the input.

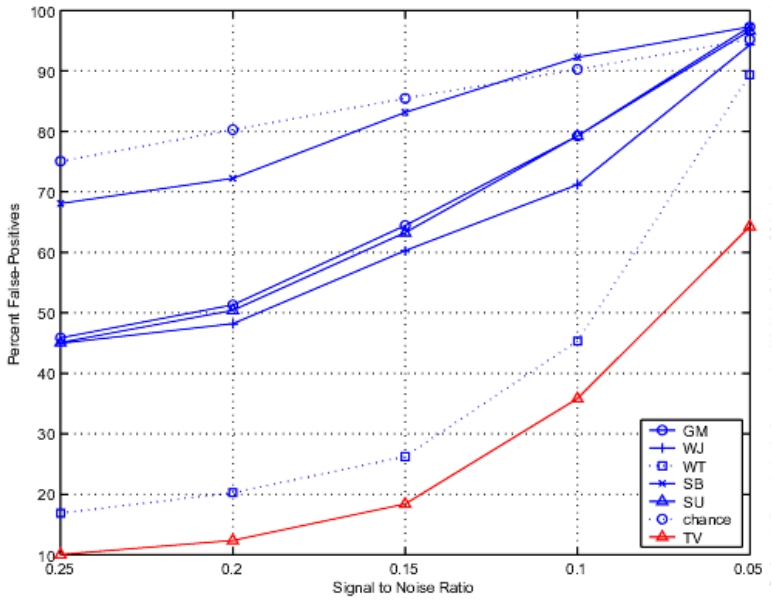


Figure 4.4: False positive rate at each SNR level. Our results outperform those of Williams and Thornber and the other methods described in [171].

4.6 Integrated Curve and Keypoint Inference

In this section, we describe how curves and junctions can be inferred based on the tensors derived after tensor voting and the additional information provided by the feature detectors, that is the direction of contrast and the stepness ratio. An important aspect of curve and keypoint

inference is the interaction between them. Keypoints are located where special events related to curves occur. These include:

- *termination*, which must be associated with an endpoint,
- *sharp change of orientation*, which is indicated by an L-junction or an ridge corner, an
- *intersection*, which produces T-junctions and ridge intersections.

At the same time, curves are attracted to keypoints, especially in the case of step edges that are region boundaries and have to be closed.

The first step of this stage is the inference of connected curves. It is accomplished by selecting pixels with maximum stick saliency, which have not been visited, as seeds and marching following the path of locally maximum stick saliency. For each new seed, marching begins in one of the two possible directions, while the other direction is examined after the first one stops. The marching direction is maintained for each branch by ensuring it is consistent with the incoming direction. The new pixel to be added to the curve can be one of three candidates among the 8-neighbors of the current pixel. The three candidate continuations are the pixel pointed to by the estimated tangent of the current pixel and its two neighbors on the perimeter of a 3×3 window centered at the current pixel (Fig. 4.5).

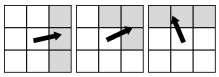


Figure 4.5: Possible continuations during marching at the direction indicated by the tangent of the current pixel

For a pixel to be added to the curve, it must have:

- the maximum stick saliency among the three candidates,
- the same sign of stepness ratio as the current pixel
- and a positive inner product between its contrast direction with the contrast direction of the current pixel.

Marching stops at potential junctions and endpoints, marked at the non-maximum suppression stage, or when the ball saliency of the candidate to be added is larger than its stick saliency. At this stage, we do not connect step edges and ridges, even though this is possible for certain image configurations where region boundaries have low contrast. We also do not allow contrast direction reversals, which cannot occur without the presence of a junction. These cases are handled at the next step when keypoints are considered. Curves with fewer than 10 pixels are considered outliers and the pixels belonging to them are labeled as non-features.

A methodology for reasoning about contour completion from simple curves has been proposed by Saund in [136]. Our approach is somewhat simpler since we do not aim at inferring potential object outlines. A key concept is that good continuation should be preferred when attempting to complete ridges (as in [42, 88, 170, 171]), but the formation of L-junctions should typically occur first when completing step edges. The latter can be interpreted as local evidence for closure and convexity, which are likely properties of surfaces.

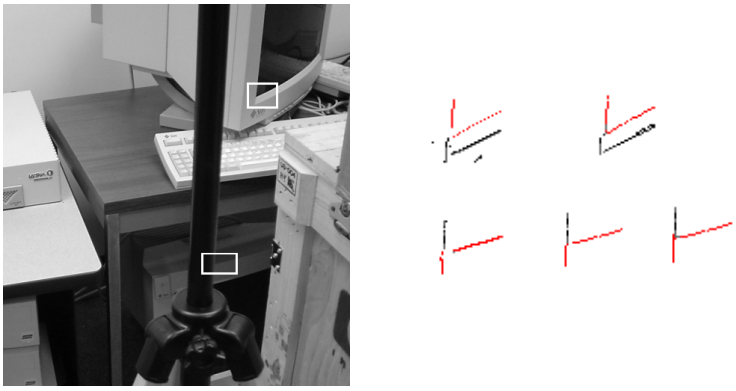
(a) Legal L-junctions (b) Illegal T-junction

Figure 4.6: Legal L-junction and illegal T-junction configurations according to contrast direction. The arrow at each edge points to the bright side. Symmetric cases have been omitted.

The curve marching stage has produced a set of simple curves and keypoints most of which are endpoints since the detector responses tend to attenuate as we approach keypoints of cardinality two or greater. At this stage we aim at connecting fragmented smooth contours and forming L-junctions. We begin by examining endpoints for possible completions in a small window that progressively grows, typically in steps of 5 pixels. If only one possible completion is found, two cases are possible. In case of a fragmented contour, a smooth connection is made following maximum stick saliency and the two curves are merged, as long as there is no contrast direction reversal. In case of an L-junction, the junction is formed at the local maximum of ball saliency, if it is legal with respect to contrast directions (see [135] and Fig. 4.6(a)). If multiple alternatives exist, the choice is made according to the following criteria that are evaluated in order:

- good continuation is preferred in case of ridges, while the formation of an L-junction is preferred in case of step edges, provided that they are legal,
- the nearest connection of the preferred type is chosen,
- or, in case of multiple alternatives at the same distance, the angle closest to 0^o for fragment completion, or the one closest to 90^o for L-junctions is preferred.

These process is iterated typically three times in progressively increasing window sizes to allow the formation of viable alternatives that were not initially selected. Finally, junctions of cardinality greater than two are formed by connecting compatible endpoints to existing junctions. The illegal T-junction configuration shown in Fig. 4.6(b) is not allowed. Only the presence of at least one more curve and the formation of a junction of higher cardinality would make this



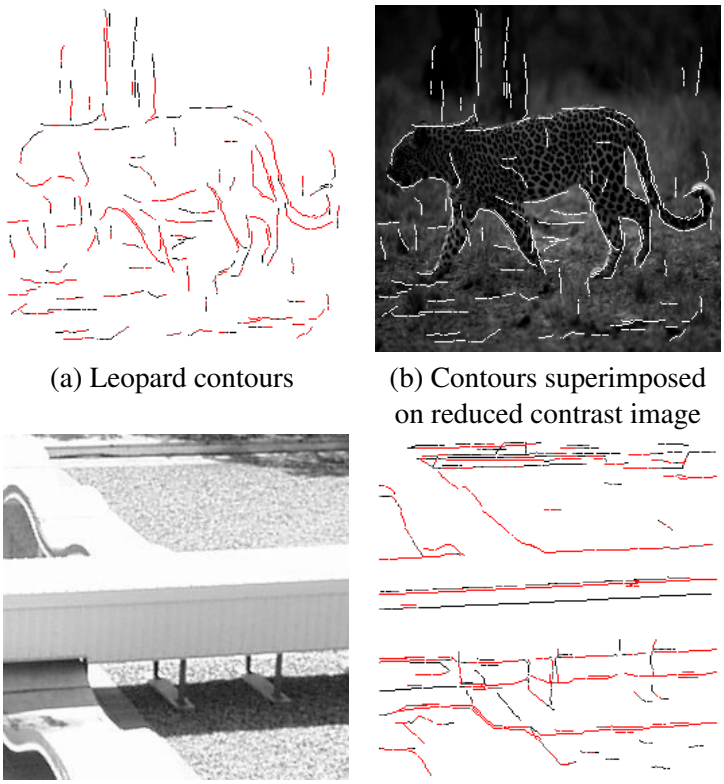
(a) Input (b) Formation of junctions

Figure 4.7: Junction completion in a real image. The inputs and the completions of an L and a T-junction in the highlighted areas are shown in (b).

configuration valid. We have found that the size of the window and the number of iterations are not critical as long as they are kept small. The completion of an L and a T-junction in the highlighted parts of the input image can be seen in Fig. 4.7.

4.7 Experimental Results

Results on contour completion from point data, where the feature detectors did not have to be used, were shown in Section 4.5.1. Here we present results on images of natural scenes. They are very challenging because they contain very large numbers of texture edges and illumination edges that produce strong filter responses even though they would not be marked as salient edges by a human observer. Even though our results are far from being considered entirely satisfactory, they show the potential of the proposed approach to infer features that agree with human perception of saliency. We are able, for instance, to ignore texture edges and focus on surface outlines.



(a) Leopard contours

(b) Contours superimposed on reduced contrast image

(c) Roof

(d) Contours

Figure 4.8: Results on real images containing texture. Red (gray) lines denote step edges and black lines denote ridges.

Figure 4.8 shows results on natural images with texture. The "roof" image was taken from the internet, while the "leopard" from the Berkeley Segmentation Dataset (see also Fig. 4.2). An example of an image we took can be seen in Fig. 4.9. Note the detected ridges on the side of the stapler and the completion of the striped pattern that is also shown on the right. Illumination effects, which we do not account for, cause some illumination edges to be detected along with the real edges (an effect also observed in [135]). What is important, however, is that the localization of critical keypoints, such as the T-junctions where the block occludes the stapler, is still acceptable.

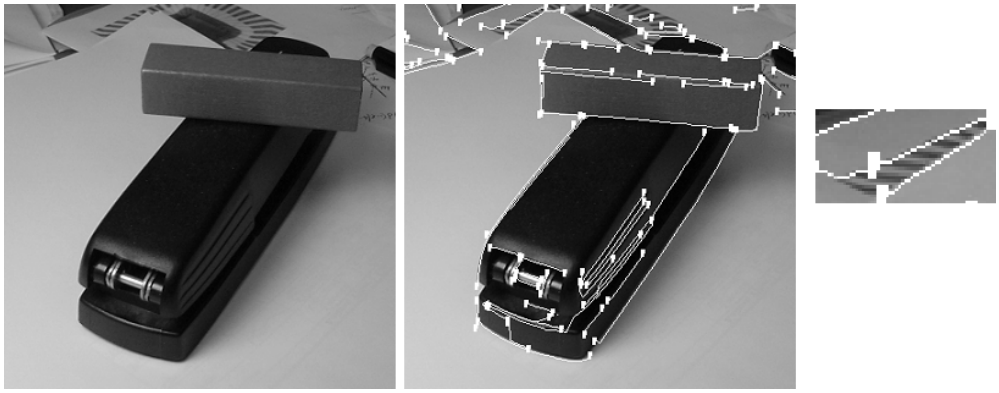


Figure 4.9: Results in a real image (keypoints are displayed as squares). Note the completion (at the top left) that is magnified on the right.

4.8 Discussion

A major contribution of this chapter is the application of tensor voting to real images. With the exception of [91], it had not been applied directly to images before. Nevertheless, information propagation via tensor voting proves to be effective in detecting salient features based on the Gestalt principles of proximity and good continuation. Quantitative evaluation of our results for the data of [171], in Section 4.5.1, proves the power of our approach in feature saliency estimation.

Furthermore, the fact that the saliency of a feature is evaluated after local support has been aggregated allows us to process images with stochastic texture. Local detectors respond strongly in textured areas, but candidate curves and junctions in these areas do not form well-aligned configurations, and therefore do not amass enough support to be considered salient. Results in images with texture have only been presented in [119], where texture is treated explicitly, but junctions are not detected.

Our framework also allows the integrated representation of curves and keypoints via tensors, as well as interactions between them via voting. This is of great importance for junction and intersection inference which is accomplished by the accumulation of votes that convey multiple orientations, rather than by examining local filter responses to intensity patterns. The inference of keypoints makes our results more suitable as inputs to high level processes, such as [135, 136], which aim at surface completion and occlusion analysis. Junctions play a key role in the generation of hypotheses for these tasks as local cues for occlusion or fragmentation. In the following chapter we address figure completion, the cues for which are features such as endpoints and T-junctions that have to be inferred by a process such as the one described here.

Chapter 5

Figure Completion

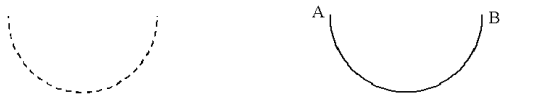
In this chapter, we address the issues associated with figure completion, a fundamental perceptual grouping task. This part of our research was published in [104]. Endpoints and junctions play a critical role in contour completion by the human visual system and should be an integral part of a computational process that attempts to emulate human perception. A significant body of evidence in the psychology literature points to two types of completion, modal (or orthogonal) and amodal (or parallel). We present a methodology within the tensor voting framework which implements both types of completion and integrates a fully automatic decision making mechanism for selecting between them. It proceeds directly from tokens or binary image input, infers descriptions in terms of overlapping layers and labels junctions as T, L and endpoints. It is based on first and second order tensor voting, which facilitate the propagation of local support among tokens. The addition of first order information to the original framework is crucial, since it makes the inference of endpoints and the labeling of junctions possible. We illustrate

the approach on several classical inputs, producing interpretations consistent with those of the human visual system.

5.1 Introduction

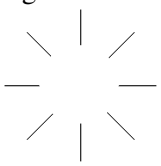
Figure completion is an important component of image understanding that has received a lot of attention from both the computational vision and the neuroscience community over the last few decades. While we do not claim that our approach is biologically plausible, the human visual system serves as the paradigm, since even a small fraction of its performance is still evading all attempts to emulate it. In this chapter, we show that the interpretations we infer are consistent with human perception even in difficult cases. Consider the fragmented contour of Fig. 5.1(a) which supports amodal completion of the half circle, as in Fig. 5.1(b). Note that completion stops at the two endpoints, marked A and B. Now consider Fig. 5.1(c), which is known as the Ehrenstein stimulus. This supports modal completion and produces a strong and unique perception of an illusory circle (Fig. 5.1(c)). Note that the outer circle is not perceived, most probably because it would require a concave occluder which is unlikely and thus not preferred by the human visual system. What makes this input interesting to us is that both types of completion are supported by the data: modal completion of the occluding disk and amodal completion of the occluded lines. We aim at inferring a description which is consistent with human perception that produces a layered interpretation of the image, with a white disk occluding a set of black lines on a white background (Fig. 5.1(e)).

An aspect of computer vision that has not been solved is the selection between modal and amodal completion. Most researchers assume that the type of completion to be performed is

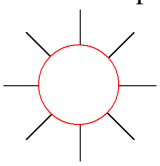


(a) Fragmented contour

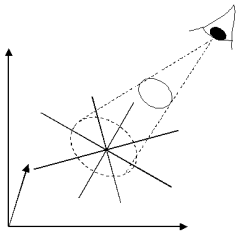
(b) Amodal completion



(c) The Ehrenstein stimulus



(d) Modal completion



(e) Layered interpretation of the Ehrenstein stimulus

Figure 5.1: Amodal and modal completions

known in advance. For instance, modal completion in most cases starts only when the endpoints and the orientation of completion are provided as inputs. We aim at inferring the endpoints and junctions from unoriented data, and then, automatically making decisions whether further completion is supported by the data, and which type of completion should occur. The tensor voting framework is well-suited for this task since it allows the integrated inference of curves, junctions, regions and terminations. The latter is possible after the incorporation of first order information into the framework.

This chapter is organized as follows. Related work is presented in the next section, an overview of the approach is presented in Section 5.3, aspects of tensor voting directly related with the problem at hand in Section 5.4, figure completion in Section 5.5 and experimental

results are shown in Section 5.6. A summary of our preliminary results and perspectives of our future work are given in Section 5.7.

5.2 Related Work

Work related to this chapter appears in Sections 2.1 and 4.2. Here we only present research on figure completion. Classical approaches on the inference of salient contours include the work of Grossberg, Mingolla and Todorovic [39, 40] who developed the *Boundary Contour System* and the *Feature Contour System* that can group fragmented and even illusory edges to form closed boundaries and regions by feature cooperation and competition in a neural network. Gove *et al.* [37] present a paper within the BCS/FCS framework that specifically addresses the perception of illusory contours. In the work of Heitger and von der Heydt [47] elementary curves are grouped into contours via convolution with a set of orientation selective kernels, whose responses decay with distance and difference in orientation. Mechanisms for both parallel and orthogonal grouping based on keypoints such as junctions and line ends are also proposed. Williams and Thornber [172] extend the *stochastic completion fields* framework of Williams and Jacobs [170] to address modal and amodal completion. They describe the inference of closed illusory contours from position and orientation constraints that can be derived from line terminations.

Our work differs from other approaches in that we can proceed from unoriented, unlabeled data and simultaneously infer curves, junctions, regions and boundaries. Moreover, we propose an **automatic** mechanism for making decisions between modal and amodal completion without having to know the type of completion in advance.

5.3 Overview of the Approach

The input to our algorithm is a set of tokens in a two-dimensional space. The tokens indicate the presence of a primitive that potentially belongs to a larger configuration and can be generated by a process such as the one of the previous chapter. The output is a layered description of the image in terms of salient curves, junctions and regions. Our first goal is to detect salient groupings based on the support tokens receive from their neighbors. The amount of support from one token to another is in the form of a first and a second order vote whose properties depend on proximity, colinearity and cocurvilinearity. Since the representation of each token can simultaneously encode its behavior as a curvel, a junction, or a region inlier, tokens do not have to be classified prematurely, and decisions are made when enough information is available. Here, we take advantage of first order properties to infer endpoints and region boundaries, as well as label junctions.

The novelty of our work is the mechanism for modal and amodal completion using the endpoints and junctions inferred at the previous stage. Endpoints, T-junctions and L-junctions offer two possibilities for completion. Either completion along the endpoint's tangent or the T-junction's stem if a corresponding keypoint with compatible orientation exists within a certain vicinity (amodal completion), or completion along the orthogonal orientation of an endpoint, the bar of a T-junction or an edge of an L-junction (modal completion). To facilitate modal and amodal completion, two new voting fields are defined based on the original voting field. The decision whether completion is supported by the data is made by examining the support for both options at every keypoint. If at least one more keypoint supports modal or a modal completion, the current keypoint is labeled as one that supports completion, and the appropriate voting field

is used. If both modal and amodal completion are supported, both are further pursued and a layered description is inferred. The process and the fields are described in more detail in Section 5.5.

5.4 Tensor Voting on Low Level Inputs

The input tokens can be generated as in the previous chapter. Since natural images do not typically exhibit modal completion, we use synthetic data to demonstrate our approach. Edge and junction detection are considerably easier for these synthetic images, but the focus here is on the higher level processing stage, where completion occurs. In the examples presented here, black pixels are encoded as unit ball tensors. Since we want to collect saliency information everywhere, the remaining grid positions are initialized with null tensors. Computational complexity is not increased, since the null tensors do not cast votes and the increase in the number of receivers is not dramatic. Dense saliency maps allow us to perform non-maximal suppression for the inference of salient structures, as described in the previous chapter.

Besides curves and keypoints, regions are also inferred based on their high λ_2 and the fact that they are enclosed by region boundaries. The latter can be inferred after non-maximum suppression with respect to polarity along the direction of the polarity vector. If regions are inferred in the data, an additional step is required. The boundaries of the inferred regions, which are detected based on their high polarity and ball saliency, participate in a round of tensor voting along with the curvels and junctions to infer region-curve intersections. The set of endpoints and junctions that is passed on to the next module is determined after analyzing the saliency maps after this additional stage.

5.5 Completion

Now that endpoints and junctions have been inferred and labeled, we need to consider how they interact to produce figure completion. There are two types of completion: modal and amodal. In *amodal* completion, endpoints extend along their tangent and T-junctions along their stem. This case corresponds to the completion of an *occluded* surface behind the occluder. In *modal* completion connections occur along the bar of T-junctions; orthogonally to the tangent of endpoints, which are interpreted as low-contrast T-junctions; or along one of the edges of L-junctions, which are also interpreted as low-contrast T-junctions. Modal completion is the completion of the *occluding* surface on top of the occluded ones. Amodal and modal completion are termed parallel and orthogonal grouping by Heitger and von der Heydt [47].

Our framework makes automatic decisions on which type of completion occurs. Either type has to be supported by at least one other key point within a certain range, which is larger than that of the first stage. For instance, amodal completion from an endpoint has to be supported by another endpoint with similar orientation and opposite polarity. Then, tensor voting is performed to infer a contour connecting the two endpoints, as with the endpoints of the curve segments of Fig. 5.1(a). Amodal completion between pairs of endpoints produces the contour of Fig. 5.1(b).

New voting fields, based on the original second order stick voting field, have to be defined to facilitate the propagation of information from the voting endpoints and junctions. The orientation and saliency of the votes are the same as in Eqs. 2.2 and 2.3, but the fields are one-sided. This is because amodal completion can only occur along one direction: away from the curve that was inferred at the first stage or in the direction that would make the T-junction

(a) Parallel field (b) Orthogonal field

Figure 5.2: Voting fields used for amodal and modal completion. P is the polarity vector at the voting endpoint or junction. Voting is allowed only towards the opposite half-plane.

an X-junction. Modal completion is possible only towards the direction that results in a convex illusory contour. This is due to a strong preference in the human visual system for convex modal completions [42]. Even though the two fields are orthogonal, the polarity vector, in both cases, indicates the direction opposite to completion. Figure 5.2 shows the fields used for these cases. Before voting, the following cases have to be considered for each keypoint:

- There is no possible modal or amodal continuation due to the absence of other keypoints that support either option. In this case endpoints are just terminations, like A and B in Fig. 5.1(b).

- Amodal completion is supported by another keypoint of the same type with similar orientation and opposite polarity, while modal is not (Fig. 5.1(a)). Endpoints and stems of T-junctions cast votes with the parallel field along their tangent.

- Modal completion is possible, but amodal is not (Fig. 5.3). Support in this case has to come from keypoints with similar curve orientation and polarity. Completion occurs orthogonally to the tangent of endpoints or along the bars of T-junctions and edges of L-junctions, using the orthogonal field.

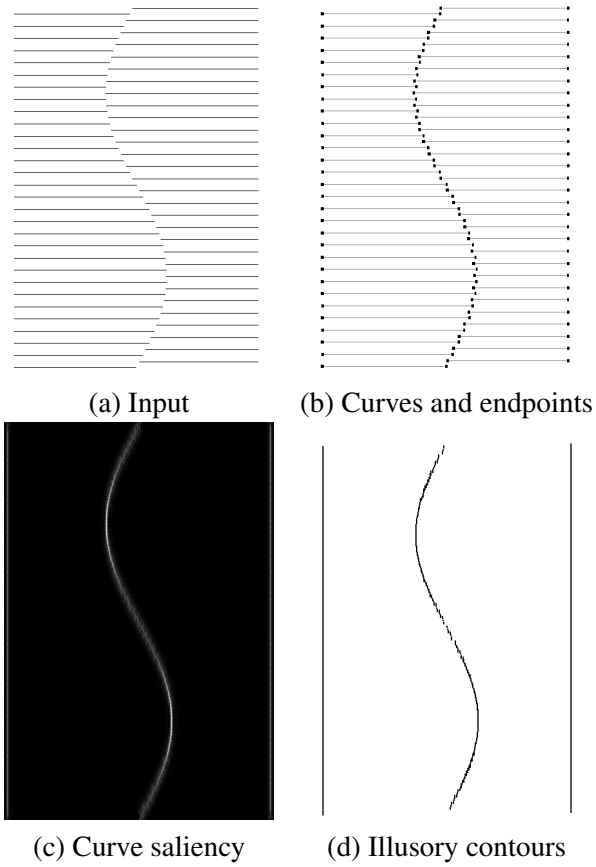
- Both types of completion are possible (Fig. 5.1(c)). In this case, the modal completion is perceived as occluding the amodal completion. In the case of the Ehrenstein stimulus, a disk is perceived to occlude the crossing lines (Fig. 5.1(e)).

Once the above cases have been examined, the keypoints can be labeled with respect to whether they support completion or not. The appropriate field is used according to the type of completion. Analysis of the votes is performed and curves, junctions and endpoints are inferred as in the previous section. Now, junctions can be fully labeled according to their cardinality and polarity. Polarity helps to discriminate between X and W junctions for instance, since the former have very low polarity while the latter have high polarity.

5.6 Experimental Results

We now present experimental results on a variety of examples. A small scale is used for the original data and a large scale (typically 20 to 30 times larger) is used to infer completions at the second stage, where the distance between tokens is considerably larger.

Illusory contour without depth ordering This is an example of the formation of an open illusory contour that does not induce the perception of a depth ordering. In that sense it is similar to Fig. 8 of [135]. The input consists of a set of unoriented tokens that form line segments and can be seen in Fig. 5.3(a). The curves and endpoints detected after a first pass of tensor voting can be seen in Fig. 5.3(b). The endpoints are marked in black and their inferred normals are orthogonal to the segments. Three illusory contours can be inferred after voting using the orthogonal field of Fig. 5.2(b). Curve saliency and the illusory contours can be seen



(a) Input (b) Curves and endpoints

(c) Curve saliency (d) Illusory contours

Figure 5.3: Modal completion. Starting from unoriented inputs, endpoints are detected and used as inputs for modal completion.

in Fig. 5.3(c-d). Higher intensity corresponds to higher saliency, as in all saliency maps shown in this document. The contour is still inferred, even though its convexity changes, since locally endpoints from either the left or the right side form a convex contour and propagate votes that support the entire sinusoidal contour.

Koffka crosses An interesting perceptual phenomenon is the Koffka crosses [172]. The perceived illusory contour changes from a circle to a square as the arms of the cross become wider.

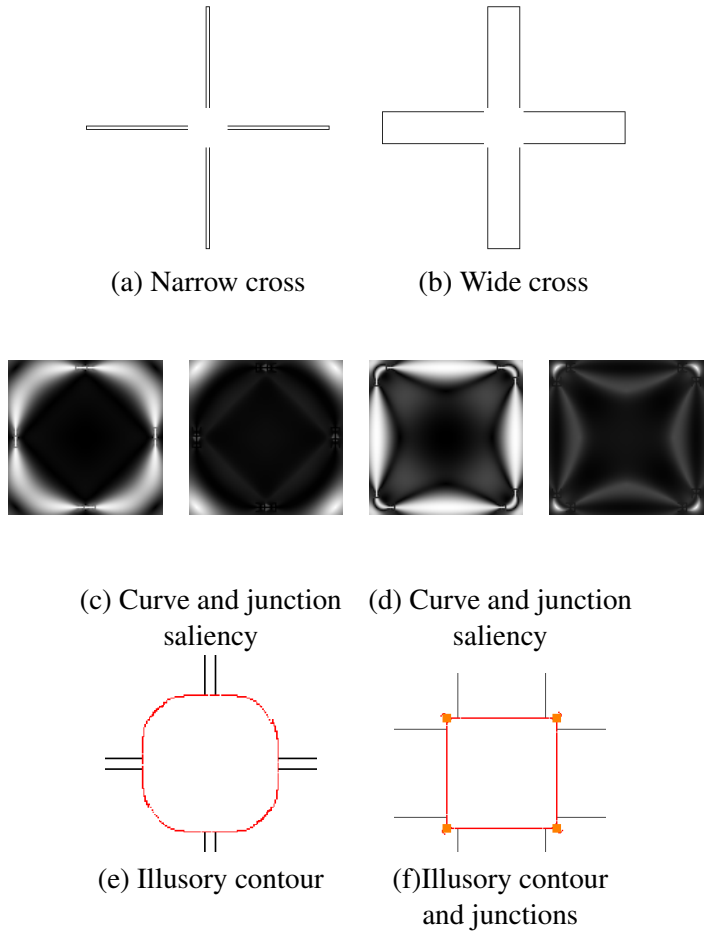
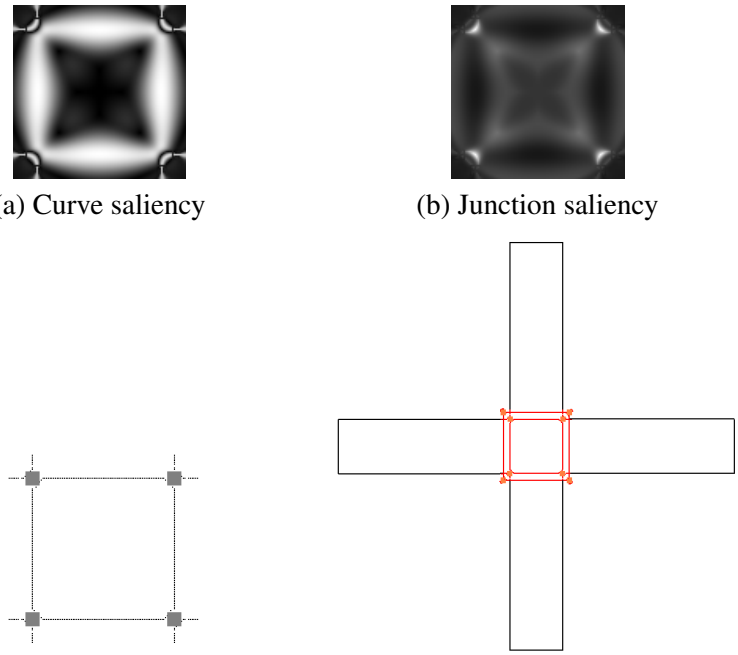


Figure 5.4: Koffka crosses. Inputs, saliency maps and inferred contours and junctions (marked as squares).

Two examples of Koffka crosses can be seen in Fig. 5.4(a-b). The black pixels of these images are encoded as unoriented ball tensors and the endpoints of the segments are extracted as before. Modal completion is possible, since the endpoints can be hypothesized as T-junctions with zero contrast bars, and voting is performed using the orthogonal field. Curve and junction saliencies are shown in Fig. 5.4(c-d). Note that the saliencies in each map are normalized independently so that white corresponds to the maximum and black to the minimum. The maximum junction



(a) Curve saliency (b) Junction saliency

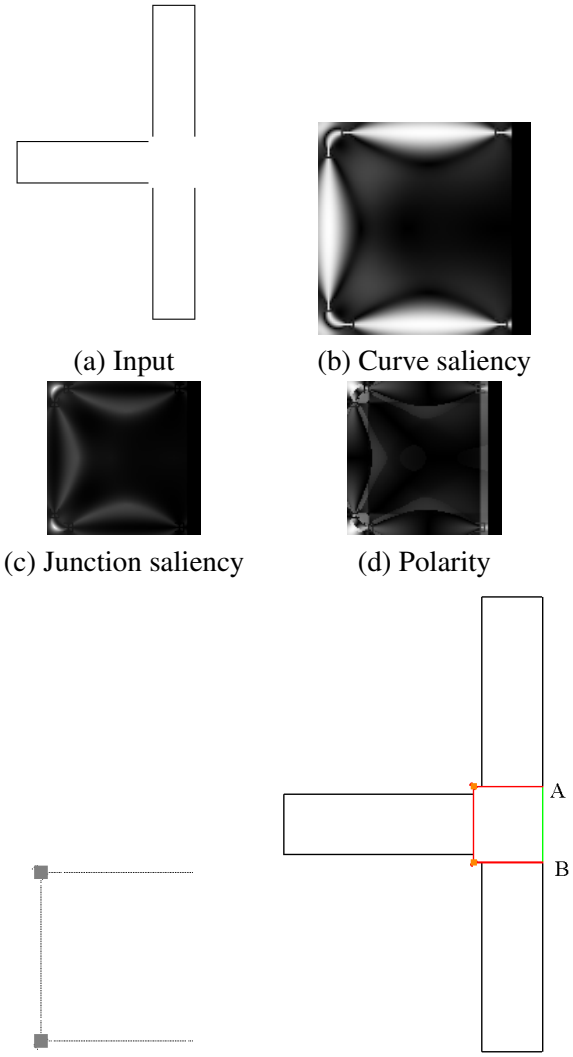
(c) Illusory contours and junctions (d) All completions

Figure 5.5: Amodal contour completion and illustration of all possible completions, including the occluded ones

saliency is 90.4% of the maximum curve saliency for the wide cross and only 9.8% of the maximum curve saliency for the narrow cross, where no junctions are inferred. Figures 5.4(e-f) show the inferred modal completion. Intermediate widths of the arms produce intermediate shapes of the illusory contour, such as rounded squares, which are consistent with human perception.

The case of amodal completion from the detected endpoints to compatible endpoints must also be considered. The parallel voting field of Fig. 5.2(a) should be used in this case. Figures 5.5(a-b) show the curve and junction saliencies for the wide Koffka cross of Fig. 5.4(b). Four contours that connect corresponding endpoints and four X-junctions are inferred (Fig. 5.5(c)).

This interpretation is also consistent with human perception, which is a layered interpretation of the scene that consists of a white square, occluding a cross, on top of a white background.



(a) Input (b) Curve saliency

(c) Junction saliency (d) Polarity

(e) Illusory contours (f) Visible completions

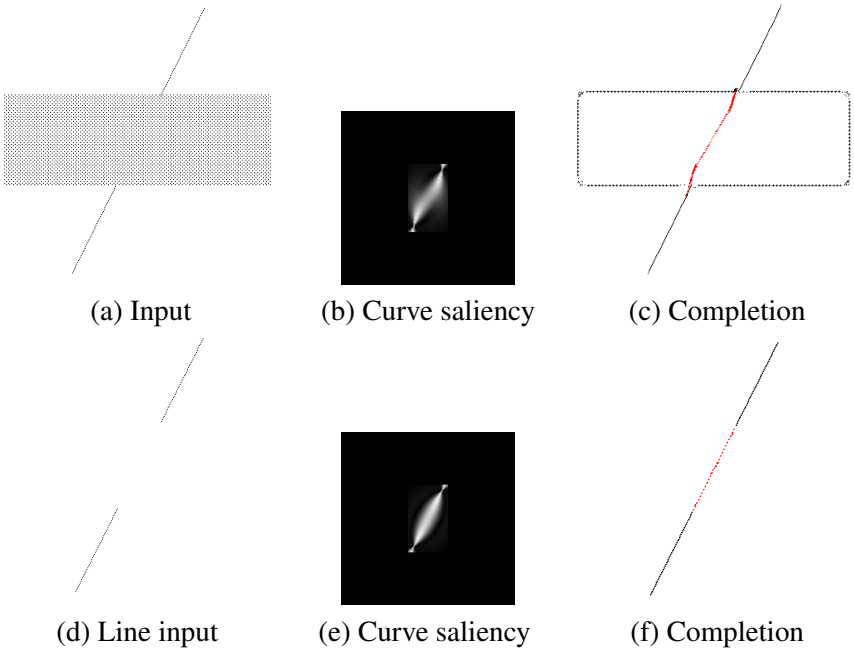
Figure 5.6: Modal completion for the three-armed cross

We also performed an experiment with one arm of the wide cross missing. The input can be seen in Fig. 5.6(a). Voting with the orthogonal field produces the saliency and polarity maps

seen in Fig. 5.6(b-d). The polarity map has four local maxima: at the two L-junctions and at two of the six voting endpoints. The inferred description consists of three straight contours, two L-junctions and two endpoints at the wrongly hypothesized T-junctions, A and B, which based on polarity can now be correctly labeled as L-junctions. Additional modal completion is now possible starting from the L-junctions that results in the contour AB on Fig. 5.6(f).

Poggendorff illusion In the final example of this chapter, we attempt to explain an illusion that occurs when a straight line does not appear perfectly straight when it passes through a region. The input can be seen in Fig. 5.7(a) and consists of unoriented points that form a line and a region. The region boundaries are inferred based on their high λ_2 and high polarity and are used as inputs to a second pass of tensor voting along with the curvels. After the second pass, four L-junctions and two T-junctions are inferred. There is no support for completion based on the L-junctions so they do not contribute any further. The T-junctions should be treated carefully in this case. The orientation of the stems is locally unreliable, since as one approaches the junction, curve saliency decreases almost to zero while junction saliency increases. Even if the orientation θ of the line farther away from the stem is used, some uncertainty remains. On the other hand, the bar and the stem of a T-junction are expected to be orthogonal [42]. Combining these two sources of evidence, we can set the orientation of the stem equal to $\alpha 90^o + (1-\alpha)\theta$. Voting from the two T-junctions using the parallel field, for a range of values of α, produces a curve saliency map like the one of Fig. 5.7(b) where completion is not straight and an inflection point always exists in the contour (Fig. 5.7(c)). This is one possible explanation for the illusion. Also, note that the illusion does not occur if the line is orthogonal to the region boundaries.

The saliency map in the absence of the region is the one shown in Fig. 5.7(e) where straight continuation occurs, as expected.



(a) Input (b) Curve saliency (c) Completion

(d) Line input (e) Curve saliency (f) Completion

Figure 5.7: Possible explanation of the illusion that straight lines appear bent when passing through a region

5.7 Discussion

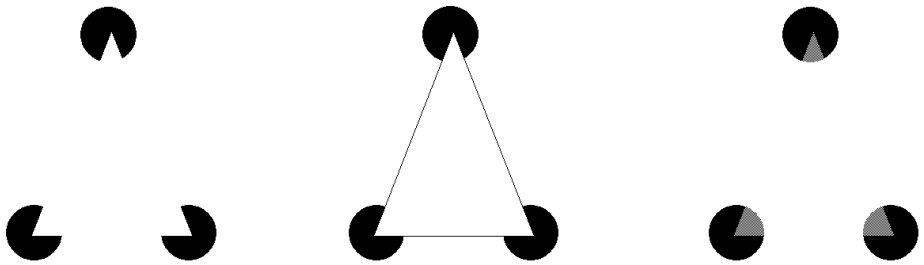
In this chapter we have presented an approach within the tensor voting framework that allows us to infer richer descriptions from single images. We begin by inferring the endpoints of curves and the boundaries of regions, and assigning preliminary labels to junctions. Moreover, we have addressed figure completion of both modal and amodal type in an automatic and integrated way. Our framework does not require a priori knowledge of the type of completion that has to

be performed, but can infer it from the data. We demonstrated how it can be applied to the interpretation of line drawings that exhibit complex perceptual phenomena.

Consistently with the body of work on tensor voting, we have avoided premature decisions based on local operators. For instance, we do not classify a location as an L-junction just because a corner detector produces a strong response. Instead, we only assign a preliminary label based on the results of first and second order voting, which for an L-junction have to produce high ball saliency and high polarity. The final labeling occurs only after the completion possibilities have been examined.

The results are encouraging. However, there are still numerous issues that need to be addressed, even in simple line drawings. Consider for instance Fig. 5.8(a) that depicts the Kanizsa triangle [64]. It contains six L-junctions and each junction supports two possibilities for completion. Either completion along the straight edge which produces the triangle of Fig. 5.8(b), or completion along the circular edge which produces the three disks seen in 5.8(c). This example is an excellent demonstration of a scenario that cannot be handled by the current state of our research. Moreover, note that making the decision on one vertex of the triangle affects the other two vertices. As demonstrated by numerous visual illusions, drawings of impossible objects and in [135], for instance, locally consistent perceptions that are globally impossible are accepted by the human visual system. Therefore, the decision on one vertex does not automatically resolve the other two. What is clear, however, is that the computer vision, psychology and neuroscience literature provide abundant examples for which a more sophisticated decision making mechanism than the one presented here are needed.

In this chapter, we have also scratched the surface of inferring hierarchical descriptions. Typically processing occurs in two stages: in the first stage, tensor voting is performed on the original low level tokens, while in the second stage, completion based on the previously inferred structures is performed. The processing stages are three in case regions are present in the dataset. Then, region boundaries are inferred in the first stage and interact with other tokens at the second stage. Completion now occurs at the third stage. More complicated scenarios may include more stages. It is reasonable to assume that scale increases from stage to stage, as the distances between the "active" points increase. A more systematic investigation of the role of scale in this context is also required. It is possible that the interpretation of certain inputs changes as the scale of voting varies.



(a) The Kanizsa triangle (b) Completion of the triangle (c) Completion of the disks

Figure 5.8: The Kanizsa triangle and two possible interpretations

Chapter 6

Binocular Stereo using Monocular Cues

In this chapter, we address the fundamental problem of matching in two static images. Significant progress has been made in this area, but the correspondence problem has not been completely solved due mostly to occlusion and lack of texture. We propose an approach that addresses these difficulties within a perceptual organization framework, taking into account both binocular and monocular sources of information. Initially, matching candidates for all pixels are generated by a combination of several matching techniques. The matching candidates are then reconstructed in disparity space, where perceptual organization takes place in 3-D neighborhoods and, thus, does not suffer from problems associated with scanline or image neighborhoods. The assumption is that correct matches form salient coherent surfaces, while wrong matching candidates do not align. Surface saliency, therefore, is used as the criterion to disambiguate matches. The matching candidates that are kept are grouped into smooth layers.

Surface over-extensions, which are systematic errors due to occlusion can be corrected at this stage by ensuring that each match is consistent in color with its neighbors of the same layer in *both* images. Matches that are not consistent in both images are most likely errors due to the foreground over-extending and covering occluded parts of the image. These are removed and the labeled surfaces are refined. Finally, the projections of the refined surfaces on both images can be used to obtain disparity hypotheses for pixels that remain unmatched. The final disparities are selected after a second tensor voting stage, during which information is propagated from more reliable pixels to less reliable ones. The proposed framework takes into account both geometric and photometric smoothness. The use of segmentation based on geometric cues to infer the color distributions of scene surfaces is arguably the most significant contribution of our research. We present results on widely used, benchmark stereo pairs taken from the Middlebury Stereo Vision Webpage (http://cat.middlebury.edu/stereo/). The left image of each pair can be seen in Fig. 6.1. A preliminary version of this work appeared in [105].

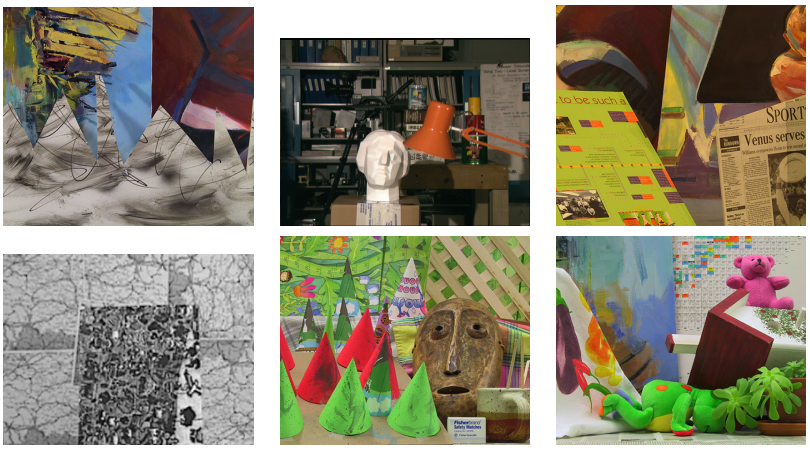


Figure 6.1: Left images from “Sawtooth, “Tsukuba”, “Venus”, “Map”, “Cones” and “Teddy” stereo pairs of the Middlebury Stereo evaluation

6.1 Introduction

The premise of shape from stereo comes from the fact that, in a set of two or more images of a static scene, world points appear on the images at different disparities depending on their distance from the cameras. Establishing pixel correspondences on real images, though, is far from trivial. Projective and photometric distortion, sensor noise, occlusion, lack of texture, and repetitive patterns make matching the most difficult stage of a stereo algorithm. Here we focus on occlusion and insufficient or ambiguous texture, which are inherent difficulties of the depicted scene, and not of the sensors.

To address these problems, we propose a stereo algorithm that operates as a perceptual organization process in the 3-D disparity space, keeping in mind that false matches will most likely occur in textureless areas, and close to depth discontinuities. Since binocular processing has limitations in these areas, we use monocular information to overcome them. We begin by generating matching hypotheses for every pixel within a flexible framework that allows the use of matches generated by any matching technique reported in the literature. These matches are reconstructed in a 3-D (x, y, d) space, where d denotes the disparity. In this space, the correct matches align to form surfaces, while the wrong ones do not form salient structures. We can, then, infer a set of reliable matches based on the support they receive from their neighbors as surface inliers via tensor voting. These reliable matches are grouped into layers. Note that the term layer is used interchangeably with surface, since by layer we indicate a smooth, but not necessarily planar, surface in 3-D disparity space. The surfaces are refined by rejecting matches that are consistent in color with their neighbors in both images. The refined, segmented surfaces

serve as the "unambiguous component" as defined in [128] to guide disparity estimation for the remaining pixels.

Segmentation using geometric properties is arguably the most significant contribution of our research. It provides very rich information on the position, orientation and appearance of the surfaces in the scene. Moreover, grouping in 3-D circumvents many of the difficulties associated with image segmentation. It is also a process that treats both images symmetrically, unlike other approaches where only one of the two images is segmented. Candidate disparities for unmatched pixels are generated after examining the color similarity of the each unmatched pixel with its nearby layers. If the color of the pixel is compatible with the color distribution of a nearby layer, disparity hypotheses are generated based on the existing layer disparities and the disparity gradient limit constraint. Tensor voting is then performed locally and votes are collected at the hypothesized locations. Only matches from the appropriate layer cast votes to each candidate match. The hypothesis that is the smoothest continuation of the surface is kept as the disparity for the pixel under consideration. In addition, assuming that the occluded surfaces are partially visible and that the occluded parts are smooth continuations of the visible ones, we are able to extrapolate them and estimate the depth of monocularly visible pixels. Under this scheme, smoothness with respect to both shape, in the form of surface continuity, and appearance, in the form of color similarity, is taken into account before disparities are assigned to unmatched pixels.

This chapter is organized as follows: Section 6.2 reviews related work; Section 6.3 is an overview of the algorithm; Section 6.4 describes the initial matching stage; Section 6.5 the detection of correct matches using tensor voting; Section 6.6 the segmentation and refinement

process; Section 6.7 the disparity computation for unmatched pixels; Section 6.8 contains experimental results; and Section 6.9 concludes the chapter.

6.2 Related Work

In this section, we review research on stereo related to ours. We focus on area-based and pixel-based methods since their goal is a dense disparity map. Feature-based approaches are not covered, even though the matches they produce can be integrated in our framework. We also consider only approaches that handle discontinuities and occlusions. The input images are assumed to be rectified and the epipolar lines to coincide with the scanlines. If this is not the case, the images can be rectified using methods such as [180].

The problem of stereo is often decomposed as the establishment of pixel correspondences and surface reconstruction, in Euclidean or disparity space. These two processes, however, are strongly linked, since the reconstructed pixel correspondences form the scene surfaces, while on the other hand, the positions of the surfaces dictate pixel correspondences in the images. In the remainder of this chapter, we describe how surface saliency is used as the criterion for the correctness of matches, as in [78] and [79]. Arguably, the first approach where surface reconstruction does not follow but interacts with feature correspondence is that of Hoff and Ahuja [49]. They integrate matching and surface interpolation to ensure surface smoothness, except at depth discontinuities and creases. Edge points are detected as features and matched across the two images at three resolutions. Planar and quadratic surface patches are successively fitted and possible depth or orientation discontinuities are detected at each resolution. The

patches that fit the matched features best are selected while the interpolated surfaces determine the disparities of unmatched pixels.

Research on dense area-based stereo with explicit treatment of occlusion includes numerous approaches (see [139] and [15] for comprehensive reviews of stereo algorithms). They can be categorized as follows: local, global, and approaches with extended local support, such as the one we propose. Local methods attempt to solve the correspondence problem using local operators in relatively small neighborhoods. Kanade and Okutomi [60] used matching windows whose size and shape adapt according to the intensities and disparities of the pixels included in them. The goal is to include as many pixels from the same disparity level as possible, and thus to increase the reliability of the match. In [162] Veksler presented a method that takes into account the average matching error per pixel, the variance of this error and the size of the window to define new matching costs and adapt the window size. Birchfield and Tomasi [9] introduced a new pixel dissimilarity measure that alleviates the effects of sampling, which are a major source of errors when one attempts to establish pixel correspondence. Their experiments, as those of [150] and ours, demonstrate the usefulness of this measure, which we use in the work presented here. Unlike all previous approaches that attempt to include in the window a large number of pixels that share the disparity of the pixel under consideration, Agrawal and Davis [1] used the matching cost of [9] for windows that can contain two different disparity values. The assignment of disparities to the pixels in each window is a bi-labeling problem that can be efficiently solved using graph cuts. The presence of an intensity edge at a pixel reduces the penalty for a depth discontinuity, thus forcing the discontinuities to align with the intensity

edges. The results of the window-based matching are then fed to a global optimization stage using graph cuts.

On the other hand, global methods arrive at disparity assignments by optimizing a global cost function that usually includes penalties for pixel dissimilarity and violation of the smoothness constraint. The latter introduces a bias for constant disparity at neighboring pixels, thus favoring frontoparallel planes. Chronologically, the first global optimization approaches to stereo were based on dynamic programming. Since dynamic programming addresses the problem as a set of 1-D sub-problems on each epipolar line separately, these approaches suffer from inconsistencies between adjacent epipolar lines that appear as streaking artifacts. Efforts to address this weakness were published as early as 1985, when Ohta and Kanade used edges to provide intra-scanline constraints [112]. Recent work also attempts to mitigate streaking by enforcing inter-scanline constraints, but the problem is not entirely eliminated.

Dynamic programming methods that explicitly model occlusion include [5, 6, 9, 11, 33, 53]. Belhumeur and Mumford [6] proposed a Bayesian approach to stereo which was extended by Belhumeur [5]. After a Bayesian formulation of image formation, the authors consider three "worlds" as the prior model of the scene. Each model is more complicated than the previous, as object boundaries and surface orientation discontinuities are added to the models. Optimization is performed by dynamic programming taking into account depth, surface orientation, depth discontinuities, surface creases and occlusion. The disparity gradient limit constraint is used to determine occluded pixels. A second stage of optimization, termed "iterated stochastic dynamic programming", is necessary to achieve inter-scanline smoothness. Bobick and Intille [11, 53] used highly reliable matches, which they termed ground control points, to constrain the path of

dynamic programming in the disparity-space image (DSI) representation. Two ground control points in the same epipolar line define the number of occluded pixels that can exist between them (as long as the ordering constraint holds), while edges in the DSI indicate likely occlusion edges. The authors also observed that the presence of an occluded region in the left image should create an intensity edge in the right image and vice versa. In addition, edges in the images provide inter-scanline constraints, which, however, do not remove all the streaking from the results. Geiger *et al.* [33] address stereo in a Bayesian framework with a smoothness prior that models occlusion and treats both images equally. It is based on the fact that a discontinuity in disparity must correspond to an occluded region in one of the images. Two off-center matching windows that avoid discontinuities either to the left or the right of the current pixel are used and the one with the minimum cost is selected as the correct match. The solution is found in matching space using dynamic programming. Cox *et al.* [20] proposed a maximum likelihood formulation that requires fewer assumptions and prior models than the Bayesian treatments. They also propose a novel way to avoid traditional regularization by minimizing the total number of horizontal and vertical disparity discontinuities instead of adding a term in the cost function. A post-processing step that enforces consistency between adjacent epipolar lines is necessary to limit the appearance of streaking artifacts. Birchfield and Tomasi [8] propose an approach based on dynamic programming with disparity propagation along columns according to reliability labels that are assigned to each pixel. Since the use of matching windows and intensity pre-processing are not valid at discontinuities, the algorithm operates at the pixel level using the pixel dissimilarity measure of [9]. The novelty of the cost function is that it rewards pixel

matches, while penalizing the number of occlusions and not the total number of occluded pixels. Thus it avoids the staircase-like results that are often produced by dynamic programming.

Consistency among epipolar lines is guaranteed by using graph cuts to optimize the objective function, since they operate in 2-D. Roy and Cox [123] find the disparity surface as the minimum cut of an undirected graph. In this framework, scanlines are no longer optimized independently, with inter-scanline coherence enforced later in a heuristic way, but smoothness is enforced globally over the entire image. A multi-camera algorithm is also proposed, but it lies outside the scope of this chapter. Ishikawa and Geiger [54] advanced graph-cut stereo by explicitly modeling occlusion and uniqueness, using a directed graph, as opposed to the undirected one of [123]. Pixels are classified as ordinary, edges and junctions, with the latter two categories providing additional constraints. The set of energy functions that can be optimized, however, is limited to convex functions which do not perform well at discontinuities. Kolmogorov and Zabih [68] propose an optimization technique based on graph cuts, that was first published in [12], which is applicable to more general objective functions. This allows a better handling of occlusion, symmetric treatment of both images and enforcement of the uniqueness constraint. In addition, unlike the majority of the methods presented so far in this section, the ordering constraint, which is violated by certain scene configurations, is no longer necessary. The authors extended their work to multiple images, and at the same time improved its binocular performance in [69].

Between these two extremes are approaches that are neither "winner-take-all" at the local level, nor global. They rely on more reliable matches to estimate the disparities of less reliable

ones. Following Marr and Poggio [90], Zitnick and Kanade [182] employed the support and inhibition mechanism of cooperative stereo to ensure the propagation of correct disparities and the uniqueness of matches with respect to both images without having to rely on the ordering constraint. Reliable matches without competitors are used to reinforce matches that are compatible with them, while at the same time, they eliminate the ones that contradict them, progressively disambiguating more pixels. Luo and Burkhardt [87] proposed a Bayesian cooperative stereo approach based on the minimization of a non-convex cost function by deterministic relaxation. Inhibition is implemented based on the observation that the occlusion map of one image can be derived from the disparity map of the other image and not from its own disparity map. In other words, the existence of a point in the scene inhibits the presence of other points at certain locations. Zhang and Kambhamettu [177] extend the cooperative framework from single pixels to image regions, segmented in the reference image. Disparities are propagated within and among image segments according to a confidence measure. The size and shape of local support areas for each match are based on image segmentation. Occlusions are detected if the converged matching score is below a threshold indicating that no good match for the pixel was found. In cooperative stereo, the presence of a good match inhibits all matches along the optical rays from both cameras. For occluded pixels, however, the absence of a good match is not a necessary conditions. For instance in Fig. 6.2, the match between A and B has no competition with respect to the left optical ray, and competes with the match between A and C with respect to the right optical ray. The scheme fails in this case, since the occluded pixel B has a good match in A.

(a) Left image (b) Right Image

Figure 6.2: Illustration of wrong matches caused by occlusion. C is the correct match for A but B is selected instead since it the correlation between the windows centered at A and B is larger than between those centered at A and C. However, this is caused by pixels of the occluding grey surface.

A different method of aggregating support is nonlinear diffusion, proposed by Scharstein and Szeliski [138], where disparity estimates are propagated to neighboring points in disparity space until convergence. The disparity space contains the matching cost for all possible disparity values for each pixel. If a diffusion operation does not increase the certainty of a match it is not performed. This avoids the over-smoothing that would be caused by effectively increasing the support region at each iteration. Sun *et al.* [147, 148] formulate the problem as an MRF with explicit handling of occlusions. In the belief propagation framework, information is passed to adjacent pixels in the form of messages whose weight also takes into account image segmentation. The process is iterative and has similar properties with nonlinear diffusion.

Sara [128] formally defines and computes the largest unambiguous component of stereo matching, which can be used as a basis for the estimation of more unreliable disparities. Other similar approaches include those of Szeliski and Scharstein [150] and Zhang and Shan [178] who start from the most reliable matches and allow the most certain disparities to guide the estimation of less certain ones, while occlusions are explicitly labeled. A different approach

employing genetic algorithms was proposed by Goulermas and Liatsis [36]. The image is uniformly divided in rectangular blocks and a symbiotic genetic algorithm operates on each block. The population of each block has two objectives: its self score, which is based on image intensities and gradients as well as geometric constraints, and the symbiotic score that enforces continuity between the blocks. Low-order polynomial surface patches are fitted for each image block. Processing is parallel with interactions between adjacent blocks. Our approach is similar to those of this paragraph in the sense that disparities are propagated from reliable to less reliable matches.

The final class of methods reviewed here utilizes monocular color cues (image segmentation) to guide disparity estimation. Birchfield and Tomasi [10] cast the problem of correspondence as image segmentation followed by the estimation of affine transformations between the images for each segment. The objective energy function does not favor constant disparity and frontoparallel surfaces, but can account for affine warping and slanted surfaces. Initially the image is segmented and then the affine parameters are estimated for each segment. The final disparity map is produced after a few iterations. Tao *et al.* [154] introduced a stereo matching technique where the goal is to establish correspondence between image regions rather than pixels. It achieves outstanding results in cases where traditional stereo fails, namely in scenes with large uniform regions that lack any meaningful intensity variations. The parameters of the affine transformation of each image segment are optimized according to the projection of the segment on the target image, taking into account possible occlusions. Both these methods are limited to planar surfaces, unlike the one of [177], which was described above, where image regions are decomposed into smaller planar patches. Lin and Tomasi [83] propose a framework where

3-D shape is estimated by fitting splines, while 2-D support is based on image segmentation. Processing alternates between these two steps until convergence.

Recently, Wei and Quan [167] proposed a region-based progressive algorithm where reliable matches are used as ground control points to provide disparity estimates for image regions, which are obtained from color segmentation of the left image. Since the assumption is that regions have constant disparity, they are split if they contain ground control points with multiple disparities. Then disparities are propagated from more to less reliable regions, making sure that more reliable pixels are not occluded by newly assigned disparities. Hong and Chen [50] also start by performing color segmentation of the reference image. Planes are fitted to each region and their disparities and parameters are optimized within a graph cut framework that operates on regions instead of pixels. The last two methods achieve outstanding performance on the Middlebury stereo evaluation datasets, with the exception of the "Map" where the failure of image segmentation proves to be catastrophic, especially near depth discontinuities. This problem occurs because all these approaches, except that of [148], use image segmentation as a hard constraint, while segmentation itself is by no means a trivial problem.

6.3 Overview of our Approach

Our approach to the derivation of dense disparity maps from rectified image pairs falls into the category of area-based stereo since we attempt to infer matches for every pixel using matching windows. It has four steps, which are illustrated in Fig. 6.3, for the "Sawtooth" stereo pair. The steps are:

- *Initial matching*, where matching hypotheses are generated for every pixel by a combination of different matching techniques. The dataset now includes multiple candidate matches for each pixel and can be seen in Fig. 6.3(b).

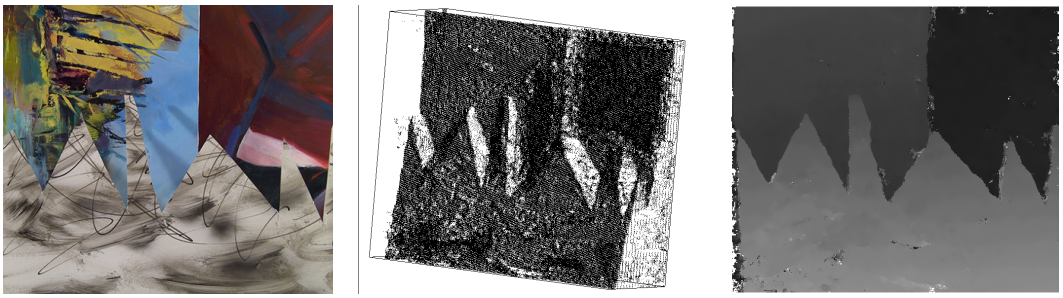
- *Detection of correct matches* which uses tensor voting to infer the correct matches from the unorganized point cloud of the previous stage as inliers of salient surfaces. After tensor voting, uniqueness is enforced with respect to surface saliency and the dataset contains at most one candidate match per pixel. The disparity map can be seen in Fig. 6.3(c).

- *Surface grouping and refinement* during which the matches are grouped into smooth surfaces, using the estimated surface orientations. These surfaces are refined by removing points that are inconsistent with the layer's color distribution resulting in the disparity map of Fig. 6.3(d).

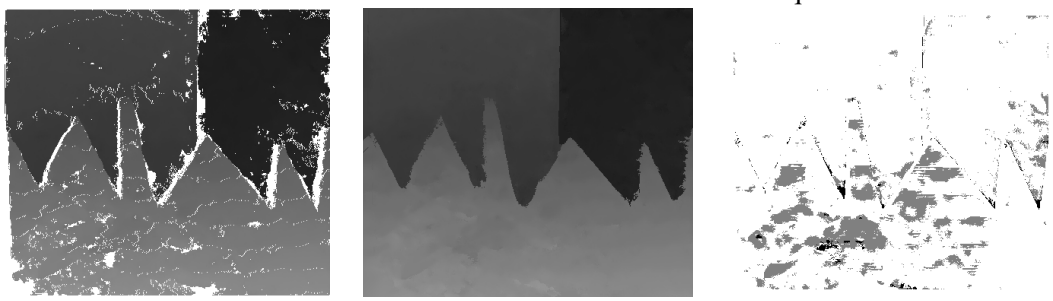
- *Disparity estimation for unmatched pixels*, where the goal is the assignment of disparities that ensure smoothness in terms of both surface orientation and color properties of the layers. The final disparity map and the error map can be seen in Figs. 6.3(e) and (f).

These steps along with experimental evaluations are presented in Sections 6.4 through 6.7. In the remainder of this section we focus on important aspects of our research and how it compares to other research on stereo.

A number of pixel matching techniques are reported in the literature, each having different strengths and weaknesses. For this reason, we propose to combine them in order to maximize



(a) Left image (b) Initial matches (c) Disparities after uniqueness enforcement



(d) Reliable matches (e) Final disparities (f) Error map

Figure 6.3: Overview of the processing steps for the "Sawtooth" dataset. The initial matches have been rotated so that the multiple candidates for each pixel are visible. Black pixels in the error map indicate errors greater than one disparity level, gray pixels correspond to errors between 0.5 and 1 disparity level, while white pixels are correct (or occluded and thus ignored).

the number of correct candidate matches, which form the scene surfaces when they are reconstructed in disparity space. The combination of multiple matching techniques significantly enhances the performance of this algorithm over the work of [79]. The choice of these local operators is critical for the success of any stereo algorithm. However, the problem of stereo in its entirety, taking into account occlusions and discontinuities, cannot be fully solved at the pixel level.

Support for each potential match has to be aggregated, so that the confidence of correct matches is increased and outliers are made explicit. Aggregation in 1-D neighborhoods is only

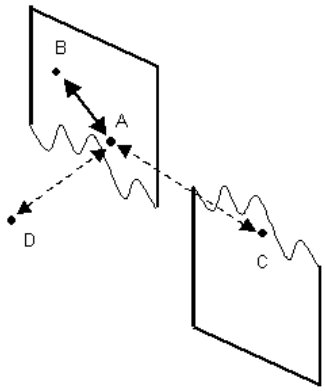


Figure 6.4: Voting in 3-D neighborhoods eliminates interference between adjacent pixels from different layers

motivated by computational simplicity and its shortcomings are well documented. While graph cut based methods have achieved outstanding results, the choice of an appropriate energy function is not an easy task. Energy functions whose global minima can be found with current optimization techniques do not necessarily model the phenomenon of stereovision in its most general form. In most cases, the disparity assignment that achieves the globally minimal energy is not the ground truth disparity of the scene. This occurs because the energy function has to satisfy certain properties to be suitable for minimization using algorithms, such as graph cuts or belief propagation [155]. For instance, the penalization of disparity changes between neighboring pixels makes these approaches well suited for scenes that consist of fronto-parallel planes and prefers stair-case looking solutions for slanted or curved surfaces. In this chapter, following the approach of Lee *et al.* [79] we aggregate support in 3-D neighborhoods via tensor voting. Pixels that are close in one image but are projections of remote world points do not interact. Figure 6.4 shows that points A and B that are close in 3-D, and therefore are likely to belong in the same scene surface, interact strongly with each other. On the other hand, points A and C

that are close in the image but not in 3-D, and therefore are most likely projections of unrelated surfaces, do not vote to each other. Finally, point D, which is isolated in 3-D and is probably generated by an error in the initial matching stage, receives no support as an inlier of a salient surface. After accumulating support by tensor voting, candidate matches that are consistent with their neighbors have high surface saliency, which validates them as correct matches.

Since the ordering constraint is violated by scene configurations that are not unlikely, such as the presence of thin foreground objects, we do not enforce it. Its popularity in the literature is mostly as a byproduct of optimization techniques. As optimization techniques have improved, most researchers have abandoned the ordering constraint. The uniqueness constraint, which states that in the absence of transparency, there should be at most one match for each pixel, should also be enforced carefully. As Ogale and Aloimonos [111] point out, if scene surfaces exhibit horizontal slant (that is if the epipolar line in the image and the intersection of the epipolar plane and the scene surface are not parallel), then M pixels in one image necessarily correspond to N pixels in the other image. Therefore, requiring a strict one-to-one correspondence for all pixels results in labeling $|M - N|$ pixels as occluded. These pixels that are interleaved with matched pixels, however, are perfectly visible in both images, just not at integer coordinate positions. Keeping this observation in mind, we only enforce uniqueness as a post-processing step allowing at most one match for each pixel of the reference image in order to derive a dense disparity map. More than one pixel of the reference image is allowed to correspond to the same pixel of the target image (with integer or subpixel disparities) if the surface appears wider in the reference image. Thus, visible pixels are not marked as being occluded.

A rather safe conclusion that can be drawn from the Middlebury stereo evaluation is that the use of monocular information, such as color, contributes to the performance of a stereo algorithm. In [105], we proposed a novel way of integrating monocular information that requires very few assumptions about the scene and does not fail when image segmentation fails. The use of monocular information is a major improvement over the work of [78, 79]. Instead of trying to identify the scene surfaces by their projections on the images via their color properties, we try to infer them in disparity space via image segmentation. Candidate matches that were retained after tensor voting are grouped into smooth surfaces based on their positions and estimated surface normals. Then, these surfaces are re-projected to both images and points that are inconsistent with the other points of the surface in terms of color distribution in either image are rejected. This step removes erroneous matches for occluded pixels, which are usually assigned the disparity of the foreground. They are removed since they do not project to the same surface in both images, and thus the color distributions are inconsistent. Under this scheme, both images are treated symmetrically, unlike most segmentation-based methods where only the reference image is segmented. Furthermore, we do not attempt to segment the image, but instead solve a simpler problem: grouping points, that were selected as surface inliers, into smooth 3-D surfaces.

The final step is the assignment of disparities to unmatched pixels. One can view the retained matches from the previous stage as the "reliable" matches of a progressive scheme, since they are both consistent geometrically with their 3-D neighbors and color-consistent with their neighboring pixels in both images. Disparities are propagated from these pixels to unmatched ones ensuring smoothness in terms of both geometry and color properties. We are also able to

obtain disparity estimates for occluded pixels by enforcing smoothness with respect to surface orientation and color consistency with respect to the appropriate image. These estimates are accurate as long as there are no abrupt changes in the monocularly visible parts of each surface.

6.4 Initial Matching

A large number of matching techniques have been proposed in the literature [139]. We propose a scheme for combining a variety of matching techniques, thus taking advantage of their combined strengths. For the results presented in this chapter, four matching techniques are used, but any type of matching operator can be integrated in the framework. The techniques used here are:

- A small (typically 5×5) normalized cross correlation window, which is small enough to capture details and only assumes constant disparity over small windows of the image. This technique is referred to as the "correlation window" in the remainder of this chapter.

- A *shiftable* normalized cross correlation window of the same size as the above. The fact that it is shiftable improves performance near discontinuities. It is referred to as the "shiftable window" in the remainder of this chapter.

- A 25×25 normalized cross correlation window, which is applied only at pixels where the standard deviation of the three color channels is less than 20. The use of such a big window over the entire image would be catastrophic, but it is effective when applied only in virtually textureless regions, where smaller windows completely fail to detect correct matches. This technique is referred to as the "large window".

- A *symmetric interval* matching window (typically 7×7) with truncated cost function as in [150]. This is referred to as the "interval window".

Note that the typical window sizes are for image resolutions similar to those of the Middlebury image pairs, which range from 284×216 to 450×375. Larger window sizes would most likely be necessary for higher resolution images.

Correlation windows This is a one of the most common and popular approaches for the establishment of pixel correspondences. It performs well over a wide range of scene types and imaging conditions. We choose it over alternatives such as the sum of absolute or squared differences, because it is invariant to camera gain, and thus more general. The correlation coefficients for all possible disparities values of each pixel are computed and we keep all peaks of the correlation function, if their magnitude is comparable to the maximum for the pixel, since they are good candidates for correct pixel correspondences. They are used as inputs for the tensor voting stage, where the decisions are made based on surface saliency and not the correlation coefficient itself, since it can be affected by factors, such as repetitive patterns or the degree of texture of one surface over the other. These factors may cause wrong matches if decisions are made based solely on the correlation coefficients. See [129] for an analysis of the effects of texture in correlation-based matching.

Shiftable windows We also use shiftable correlation windows due to their superior performance near depth discontinuities. The limitation of window-based matching is that, no matter how small the window is, pixels from two or more surfaces are included in it at discontinuities. By not centering the window on the pixel under consideration, we can find a shift that

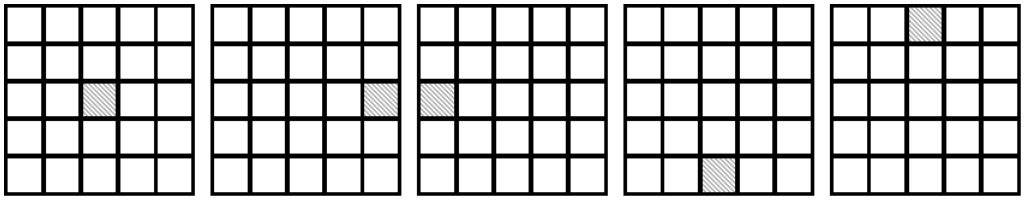


Figure 6.5: The five shiftable windows applied for each disparity choice at every pixel. The shaded square corresponds to the pixel under consideration. The same window is applied to the target image.

includes as many pixels from the same surface as the pixel under consideration as possible. See Fig. 6.5 for the five windows used here. Given a pixel in the reference image, we compute cross correlation for each disparity level for five different window shifts around the pixel under consideration, and keep the one with the maximum correlation coefficient as the score for that disparity level. As with correlation windows, we keep all significant peaks of the score function as candidate matches. The following table shows the performance of regular and shiftable correlation windows on the four initial Middlebury image pairs. The performance metric used is the number of correct matches, up to one disparity level off from the ground truth, over the total number of matching candidates. The same metric is reported for pixels at discontinuities based on the discontinuities maps provided by the authors of the webpage. Even though shiftable windows perform better, correlation windows also add valuable candidate matches and reinforce the ones on which both operators agree. Our experiments show that using both improves performance.

Large windows This a different correlation-based matching technique that aims at producing correct disparity estimates for untextured areas. It is only applied to parts of the image where color variance is below a certain threshold, and thus never near discontinuities, where color

Image Pair	Size	Regular		Shiftable	
		Total (%)	Disc. (%)	Total (%)	Disc. (%)
Tsukuba	5×5	62.1	57.2	62.0	62.1
	7×7	66.9	55.6	66.9	62.3
	9×9	70.3	50.0	69.4	60.8
Sawtooth	5×5	88.3	51.2	90.0	69.8
	7×7	92.4	63.1	94.4	68.8
	9×9	93.7	59.6	95.7	68.3
Venus	5×5	75.7	61.7	75.4	64.7
	7×7	81.9	58.2	81.9	62.0
	9×9	85.9	54.9	85.7	60.3
Map	5×5	96.9	68.6	98.0	70.9
	7×7	98.4	67.7	98.9	70.2
	9×9	98.2	64.4	98.8	68.7

Table 6.1: Percentage of good matches generated by regular and shiftable correlation windows over all un-occluded pixels and discontinuities

variance is necessarily high. For the experiments presented here, a 25×25 window was applied to pixels where the color variance, within the 25×25 window, was less than 20 intensity levels. A final step is required in this case for the rejection of unreliable matches. This is especially important here, since we are specifically targeting the most ambiguous pixels of the image. We perform a simple test to determine the reliability of each match. The correlation coefficients for each disparity of a given pixel are divided by the sum of all correlation coefficients for that pixel to give the "normalized score" of each disparity. This allows us to detect matching candidates with high uncertainty. For instance, if the disparity range is 20 and all correlation coefficients are equal, the normalized score for all disparity levels would be 5%. If the normalized score of a matching candidate is not significantly larger than 5%, the matching candidate is unreliable. For the experiments in the remainder of the chapter, we reject matching candidates with normalized

scores below 20% of the average normalized score over all matching candidates. What should be noted is that multiple matches for each pixel are still allowed, and often occur.

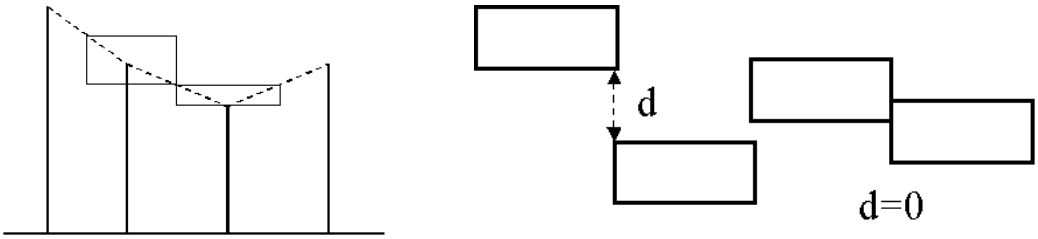
Interval windows The final matching technique is very different from the above, not only because we use the matching cost of [9], but mostly because of the truncation of the cost for each pixel at a certain level. That makes the behavior robust against pixels from different surfaces that have been included in the window. Our implementation is that of Szeliski and Scharstein [150]. Both images are linearly interpolated along the x-axis so that samples exist in half-pixel intervals. The intensity of each pixel, in each of the three color channels, is now represented as the interval between the minimum and maximum value of the intensity at the integer pixel position and the half-pixel positions before and after it on the scanline, as shown in Fig. 6.6.

Numerically, the cost for matching pixel (x_L, y) in the left image with pixel (x_R, y) in the right image is the minimum distance between the two intervals, which is given by the following equation and is zero if they overlap:

$$C(x_L, x_R, y) = \sum_{c \in \{R,G,B\}} min\{dist(I_{Lc}(x_i, y), I_{Rc}(x_j, y)), c_{trunc} : \\ x_i \in [x_L - \frac{1}{2} \quad x_L + \frac{1}{2}], x_j \in [x_R - \frac{1}{2} \quad x_R + \frac{1}{2}]\} \quad (6.1)$$

The summation is over the three color channels and $dist()$ is the Euclidean distance between the value of a color channel I_{Lc} in the left image and I_{Rc} in the right image. If the distance for any channel exceeds the truncation parameter c_{trunc}, the total cost is set to $3c_{trunc}$. Typical values for c_{trunc} are between 3 and 10. For the experiments presented in Section 6.8, a value

of 5 was used. Even though, statistically, the performance of interval windows is slightly worse than that of the shiftable windows, both overall and at discontinuities, and worse overall than the correlation windows, they are useful because they produce correct disparity estimates for pixels where the other windows fail due to the different nature of the dissimilarity measure and the robust formulation we use.



(a) Interpolation and interval representation (b) Distance between intervals

Figure 6.6: Symmetric interval matching. Both images are interpolated and color distance is computed between the left and right interval (and not an interval and a pixel).

Each matching technique is repeated using the right image as reference and the left as target. This increases the true positive rate especially near discontinuities, where the presence of occluded pixels in the reference window affects the results of matching. When the other image is used as reference, these pixels do not appear in the reference window. A simple parabolic fit [139] is used for subpixel accuracy, mainly because it makes slanted or curved surfaces appear continuous and not staircase-like. Computational complexity is not affected since the number of matching hypotheses is unchanged and it is independent of the number of permissible disparity levels. Besides the increased number of correct detections, the combination of these matching techniques offers the advantage that the failures of a particular technique are not

detrimental to the success of the algorithm, as long as the majority of the operators do not produce the same erroneous disparities. Our experiments have also shown that the errors produced by small windows, such as the 5×5 and 7×7 used here, are randomly spread in space and do not usually align to form non-existent structures. This property is important for our methodology that is based on the perceptual organization, due to good alignment, of candidate matches in space. On the other hand, the large window, which is more susceptible to systematic errors, is never applied near discontinuities.

6.5 Detection of Correct Matches

This section describes how correct matches can be found among the hypotheses of the previous stage by examining how much support they receive from their neighboring candidate matches to form smooth 3-D surfaces. The goal here is to address stereo as a perceptual organization problem, based on the premise that the correct matches should form coherent surfaces in disparity space. This is the only part of our approach that is based on [79]. The input is a cloud of points in a 3-D space $(x, y, zscale \times d)$, where $zscale$ is a constant used to make the input less flat with respect to the d-axis, since is disparity space is usually a lot flatter than actual (x, y, z). Its typical value is 8 and the sensitivity is extremely low for a reasonable range such as 4 to 20. The quantitative matching scores are disregarded and all candidate matches are initialized as unoriented ball tensors with saliency equal to one. If two or more matches fall within the same $(x, y, zscale \times d)$ voxel their initial saliencies are added, thus increasing the confidence of candidate matches confirmed by multiple matching techniques. Since d is

estimated with subpixel accuracy each integer disparity level has $zscale$ possible subpixel levels. Therefore, quantization occurs at a finer resolution and the dimensions of each voxel are $pixel_{size} \times pixel_{size} \times zscale$.

The inputs, encoded as ball tensors, cast votes to their neighbors. What should be pointed out here is the fact that, since information propagation is performed in 3-D, there is very little interference between candidate matches for pixels that are adjacent in the image but come from different surfaces as shown in Fig. 6.4. This is a major advantage over information propagation between adjacent pixels, even if it is mitigated by some dissimilarity measure.

When voting is completed, the surface saliency of each candidate match can be computed as the difference between the two largest eigenvalues of the tensor. Uniqueness is enforced with respect to the left image by retaining the candidate with the highest surface saliency for every pixel. We do not enforce uniqueness with respect to the right image since it is violated by slanted surfaces which project to a different number of pixels on each image. Since the objective is disparity estimation for every pixel in the reference image, uniqueness applies to that image only. Surface saliency is a more reliable criterion for the selection of correct matches than the score of a local matching operator, because it requires that candidate matches, identified as such by local operators, should also form coherent surfaces in 3-D. This scheme is capable of rejecting false positive responses of the local operators, which is not possible at the local level. The resulting datasets still contain errors, which are corrected at the next stage. Note that we have eliminated a free parameter from the algorithm of [105] by not thresholding with respect to surface saliency, but instead feeding all matching candidates after uniqueness enforcement to the next stage.

6.6 Surface Grouping and Refinement

Candidate matches that have not been rejected are grouped in layers using a simple growing scheme. By layers, we mean surfaces with smooth variation of surface normal. They do not have to be planar, and the points that belong to them do not have to form one connected component.

Labeling starts from seed matches that have maximum surface saliency. Since the input to this stage is rather dense and includes candidate matches for a large percentage of the matches, we only examine the eight nearest neighbors of the seed. If they are smooth continuations of the growing surface they are added to it and their neighbors are also considered for addition. The disparity gradient limit constraint dictates that the maximum disparity jump between two pixels of the same surface is one. When no more matching candidates can be added to the surface, the unlabeled point with maximum surface saliency is selected as the next seed. Small surfaces comprising less than 0.5% of image pixels are removed, since they are probably noisy patches, unless they are compatible with a larger nearby surface in terms of both position and orientation. Support from a larger surface means that the small part is most likely correct, but due to occlusion or failure of the matching operators, is not connected to the main part of the surface. After this step, the dataset consists of a set of labeled surfaces, which contain errors mostly due to foreground over-extension. A number of candidate matches that survived uniqueness enforcement while not being parts of large salient surfaces are also removed here. These include wrong matches in uniform areas, which are not aligned with the correct matches.

The next step is the refinement of the layers. The goal is to remove the over-extensions of the foreground by ensuring that the color properties of the pixels, which are the projections of the grouped points, are *locally* consistent within each layer. Color consistency of a pixel is

checked by computing the ratio of pixels of the same layer with similar color to the current pixel over the total number of pixels of the layer within the neighborhood. This is repeated in the target image and if the current assignment does not correspond to the maximum ratio *in both images*, then the pixel is removed from the layer. The color similarity ratio for pixel (x_0, y_0) in the left image with layer i can be computed according to the following equation:

$$R_i(x_0, y_0) = \frac{\sum_{(x,y)\in N} T(lab(x,y) = i \, AND \, dist(I_L(x,y), I_l(x_0,y_0) < c_{thr}))}{\sum_{(x,y)\in N} T(lab(x,y) = i))} \tag{6.2}$$

Where $T()$ is a test function that is 1 if its argument is true, $lab()$ is the label of a pixel and c_{thr} is a color distance threshold in RGB space, typically equal to the c_{trunc} parameter of the interval windows. The same is applied for the right image for pixel $(x_0 - d_0, y_0)$. The size of the neighborhood is the second and final parameter of this stage. It can be set equal to the range of the voting field during tensor voting.

Image Pair	Total before	Error rate before	Total after	Error rate after
Tsukuba	84810	5.31%	69666	1.33%
Sawtooth	144808	2.95%	136894	1.08%
Venus	147320	6.16%	132480	1.24%
Map	48657	0.44%	45985	0.05%
Cones	132856	4.27%	126599	3.41%
Teddy	135862	7.24%	121951	4.97%

Table 6.2: Total and wrong matches in each dataset before and after surface grouping and refinement

This step corrects surface over-extension that occurs near occlusions, since the over-extensions are usually not color-consistent in both images and are thus detected and removed. Table 6.2 shows the total number of candidate matches and errors before and after refinement for the

four Middlebury image pairs. The disparity maps for the "Sawtooth" example before and after grouping and refinement can be seen in Figs. 6.3(c) and (d).

6.7 Disparity Estimation for Unmatched Pixels

The goal of this stage is to generate candidate matches for the remaining unmatched pixels. Given the already estimated disparities and labels for a large set of the pixels, there is more information available now that can enhance our ability to estimate the missing disparities. We opt for a progressive approach, under which only the most reliable correspondences are allowed at first. These are correspondences that satisfy strict geometric and color requirements in both images. The requirements become less strict as we proceed.

Given an unmatched pixel in the reference image, we examine its neighborhood for layers to which the pixel can be assigned. Color similarity ratios are computed for the pixel with respect to all layers as in Eq. 6.2. The layer with the maximum ratio is selected as the potential layer for the pixel. Then, we need to generate a range of disparities for the pixel. This is done by examining the disparity values of the selected layer's pixels in the neighborhood. The range is extended according to the disparity gradient limit constraint, which holds perfectly in the case of rectified parallel stereo pairs. Disparity hypotheses are verified one by one on the target image by computing similarity ratios, unless they are occluded, in which case we allow occluded surfaces to grow underneath the occluding ones. On the other hand, we do not allow new matches to occlude existing consistent matches. Votes are collected at valid potential matches in disparity space, as before, with the only difference being that only matches from the appropriate layer cast votes (see Fig. 6.7). The most salient among the potential

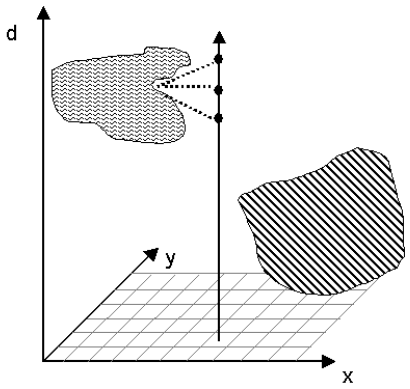


Figure 6.7: Candidate generation for unmatched pixels based on segmented layers. The unmatched pixel is compatible with the left surface only, thus votes are collected at disparity hypotheses generated by matches of the left surface. Also note that only matches from the appropriate layer vote at each candidate.

matches is selected and added to the layer, since it is the one that ensures the smoothest surface continuation.

For the results presented here, we applied the following progressive growing scheme, which has two parameters: c_{thr}, which is the color threshold used for computing the similarity ratios and σ_3, the scale of voting for densification which also defines the size of the neighborhood in which similarity ratios are computed. For the first iteration, we initialize the parameters with $c_{thr} = 1$ and $\sigma_3^2 = 20$. These are very strict requirements and have to be satisfied on both images for a disparity hypothesis to be valid. Votes are accumulated on valid hypotheses, which also do not occlude any existing matches, and the most salient continuation is selected. We, then, repeat the process without requiring consistency with the target image and add more matches, which usually are for occluded pixels that are very similar to their unoccluded neighbors. The added matches are generally correct, but valid hypotheses cannot be generated for all pixels. In the second iteration we increment both c_{thr} and σ_3^2 by their initial values and repeat the same

process. The choice of parameters here is not critical. For instance, maintaining a constant σ_3 produces very similar results. For the experiments shown here, both parameters are increased by constant increments at each iteration until convergence.

Typically, there are a few pixels that cannot be resolved because they exhibit low similarity to all layers, or because they are specular or in shadows. Candidates for these pixels are generated based on the disparities of all neighboring pixels and votes are collected at the candidate locations in disparity space. Again, the most salient ones are selected. We opt to use surface smoothness at this stage instead of image correlation, or other image based criteria, since we are dealing with pixels where the initial matching and color consistency failed to produce a consistent match.

6.8 Experimental Results

This section contains results on the color versions of the four image pairs of [139] and the two proposed in [140], which are available online at http://cat.middlebury.edu/stereo/. *All six examples were processed with identical parameters*. The initial matching in all cases was done using the four matching operators presented in Section 6.4 using both the left and right image as reference. The correlation and shiftable windows were 5×5, the interval windows were 7×7 with the truncation parameter set at 5 and the large window was 25×25, applied at pixels with intensity variance less than 20. For the large windows only, pixels with normalized score below 20% of the average were rejected. The scale of the voting field for the detection of correct matches was $\sigma^2 = 50$ which corresponds to a voting radius of 14, or a neighborhood of $29 \times 29 \times 29$.

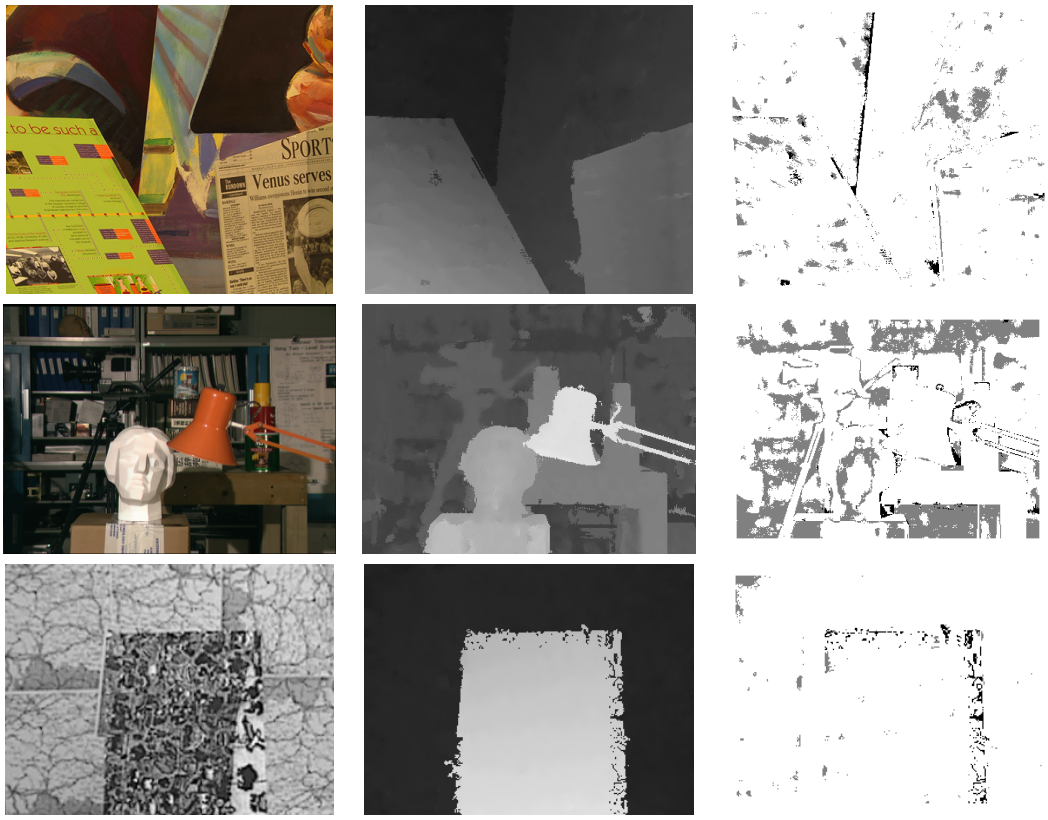


Figure 6.8: Left images, final disparity maps and error maps for the “Venus”, “Tsukuba” and “Map”image pairs from the Middlebury Stereo evaluation

Refinement was performed with a voting radius of 18 and c_{thr} equal to 5. In the final stage, c_{thr} was initialized as 1 and incremented by 1 for 25 iterations, while σ_3^2 was initialized as 20 and incremented by 20 at each iteration.

A second surface refinement operation was performed to remove errors around the surface discontinuities. This time, the voting radius was significantly smaller, set equal to 7, since we are only interested in correcting the borders of each surface. The value of c_{thr}, on the other

Image pair	Error Rate (%)	Rank	Best (%)
Tsukuba	1.51	11	0.97
Sawtooth	0.70	12	0.19
Venus	1.09	12	0.08
Map	1.31	24	0.16

Table 6.3: Quantitative evaluation for the original Middlebury stereo image pairs. The last two columns report our current ranking and the best current performance by any algorithm.

hand, was equal to 40, to allow larger color variation within each surface. The parameters for the final stage were identical with those of the previous paragraph.

The error metric reported in the tables is the one proposed in [139], where matches are considered erroneous if they correspond to un-occluded image pixels and their disparity error is greater than one integer disparity level. Table 6.3 contains the error rates we achieved, as well as the rank our algorithm would achieve among the 38 algorithms in the evaluation. We refer readers to the Middlebury College Stereo Evaluation webpage for results obtained by other methods. Based on the overall results for unoccluded pixels, our algorithm would rank 15^{th} in the evaluation as of June 7, 2005.

Table 6.4 reports results for the two image pairs of [140] and results of three stereo algorithms, sum of squared differences (SSD), dynamic programming (DP) and graph cuts (GC) implemented by the authors of [140]. To our knowledge, our results are the best published for these image pairs. These image pairs contain curved and slanted surfaces, with different degrees of detail and texture, and are, thus, more challenging. This is more pronounced for methods that make the assumption that scene surfaces are planar and parallel to the image plane. This assumption is explicitly made when one penalizes disparity differences between neighboring pixels.

Image pair	Our result	SSD	DP	GC
Cones	3.76%	10.6%	10.5%	7.7%
Teddy	6.49%	16.5%	14.0%	16.5%

Table 6.4: Quantitative evaluation for the new Middlebury stereo image pairs

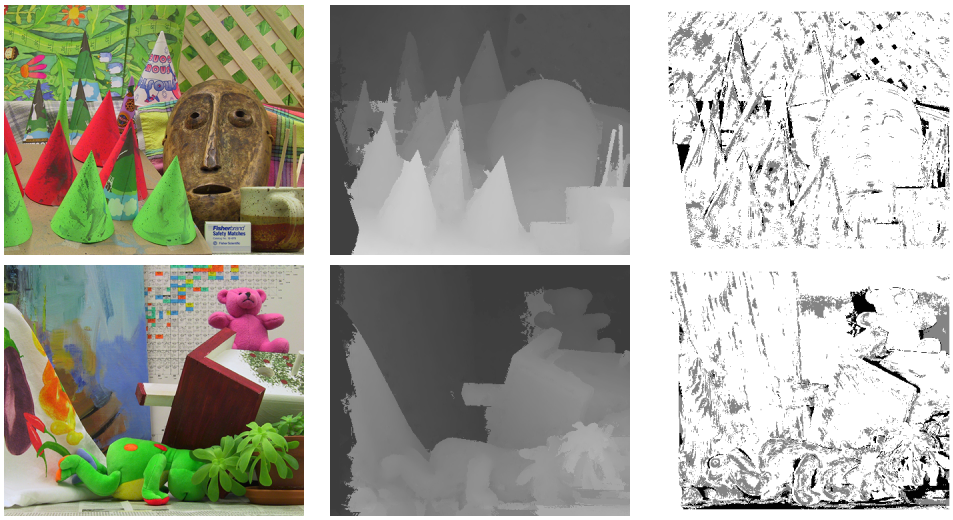
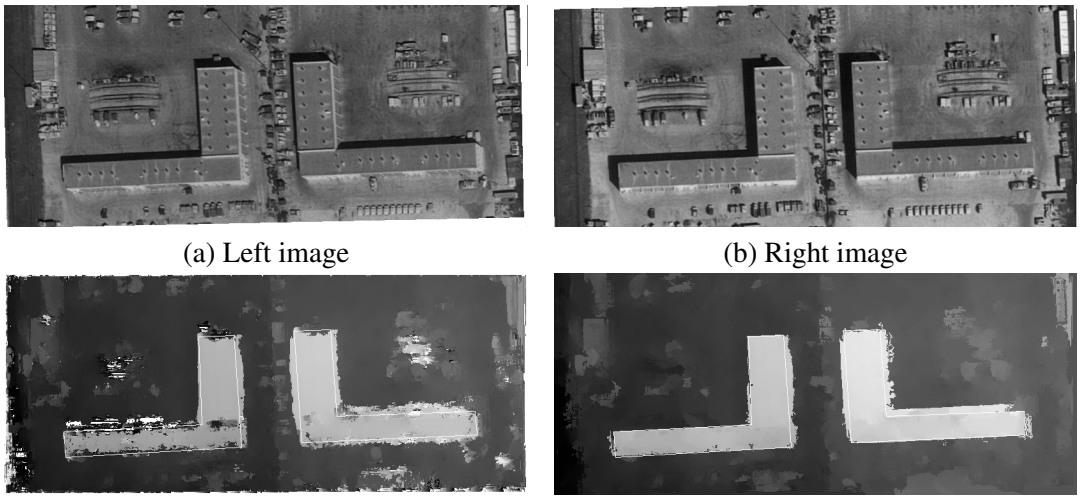


Figure 6.9: Left images, final disparity maps and error maps for the “Cones” and “Teddy” image pairs from the Middlebury Stereo evaluation

Figures 6.8 and 6.9 show the left image, the final disparity map and the error map for the “Venus”, “Tsukuba”, “Map”, “Cones” and “Teddy” image pairs. The results for ”Sawtooth” appear in Fig. 6.3. White in the error maps indicates an error less than one half of a disparity level or occluded pixel, grey indicates an error between one half and one disparity level (acceptable) and black indicates large errors above one disparity level.

Results on Aerial Images The final results of this chapter are on less controlled images, taken from an airplane under sunlight. The left and right images are shown in Fig. 6.10(a) and (b). They are grayscale, with different camera gains, and contain large shadows. The same matching techniques, with the exception of the interval windows, which are not very effective



(a) Left image (b) Right image

(c) Refined surfaces (d) Final disparity map

Figure 6.10: Input aerial images, results of surface grouping and refinement, and final disparity map. The outlines of the buildings have been super-imposed manually.

for grayscale images, are used to produce the initial matches, and voting is performed to infer surface saliencies. The results after surface grouping and refinement can be seen in Fig. 6.10(c) and the final disparity map in Fig. 6.10(d). The outlines of the buildings have been super-imposed manually to show the accuracy of the reconstruction.

6.8.1 Computational Complexity

In this section, we provide an analysis of the computational complexity of each step as a function of the number of pixels N, the number of possible disparities D and the number of pixels included in the matching windows W. Processing times for un-optimized C++ code refer to a Pentium 4 processor at 2.8MHZ.

In the initial matching stage, the complexity is linear with respect to the number of pixels, possible disparities and the window size, since the score or cost is computed over all pixels of the window for each disparity of each pixel of the reference image. Complexity is $O(NDW)$. Keep in mind, however, that more efficient implementations to avoid repeating computations are possible. For instance, Veksler [162] proposes a method for computing matching costs that is independent of the window size by using the integral image. The execution time for the 5×5 cross-correlation window on the Tsukuba image, which is 384×288 with 20 possible disparity levels, is 4 seconds. Execution times for other techniques are similar and scale linearly with the three parameters.

The complexity of tensor voting stage is $O(NlogN)$, since there is at most a small fixed number of matches for each pixel. Tensor voting is only performed on candidate matches and thus is independent of the number of possible disparities. It takes 2 minutes and 30 seconds for 517819 matching candidates for the Tsukuba image pair.

Uniqueness enforcement is performed at the pixel level and is virtually instantaneous. Surface grouping is linear in the number of pixels, since it is performed as a single pass over the matching candidates, which are one or none for each pixel, and only the 8-neighbors of the corresponding pixels are examined. Subsequent operations, such as the rejection of groups with too few members, are linear in the number of groups, and thus negligible. During surface refinement, pixels within the neighborhood of the re-projections of the grouped matching candidates on both images are examined. The process takes 6 seconds for the Tsukuba image pair with a neighborhood radius of 18 pixels. All these steps are $O(N)$ and take very few seconds, dominated by the surface refinement step.

Finally, disparity estimation for unmatched pixels is linear in the number of unmatched pixels, which, typically, are a small subset of all pixels, as well as the number of allowable disparities for each of them, as indicated by their neighbors that belong to the most similar surface. Complexity for the worst case is $O(NDlogN)$. The processing time for 24551 unmatched pixels of the Tsukuba is 5 minutes and 22 seconds.

6.9 Discussion

We have presented a novel stereo algorithm that addresses the limitations of binocular matching by incorporating monocular information. We use tensor voting to infer surface saliency and use it as a criterion for deciding on the correctness of matches as in [78] and [79]. However, the quality of the experimental results depends heavily on the inputs to the voting process. The superior performance of the new algorithm is due to the flexible initial matching stage and the combination of the geometric and photometric consistency we enforce on the surfaces. Textured pixels away from depth discontinuities can be easily resolved by even naive stereo algorithms. As stated in the introduction, we aimed at reducing the errors at untextured parts of the image and near depth discontinuities which cause occlusion. Under our approach, the typical phenomenon of the over-extension of foreground surfaces over occluded pixels is mitigated by removing from the dataset candidate matches that are not consistent with their neighboring pixels in both images. On the other hand, surface smoothness is the main factor that guides the matching of uniform pixels.

Arguably the most significant contribution is the segmentation into layers based on geometric properties and not appearance. We claim that this is advantageous over other methods that

use color-based segmentation, since it utilizes the already computed disparities which are powerful cues for grouping. In fact, grouping candidate matches in 3-D based on good continuation is a considerably easier problem than image segmentation. This scheme allows us to treat both images symmetrically and provides estimates for the layer color distribution even if it varies significantly throughout the layer. The choice of a local, non-parametric color representation allows us to handle surfaces with texture or heterogeneous and varying color distributions, such as the ones in the "Venus" images, on which image segmentation may be hard. This representation is used at the layer refinement stage to eliminate mostly foreground over-extensions.

A second significant contribution is the initial matching stage that allows the integration of any matching technique without any modification to subsequent modules. The use of a large number of matching operators, applied to both images, can be viewed as another form of consensus. While all operators fail for certain pixels, the same failures are usually not repeated, with the same disparity values, by other operators. Our experiments show that the results of combining the four techniques we used over all the image pairs are superior to those generated by using a smaller set of them. Even though a particular matching technique may produce systematic errors for a particular image pair, its inclusion is beneficial when all six image pairs are considered.

Adhering to the principles set out in Chapter 1, we employ a lest commitment strategy and avoid the use of constraints that are violated by usual scene configurations. One such constraint is the requirement that adjacent pixels should have the same disparity to avoid incurring some penalty. While this constraint aids the optimization process of many approaches, it becomes an approximation for scenes that do not consist of fronto-parallel surfaces. This is the most

likely reason that not many results for the "Cones" and "Teddy" image pairs are published. Processing in 3-D via tensor voting enforces the more general constraint of good continuation and eliminates interference between adjacent pixels from different world surfaces without having to assess penalties on them. In our work, the assumption that scene surfaces are frontoparallel is only made in the initial matching stage, when all pixels in a small window are assumed to have the same disparity. After this point, the surfaces are never assumed to be anything other than continuous. We also do not use the ordering constraint, which was introduced to facilitate dynamic programming. The uniqueness constraint is applied cautiously, since it is viewpoint dependant and its results do not hold for the target image.

Our algorithm fails when surfaces are entirely missed at the initial matching stage or when they are entirely removed at the layer refinement stage. We are not able to grow surfaces that are not included in the data before the final stage. On the other hand, we are able to smoothly extend partially visible surfaces to infer the disparities of occluded pixels, assuming that occluded surfaces do not abruptly change orientation. The major limitation of our work is that one cannot predict the usefulness of intermediate results based on the error rate. "Cleaner" datasets after layer refinement may not contain enough information to guide correct disparity estimation for the unmatched pixels. In our future work, we intend to derive a set of criteria that adapt the refinement stage according to both surface orientation and color distribution of the layers.

Chapter 7

Dense Multiple View Stereo with General Camera Placement

In this chapter, we address the problem of multiple view stereo from a perceptual organization perspective. Our methodology, at various stages of its evolution, has been published in [101–103]. As in the binocular case, the premise is that correct pixel correspondences align to form the scene surfaces which are more salient than any potential alignments of wrong matches. The main contribution in terms of the problem at hand is a computational framework for the inference of dense descriptions from multiple view stereo with a far wider range of permissible camera configurations. Thus far, research on dense multiple view stereo has evolved along three axes: computation of scene approximations in the form of visual hulls; merging of depth maps derived from simple configurations, such as binocular or trinocular; and multiple view stereo with restricted camera placement. These approaches are either sub-optimal, since they do not

maximize the use of available information, or cannot be applied to general camera configurations. Our approach does not involve binocular processing other than the detection of tentative pixel correspondences. We require calibration information for all cameras and that there exist camera pairs on which automatic pixel matching can be performed effectively. The inference of scene surfaces is based on the premise that correct pixel correspondences, reconstructed in 3-D, form salient, coherent surfaces, while wrong correspondences form less coherent structures. The tensor voting framework is suitable for this task since it can process the very large datasets we generate with reasonable computational complexity. We show results on real images that present numerous challenges.

7.1 Introduction

Reconstruction of three-dimensional scenes from sets of images is a fundamental problem in computer vision. The minimum number of images required for reconstruction, be it projective or metric, is two. The major challenge is the establishment of pixel correspondences, which is a core problem that has received extensive attention since the early days of computer vision. Even though one might justifiably argue that the core issues of stereo vision are addressed in the binocular case, the generalization to multiple images is not straightforward. Here by "multiple images or views" we always refer to cases where the number of images is greater than two.

The additional difficulties in the multiple image case stem from insufficiencies of either the representation or the computational framework. View-centered representations are inadequate for general camera placement, while the computational complexity associated with processing a possibly very large number of pixels is prohibitive for certain methodologies. To bypass these

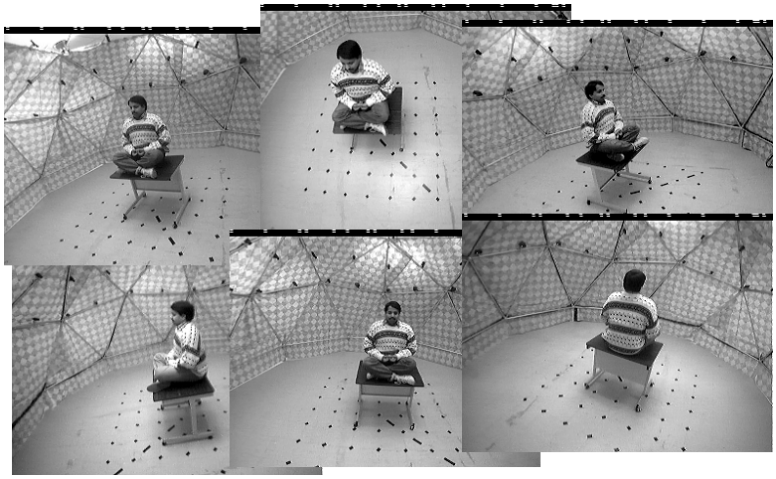


Figure 7.1: Some of the input images of the "meditation" set captured at the CMU dome. Some of the cameras are visible in each image.

difficulties, researchers have taken different paths. One such path is requiring all cameras to be "on the same side" of the scene. This class of approaches includes multi-baseline stereo and other approaches that have one or more privileged images, for which a depth map is computed and validated using the remaining images. A $2\frac{1}{2}$-D, view-centered representation is sufficient for such configurations. A different approach is to adopt a world-centered representation but restrict all processing to a limited number of features to keep computational complexity manageable. Once the structure of the feature points has been estimated, dense stereo is performed on image pairs. Conversely, other researchers compute depth maps from image pairs and merge the results. Both these approaches do not utilize all the images where a world point appears at the same time. Recently, a considerable amount of work has been devoted to computing scene approximations, instead of reconstructions, in the form of "visual hulls", inspired by Laurentini [73] and by Seitz and Dyer [142]. Section 7.2 contains a more extensive review of multiple-view reconstruction methods.

We present an approach that allows truly general camera placement, employs a world-centered representation and is able to process large numbers of potential pixel matches, typically in the order of a few millions, efficiently. No images are privileged and features are not required to appear in more than two views. The only restriction on camera placement is that cameras must be placed in pairs in such a way that, for each camera, there exists at least another one with a similar viewpoint that allows for automatic correlation-based dense pixel matching. One could place such cameras pairs arbitrarily in space with no other considerations for image overlap or relationships between the locations of camera centers and the scene. Moreover, unlike other leading multiple view reconstruction methods [28,58,69,72,175], we do not segment and discard the "background" but attempt to reconstruct it together with the foreground. Camera calibration information and a common world coordinate frame, however, has to be provided. We cast the problem as one of perceptual organization of potential pixel matches reconstructed in 3-D. Tensor voting provides an effective, computationally efficient means for achieving this type of perceptual organization even under severe noise contamination. We claim that processing all the available information at the same time can produce superior results since it does not suffer from inaccuracies that inevitably occur at the borders of partial surfaces and, moreover, since correct matches from nearby image pairs reinforce each other, thus making the correct surfaces even more salient.

We present results on challenging datasets captured for the Virtualized Reality project of the Robotics Institute of Carnegie Mellon University and distributed freely at http://www-2.cs.cmu.edu/virtualized-reality. Besides the people that appear in the images, we also reconstruct the floor and the visible parts of the dome. To our knowledge, no static reconstructions

of these scenes have been published, even thought the data have been available for a few years. Some of the input images can be seen in Fig. 7.1. The outputs presented in these chapter are in the form of 3-D point clouds. Inference of surfaces from such data is out of the scope of this chapter and can be addressed as in [31, 152].

The chapter is organized as follows. Related work is presented in the next section, an overview of the approach is presented in Section 7.3, the initial processing steps in Section 7.4, and the application of tensor voting on the candidate matches in Section 7.5. Experimental results are shown in Section 7.6, a summary and perspectives are given in Section 7.7.

7.2 Related Work

Early research on multi-baseline stereo includes the work of Okutomi and Kanade [113] who use both short and long baselines, which offer different advantages, from a set of cameras perfectly placed on a straight line with parallel optical axes. A layer based approach that can handle occlusion and transparency was presented by Szeliski and Golland [149]. The assumption is that the cameras are sufficiently far away from the scene so that it can be represented as a set of planes in 3-D. The number of these layers, however, is hard to estimate a priori and initialization of the algorithm is difficult. Kang *et al.* [63] advance the state of the art in multi-baseline stereo by taking occlusion into account when selecting the subset of images in which each pixel can be matched. The representation in all these approaches is view-centered and dense disparity maps are produced for one or more privileged images.

Approaches that employ 3-D representations are also numerous. They fall into two categories: those that reconstruct sparse features initially and then use binocular stereo to produce

dense depth estimates; and those that merge the view-centered depth maps that have been produced from binocular or multi-baseline stereo. A landmark approach of the former category is the work of Pollefeys *et al.* [118] that performs self-calibration of the cameras, even in the case of varying internal parameters, based on a set of sparse features and then uses binocular matching to complete the models. Lhuillier and Quan [81] increase the robustness of scene and camera geometry estimation by using "quasi-dense" instead of sparse features. Then, the results from image pairs and triplets are merged to produce the final reconstruction. Other approaches the fall in the category of merging depth maps, include the work of Fua [31], Narayanan *et al.* [108] and Taylor [156]. We argue that even though these schemes are effective, the fact that they do not utilize all the available information during binocular processing does not allow them to achieve the best possible quality in the produced depth maps, and thus the merged surfaces are also not as good as possible.

Recently, volumetric multiple view methods have attracted a lot of attention from the computer vision community. Kutulakos and Seitz [72] introduced the space carving framework which is based on the notion of "photoconistency" to derive the visual hull of the scene, which is an approximation of shape, from a set of images. Voxels from an initial volume, which must contain the scene, are progressively carved away if their projections on the images are not photoconsistent, that is if they exhibit large color variations. This indicates that the voxel under consideration is empty and the projections come from voxels occluded by it which are revealed once it is removed from the volume. The formulation of [72] alleviates the camera placement restrictions of [142] and the only assumption is that of "free space", which states that

the photoconsistent scene must completely lie within an arbitrary volume, which must not contain the optical centers of the cameras. A limitation of the original space carving algorithm is that it cannot recover from the erroneous carving of a voxel which leads to further carving into the volume since photoconsistency is violated by voxels behind the actual surface. Techniques that have been proposed to address this problem can be found in recent surveys of volumetric methods such as [25] and [146].

Outstanding results on binocular stereo have been achieved using graph cuts. Kolmogorov and Zabih [69] extended the framework to multiple images. The problem is formulated as finding the approximate minimum of a global energy function via graph cuts. The proposed framework treats all images equally, takes visibility into account and most importantly enforces *smoothness* to the solution while preserving boundaries. Lack of smoothness of the reconstructed surfaces is a critical limitation of the volumetric methods of the previous paragraph, which operate at the pixel-voxel correspondence level without imposing any form of coherence in the resulting surfaces. While the cameras can be placed according to the restrictions of voxel coloring [142], results are only presented for examples in which all the cameras lie on one semiplane looking at the other semiplane. Furthermore, computational complexity is quadratic with respect to the number of possible labels (depth layers), which has to be kept low (16 in the experiments presented in [69]).

Variational approaches have also been applied to multiple-view stereo and have achieved excellent results. Faugeras and Keriven [28] pose multiple view stereo in a variational framework where an initial surface evolves according to image correspondence criteria. Cross-correlation

on the tangent plane of the surface, instead of regular square windows which imply fronto-parallel surfaces, is used as the similarity measure. Yezzi and Soatto [175] address a different type of scenes in which surfaces have smooth or constant albedo, or fine homogeneous texture. To avoid the difficulties associated with the computation of image derivatives, they do not rely on local correspondence, but on region similarity measures which are very effective for the types of objects they handle. Jin *et al.* [58] extend the framework to handle any type of surface including non Lambertian ones by imposing a rank constraint on the radiance tensor. They achieve excellent results, but are limited, as are the previous two approaches, by the "blue sky" assumption [175] regarding the background which must be segmented and discarded.

Our work differs in that camera pairs can be arbitrarily placed, as long as calibration information is provided or computed as in [118], without any requirements for free space or blue sky type background. The proposed approach utilizes all available information simultaneously for reconstruction and does not approximate the scene surfaces. It is important to note that we do not make any decisions about pixel correspondence at the binocular level. We only make hypotheses for potential correspondences.

7.3 Overview of the Approach

The input to our algorithm is a set of images of a static scene and complete calibration information (both internal and external parameters). The output consists of a set of 3-D points that belong on the scene surfaces. Processing entails the following steps:

- selection of image pairs and rectification to align conjugate epipolar lines

- initial binocular matching to generate matching candidates
- reconstruction of matching candidates in 3-D
- tensor voting on the point cloud to infer surface inliers.

Before tensor voting takes place, each matching candidate is reconstructed as a 3-D point and encoded as a second order stick tensor. We propose to use surface saliency ($\lambda_1 - \lambda_2$) as the criterion for the selection of correct matches among the multiple candidates on each line of sight.

7.4 Initial Processing Steps

In this section we describe the processing stages that precede the main tensor voting stage.

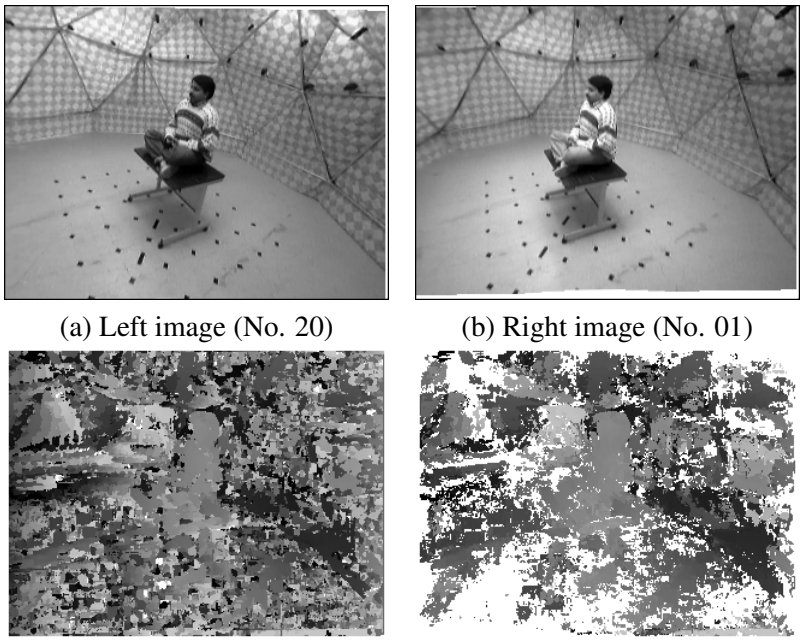
Image pair selection and rectification As can be seen in Fig. 7.1, the input images suffer from severe radial distortion. The first step, therefore, is to correct the distortion using the provided κ_1 coefficient. The next step is the selection of image pairs taken from similar viewpoints. Only when the images are taken from a similar angle and a similar distance is automatic pixel matching possible. Otherwise perspective effects make the matching process really hard. This is why we need pairs of images from similar viewpoints to proceed. Once the image pairs have been selected (manually, even though automatic selection would not have been very complicated to implement), the images are rectified in pairs so that all epipolar lines in both images become horizontal and conjugate epipolar lines share the same row coordinate. This is accomplished using the calibration information, but no correspondences, by our implementation of the

work of Gluckman and Nayar [35]. An example pair can be seen in Fig. 7.2(a-b). Note that the effects of radial distortion have not been entirely removed close to the image boundaries, and that in these areas corresponding features do not lie on conjugate epipolar lines. Each image can be used in more than one pair, but rectification has to be performed separately for each pair.

Generation of matching candidates The input to this stage is rectified image pairs. The output is matching candidates reconstructed in 3-D world coordinates, labeled according to the line of sight (going through the optical center and the image pixel) they belong to. It can be viewed as binocular processing, even though no decisions are made and the objective is the inference of potential 3-D points that belong to the scene.

Matching candidates are generated for each image pixel of the "left" or reference image of each pair by a simple 7×7 normalized cross-correlation window. We apply a simple parabolic fit using the neighboring values of peaks in the correlation function to achieve subpixel precision. All significant peaks of cross-correlation are retained as matching candidates. As a result, more than one candidate can be generated for each pixel. Figure 7.2(c) shows a disparity map of the matching candidates, where candidates for the same pixel with larger disparity are written on top of ones with smaller disparity, for one pair of the "meditation" set.

Figure 7.2(d) shows an attempt to process the pair in the same way we process datasets generated by multiple pairs in the remainder of the chapter. The results are not satisfactory since wrong matches caused by occlusion (around the head of the person for instance), misalignment due to radial distortion (close to the borders) and lack of texture (on the floor) cannot be discriminated from the correct ones. However, when processing is done using all matching candidates



(a) Left image (No. 20) (b) Right image (No. 01)

(c) Matching candidates in disparity space

(d) Disparity map after binocular processing

Figure 7.2: Two images of a rectified pair (a-b), the generated matching candidates in disparity space (c) and an attempt to produce a disparity map with binocular processing (d) (lighter intensity corresponds to larger disparity, white indicates no match).

generated from all image pairs, consistent matches generated from different image pairs align to form surfaces that stand out among the clutter. We consider this a significant advantage of processing all data simultaneously over merging inaccurate depth maps produced binocularly.

7.5 Tensor Voting on Candidate Pixel Correspondences

The input to the tensor voting stage is a cloud of points (Fig. 7.3) and the desired output is a subset of those points that includes the inliers of the most salient surfaces. Since we need to process over one million points in the examples presented here, the fact that tensor voting

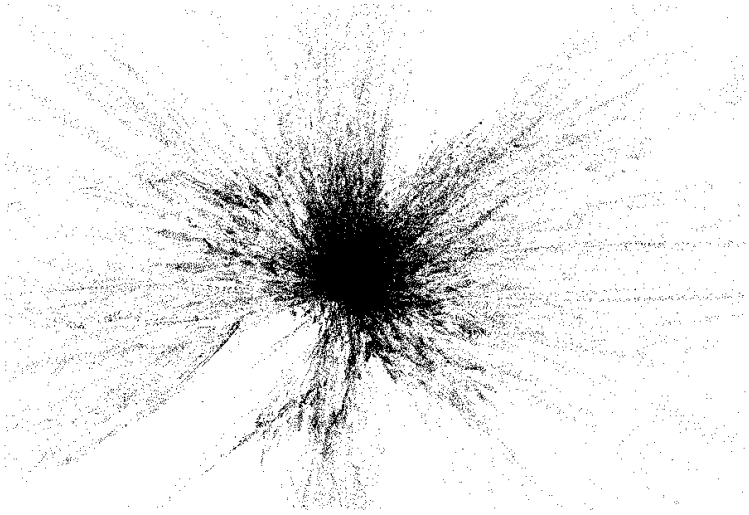


Figure 7.3: A view from above of all matching candidates (a little over one million) of the "meditation" set reconstructed in world coordinates. Notice the rays of matching candidates emanating from the cameras that are contained in the dome (close to the center of the point cloud).

operates in neighborhoods of fixed size is beneficial, since the number of votes cast by each point is a function of the size of the neighborhood and thus complexity is not quadratic.

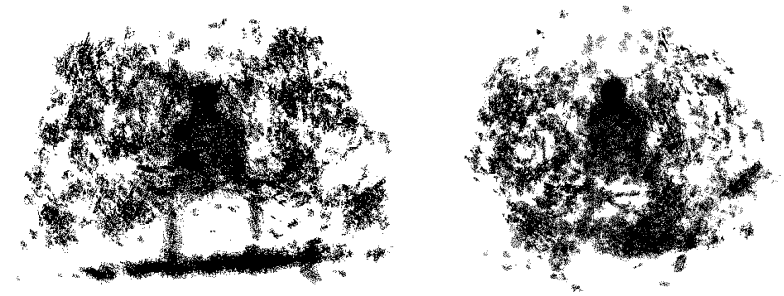
Taking visibility into account, we initialize the tensor at each point as an oriented surfel with a stick tensor pointing to the midpoint of the optical centers of the cameras that generated the matching candidate. This orientation estimate, that can be corrected after voting, assures that the point supports surfaces that could be visible from the appropriate cameras. All points cast votes to their neighbors in 3-D, regardless of which images were used to generate each point, thus combining all information in one processing stage. Analysis of the accumulated tensors after voting produces corrected surface normal estimates and saliency values based on the support each point receives.

Then, uniqueness with respect to the lines of sight is enforced, leaving one candidate for every image pixel. The criterion is maximum surface saliency. Therefore, for each pixel of all images, the match that receives the maximum support as an inlier of a salient surface, remains in the dataset. This step is not enough to guarantee the removal of all wrong matches, since we cannot assume that the matching candidates for some pixels, especially the ones in distorted or textureless areas, include the correct match. The remaining wrong matches can be rejected based on their low surface saliency, since they do not form coherent surfaces and, therefore, do not receive significant support from their neighbors as surface inliers. Simple thresholding with respect to average surface saliency is enough to produce datasets with mostly correct surface points, as shown in the next section.

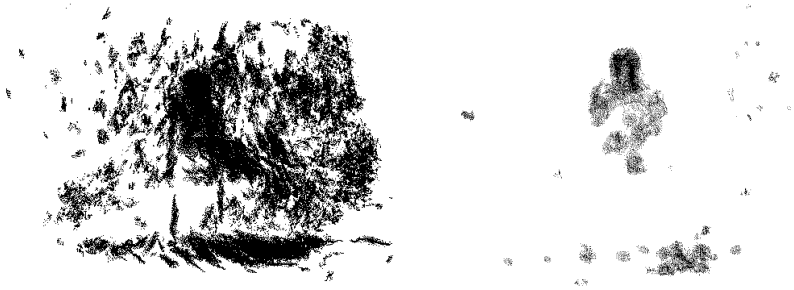
7.6 Experimental Results

For the experiments presented in this section, we used two image sets captured at the CMU dome. Both the "meditation" and "baseball" sets were captured with identical settings and the results we present were produced with the same parameters. The input consists of the same 10 image pairs using a total of 17 images that were selected among the 51 total images based on viewpoint similarity to facilitate automatic matching. The images are 320×245 with 8 bits of grayscale values. Besides the distortion issues that were mentioned above, there is also motion blur in the "baseball" set (Fig. 7.5(a-b)).

Matching candidates are generated using a 7×7 correlation window as in Section 7.4. Nine pairs are processed with the same disparity range (-40 to 40 for "meditation" and -20 to 55 for "baseball") and the last pair, which is elevated, is processed using 20 to 75 for the disparity



(a) All reconstructed points (b) 45^o view of all points



(c) Side view of all points (d) Most salient surface inliers

(e) Points at the center of the dome (f) Rotated view of center

Figure 7.4: Results for the "meditation" set. Three views of all the reconstructed points with surface saliency at least twice its average value (a-c). A view of the points with surface saliency at least 12 times the average, which are mostly on the person and the floor (d); and two views of only the center of the entire dataset (e-f).

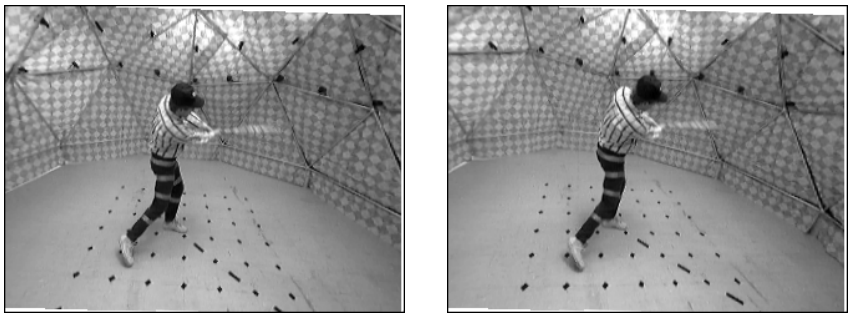
range. The only negative effect of a large disparity range is an increase in processing time for the generation of matching candidates.

The matching candidates are reconstructed in 3-D by triangulating the rays that go through the two pixels and the optical centers of the cameras. Given the poor image resolution, quantization noise is significant and the localization of the reconstructed points is not perfect. They do, however, form surfaces that are close to the true scene surfaces. We are able to infer these surfaces from the point cloud after tensor voting since the points they comprise receive more support than the outliers. Unfortunately, we do not have ground truth data to compare our results to. We use $4mm$ as the unit of distance in the world coordinate system and perform tensor voting with $\sigma^2 = 500$, which corresponds to a voting radius of 48 units of distance.

The execution times of the un-optimized implementation of our code on a Pentium IV at 2.8GHz for the "meditation" set are the following:

- The generation of matching candidates using a 7×7 normalized cross-correlation window and searching 80 possible disparity levels takes 16 seconds.
- The reconstruction of all points in 3-D takes 1 minute and 21 seconds.
- Tensor voting and analysis of the results for 1,126,554 points with $\sigma^2 = 500$ takes 44 minutes and 30 seconds. (In the case of "baseball" tensor voting for 1,117,920 points takes 40 minutes and 2 seconds.)

Due to the nature of the scenes, visualization of the results is hard. Besides the entire output which includes the person, the visible parts of the dome and the floor, we have also included in Figs. 7.4 and 7.5 views of the center of the data only, where the person is, as well as of the topmost salient surfaces, which again tend to be those of the person where maximum point concentration occurs since they are visible from a larger number of cameras.



(a) Rectified image (No. 17)

(b) Rectified image (No. 16)

(c) All reconstructed points

(d) Rotated view

(e) Points at the center of the dome

(f) Rotated view of center

Figure 7.5: Results for the "baseball" set. One of the ten pairs used (a-b). Two views of all the reconstructed points with surface saliency at least twice the average value (c-d); and two views of only the center of the dataset (e-f).

7.7 Discussion

We have presented an approach for multiple view reconstruction that can deal with scene and camera configurations that cannot be handled by current state of the art algorithms. The typical

"blue sky" and "free space" assumptions are not required, with the only constraint on camera placement being that viewpoints must be similar enough to allow automatic pixel matching. The tensor voting computational framework allows the simultaneous processing of all matching candidates, utilizing all the available images to a far greater extend than approaches based on the fusion of binocular results. The computational cost is reasonable since all processing is local and distant points do not affect each other. Methods with quadratic complexity with respect to the number of pixels would severely stretch computational resources in examples with close to one million pixels, such as the ones presented here. We have strived to maintain a high level of generality. All images are treated equally and there are no privileged views for which depth maps are computed. The representation is fully three-dimensional and there are no view-dependent elements.

Satisfactory results can be produced even with low resolution and low quality images. This is due to the fact that features are not required to appear in more than two views. Given the poor resolution of the available images (320×245), matching a feature in multiple views is hard. However, since features are validated by the support the receive as surface inliers in the form of surface saliency, the problem is transformed from validation at the point level to validation at the surface level. Therefore, as long as a matching candidate can be reconstructed in 3-D sufficiently close to the true surface, exact point correspondence in multiple images is not required. Thus, we are able to achieve reconstructions at a resolution higher than what would have been possible with methods such as [118] that require point correspondences in multiple frames or methods based on space carving [72] that require exact pixel-voxel correspondences.

Processing all data at the same time maximizes performance at occluded and textureless regions where binocular stereo typically encounters difficulties. The "fattening" of occluding surfaces, which is a problem in the binocular case, is diminished in the multiple view case because parts of both the occluding and occluded surfaces that are invisible in a particular image can be seen in different views and thus produce correct matches that outweigh the wrong ones. Errors in textureless regions that are caused by the ambiguity of matching are less severe when matching candidates from more than one pair are combined. Then, the correct surface is more likely to be formed, even in the midst of a large number of outliers. The same is not guaranteed to happen in the binocular case with the same error rate in the matching stage. Figure 7.2(d) shows a disparity map for one pair of the "meditation" set that was processed the same way as the entire dataset. We do not by any means claim that this is the best result that can be achieved for this pair, but it is worth noticing the number of errors that occur in occluded and textureless areas which are eliminated when all matching candidates are processed simultaneously. Fusion of binocular reconstructions cannot be more effective than simultaneous processing of all available information.

The proposed approach fails when the assumption that correct matches form more salient surfaces than incorrect ones does not hold. This is the case when systematic errors in the initial matching stage occur for some reason and the algorithm "hallucinates" non-existent surfaces. This is a weakness of all perceptual organization frameworks, but does not occur often. The noise that appears in the results presented in the previous section may be eliminated by another pass of tensor voting or during the surface extraction process, but this is beyond the scope of this work.

Chapter 8

Tensor Voting in High-Dimensional Spaces

In this chapter we describe the second major contribution to the tensor voting framework, besides the addition of first order information. Even though this work is very recent, we feel that tensor voting in high dimensional spaces may turn out to be the most important contribution of our research. The motivation comes from the observation that many problems, from a broad range of scientific domains, can be addressed within the tensor voting framework. In most cases, addressing these problems requires efficient, local, data-driven algorithms with high noise robustness. The Gestalt principles of proximity, similarity and good continuation, that have been identified as the factors that make configurations in two and three dimensions salient, still apply in spaces of higher dimensionality. Therefore, the tensor voting framework, that combines many of the desired properties, seems to be a well-suited approach. The main limitation that prevents its wide application to problems in high dimensions is the exponential increase in computational

and storage demands as the dimensionality of the space grows. An algorithm for the estimation of the fundamental matrix in 8-D by Tang *et al.* [153] is the only successful application of tensor voting in spaces with more than four dimensions.

8.1 Introduction

The tensor voting framework, in its preliminary version [42], is an attempt at the implementation of two Gestalt principles, namely proximity and good continuation, for grouping generic tokens in 2-D. The 2-D domain has always been the main focus of research in perceptual organization, beginning with the research of Köhler [67], Wertheimer [169] and Koffka [66] up to the recent work that was reviewed in Sections 2.1 and 4.2. The generalization to 3-D is straightforward, since salient groupings can be detected by the human visual system based on the same principles. Guy and Medioni extended the framework to three dimensions in [43]. Other perceptual organization approaches with 3-D implementations include [114, 126]. Their number is considerably smaller than that of 2-D methodologies, mostly due to the exponential increase in computational complexity with the dimensionality of the space. Regardless of the computational feasibility of an implementation, the same grouping principles apply to spaces with even higher dimensions. For instance, Tang *et al.* [153] observed that pixel correspondences can be viewed as points in the 8-D space of free parameters of the fundamental matrix. Correct correspondences align to form a hyperplane in that space, while wrong correspondences are randomly distributed. By applying tensor voting in 8-D, the authors were able to infer the

dominant hyperplane and the desired parameters of the fundamental matrix. This work was reformulated as a 4-D problem and solved for the case of multiple independently moving objects by Tong *et al.* [160].

Even though the applicability of tensor voting as an unsupervised learning technique in high-dimensional spaces seems to have merit, a general implementation is not practical, mostly due to computational complexity and storage requirements in N dimensions. The storage requirements for each token are $O(N \times N)$, which is acceptable. It can be reduced with a sparse storage scheme, but that would increase the necessary computations to retrieve information. The bottleneck is the generation and storage of first and second order voting fields, the number of which is equal to the dimensionality of the space. For instance, a second order voting field in 10-D with k samples per axis, requires storage for 10^k $N \times N$ tensors, which need to be computed via numerical integration over 10 variables. Moreover, the likelihood of each pre-computed vote being used decreases with the dimensionality. Thus, the pre-computed voting fields soon become impractical as dimensionality grows. At the same time, the computation of votes "on the fly" as they become necessary is also computationally expensive. Here, we propose a simplified vote generation scheme that bypasses the computation of uncertainty and allows the generation of votes from arbitrary tensors in arbitrary dimensions with a computational cost that is linear with respect to the dimensionality of the space. Storage requirements are limited to storing the tensors at each token since voting fields are not used any more. Secondary contributions described in this chapter include the use of the Approximate Nearest Neighbor (ANN) k-d tree data structure of Arya *et al.* [2] for storing and retrieving the tokens.

This chapter is organized as follows: the next section points out the limitations of the original implementation of tensor voting; Section 8.3 describes the new voting scheme including a comparison with the original implementation; Section 8.5 summarizes the contributions of the chapter.

8.2 Limitations of Original Implementation

The major limitations of the original implementation of tensor voting that render its generalization to N-D impractical are related to the use of pre-computed voting fields. The difficulties are due to the lack of a closed form solution for the integrals required to compute votes cast by tensors that are not purely sticks. For example, the computation of a vote cast by a plate tensor in 3-D, as shown in Eq. 2.11, requires the integration of the votes cast by a rotating stick tensor over two angles that span 360 degrees. Approximating the computation by a summation with one degree steps results in 129,600 stick vote computations. In general, the computation of a vote cast by a tensor with N normals requires $N - 1$ nested summations to span the normal space and 360^{N-1} stick vote computations. An additional problem is caused by the fact that this simple sampling scheme of constant angular steps does not produce a uniform sampling in polar coordinates, if more than one summation is necessary. More uniform samplings are possible, especially if one aligns the poles of the hyper-sphere with the non-voting direction from the voter to receiver. A uniform sampling improves the accuracy, but does not reduce computational complexity.

In terms of storage, one can exploit symmetries, such as that the stick field is essentially 2-D, regardless of the dimensionality of the space, and that the ball field is 1-D, since it is a function

of distance only, but space requirements make the storage of the voting fields impossible even for relatively modest values of N. A lesser weakness of pre-computed voting fields is the fact that votes at non-grid positions are not exact, since they are produced by linear interpolation between grid positions.

Another limitation of the previous implementation is the use of a multiple level priority queue for data storage. The weaknesses of this data structure is that search operations are optimal if the axes have been assigned optimal priority. Otherwise, search may degenerate to an $O(M)$ operation, where M is the number of tokens. This becomes more pronounced as dimensionality increases.

8.3 Tensor Voting in High Dimensional Spaces

In this section, we describe in detail the new vote generation scheme. We adopt the data representation and vote analysis of [92] and the previous chapters, but use the ANN k-d tree of [2] as the data structure that stores the data and retrieves the neighbors. Data representation and vote analysis are also presented in this chapter for completeness, as well as to illustrate the differences between 2- and 3-D and high-dimensional spaces. Note that efficiency with this formulation is considerably higher than our initial attempt in [106].

8.3.1 Data Representation

The representation of a point is still a second order, symmetric, non-negative definite tensor. (We chose not to describe the first order part here for clarity.) It can represent the structure of a manifold going through the point by encoding the normals to the manifold as eigenvectors of the

tensor that correspond to nonzero eigenvalues, and the tangents as eigenvectors that correspond to zero eigenvalues. A point in an N-D hyperplane has one normal and $N-1$ tangents, and thus is represented by a tensor with one nonzero eigenvalue associated with an eigenvector parallel to the hyperplane's normal. The remaining $N-1$ eigenvalues are zero. A point in a 2-D manifold in N-D has two tangents and $N-2$ normals, and thus is represented by a tensor with two zero eigenvalues associated with eigenvectors that span the tangent space of the manifold. The tensor also has $N-2$ nonzero eigenvalues (typically set to 1) whose corresponding eigenvectors span the manifold's normal space.

The tensors can be formed by the summation of the direct products ($\vec{n}\vec{n}^T$) of the eigenvectors that span the normal space of the manifold. The tensor at a point on a manifold of dimensionality d, with $\vec{n}_i$ spanning the normal space, can be computed as follows:

$$T = \sum_{i=1}^{d} \vec{n}_i \vec{n}_i^T \tag{8.1}$$

As before, a ball tensor is viewed as one having all possible normals and is encoded as the $N \times N$ identity matrix. Any point on a manifold of known dimensionality and orientation can be encoded in this representation by appropriately constructed tensors, as in Eq. 8.1.

On the other hand, given an N-D second order, symmetric, non-negative definite tensor, the type of structure encoded in it can be found by examining its eigensystem. Any tensor that has these properties can be decomposed as in the following equation:

$$\begin{aligned}\mathbf{T} &= \sum_{d=1}^{N} \lambda_d \hat{e}_d \hat{e}_d^T \\ &= (\lambda_1 - \lambda_2)\hat{e}_1\hat{e}_1^T + (\lambda_2 - \lambda_3)(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T) + ... + \lambda_N(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T + ... + \hat{e}_N\hat{e}_N^T) \\ &= \sum_{d=1}^{N-1}[(\lambda_d - \lambda_{d+1})\sum_{k=1}^{d}\hat{e}_d\hat{e}_d^T] + \lambda_N(\hat{e}_1\hat{e}_1^T + ... + \hat{e}_N\hat{e}_N^T) \quad (8.2)\end{aligned}$$

where λ_d are the eigenvalues in descending order and $\hat{e}_d$ are the corresponding eigenvectors. The tensor simultaneously encodes *all* possible types of structure. The saliency of the type that has d normals is encoded in the difference $\lambda_d - \lambda_{d+1}$. If a hard decision on the dimensionality is required, we assign the point to the type with the maximum confidence.

8.3.2 The Voting Process

In this section, we describe a novel vote generation scheme that does not require integration. As in the original formulation, the eigenstructure of the vote represents the normal and tangent spaces that the receiver would have, if the voter and receiver belong in the same smooth structure. What is missing is the uncertainty in each vote that resulted from the accumulation of the votes cast by the rotating stick tensors during the computation of the voting fields. The new votes are cast directly from the voter to the receiver and are not retrieved from pre-computed voting fields. They have perfect certainty in the information they convey. The uncertainty now comes only from the accumulation of votes from different votes at each token.

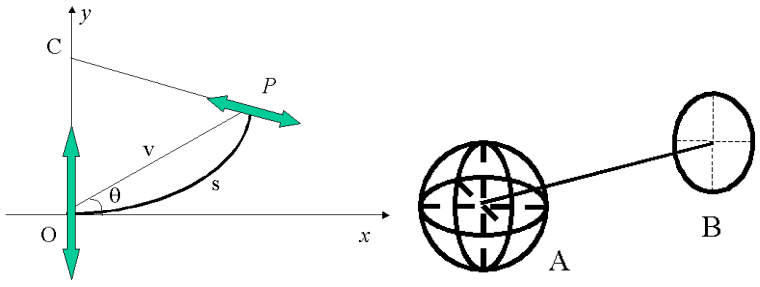
We begin by examining the case of a voter that is associated with a stick tensor of unit length, which is identical with the original formulation since it can be directly computed. The vote is generated according to the following equation, included here for completeness.

$$\mathbf{S}(s,\theta) = e^{-(\frac{s^2+c\kappa^2}{\sigma^2})} \begin{bmatrix} -sin(2\theta) \\ cos(2\theta) \end{bmatrix} [-sin(2\theta)\ cos(2\theta)]$$

$$s = \frac{\theta\|\vec{v}\|}{sin(\theta)}, \qquad \kappa = \frac{2sin(\theta)}{\|\vec{v}\|} \tag{8.3}$$

s is the length of the arc between the voter and receiver, and κ is its curvature (see Fig. 2.2(a)), σ is the scale of voting, and c is a constant. No vote is generated if the angle θ is more than 45^o. Also, the field is truncated to the extend where the magnitude of the vote is more than 3% of the magnitude of the voter.

Regarding the generation of ball votes, we propose the following direct computation. It is based on the observation that the vote generated by a ball voter propagates the voter's preference



(a) Stick voting (b) Ball voting

Figure 8.1: Vote generation for a stick and a ball voter. The votes are functions of the position of the voter A and receiver B and the tensor of the voter.

for a straight line that connects it to the receiver (Fig. 2.2(b)). (The straight line is the simplest and smoothest continuation from a point to another point in the absence of other information.) Thus, the vote generated by a ball is a tensor that spans the $(N-1)$-D normal space of the line and has one zero eigenvalue associated with the eigenvector that is parallel to the line. Its magnitude is a function of the distance between the two points, since curvature is zero. Taking these observations into account, the ball vote can be constructed by subtracting the direct product of the tangent vector from a full rank tensor with equal eigenvalues (i.e. the identity matrix). The resulting tensor is attenuated by the same Gaussian weight according to the distance between the voter and the receiver.

$$\mathbf{B}(s,\theta) = e^{-(\frac{s^2}{\sigma^2})}(\mathcal{I} - \frac{\vec{v}\vec{v}^T}{\|\vec{v}\vec{v}^T\|}) \tag{8.4}$$

Where $\vec{v}$ is a unit vector parallel to the line connecting the voter and the receiver.

To complete the description of vote generation, we need to describe the case of a tensor that has d equal eigenvalues, where d is not equal to 1 or N. (The description applies to these cases too, but we use the above direct computations, which are faster.) Let $\vec{v}$ again be the vector connecting the voting and receiving points. It can be decomposed into $\vec{v}_t$ in the tangent space of the voter and $\vec{v}_n$ in the normal space. The new vote generation process is based on the observation that curvature in Eq. 8.3 is not a factor when θ is zero, or, in other words, if the voting stick is orthogonal to $\vec{v}_n$. We can exploit this by defining a new basis for the normal space of the voter that includes $\vec{v}_n$. The new basis is computed using the Gramm-Schmidt procedure. The vote is then constructed as the tensor addition of the votes cast by stick tensors parallel to the new basis vectors. Among those votes, only the one generated by the stick tensor parallel to

$\vec{v}_n$ is not parallel to the normal space of the voter and curvature has to be considered. All other votes are a function of the length of $\vec{v}_t$ only. See Fig. 8.2 for an illustration in 3-D. Analytically, the vote is computed as the summation of d stick votes cast by the new basis of the normal space. Let N_S denote the normal space of the voter and let $\vec{b}_i, i \in [1, d]$ be a basis for it with $\vec{b}_1$ being parallel to $\vec{v}_n$. If *vote*() is the function that given a unit vector as an argument generates the stick vote from a unit stick tensor parallel to the argument to the receiver, then the vote from a generic tensor with normal space N is given by:

$$\mathbf{T} = \mathit{vote}(\vec{b}_1) + \sum_{i \in [2,d]} \mathit{vote}(\vec{b}_i) \tag{8.5}$$

In the above equation, all the terms are pure stick tensors parallel to the voters, except the first one which is affected by the curvature of the path connecting the voter and receiver and is orthogonal to it. Therefore, computation of the last $d - 1$ terms is equivalent to applying the Gaussian weight to the voting sticks and adding them at the position of the receiver. Only one vote requires a full computation of orientation and magnitude. This makes the proposed scheme computationally inexpensive.

Tensors with unequal eigenvalues are decomposed before voting according to Eq. 8.2. Then, each component votes separately and the vote is weighted by $\lambda_d - \lambda_{d+1}$, except the ball component whose vote is weighted by λ_D.

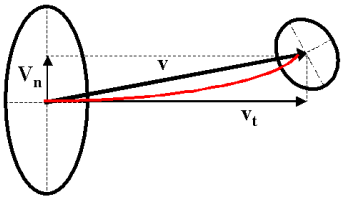


Figure 8.2: Vote generation for generic tensors. The voter here is a tensor with two normals in 3-D. The vector connecting the voter and receiver is decomposed into $\vec{v}_n$ and $\vec{v}_t$ that lie in the normal and tangent space of the voter. A new basis that includes $\vec{v}_n$ is defined for the normal space and each basis component casts a stick vote. Only the vote generated by the orientation parallel to $\vec{v}_n$ is not parallel to the normal space. Tensor addition of the stick votes produces the combined vote.

8.3.3 First Order Voting in High Dimensions

In Chapter 3, we showed the computation of the first order voting fields by integration, with the only difference being that numerically the integration was approximated by vector and not tensor summation. Here, we show how to derive the first order votes in the same way we derived the second order ones.

We consider three cases:

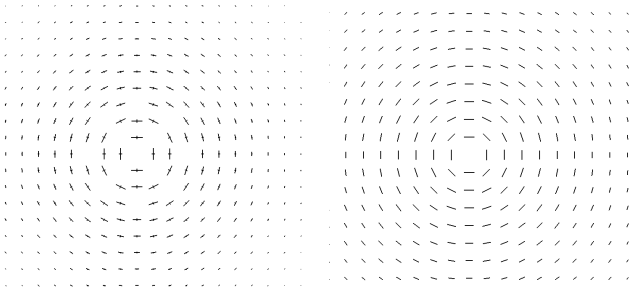
- *Stick voters* that cast first order votes that lie in the 2-D plane defined by the position and orientation of the voter and the position of the receiver, are orthogonal to the second order ones, and point towards the voter. The computation is identical to that of Eq. 3.1. Their magnitude is equal to that of the second order vote.

- *Ball voters* that cast first order votes along $\vec{v}$, which is the vector parallel to the line connecting the voter and the receiver. The first order votes point towards the voter and their magnitude is equal to the $N-1$ equal nonzero eigenvalues.

- *Arbitrary voters with d equal eigenvalues* that, unlike the second order case, require merely one direct computation. Since the first order vote has to be orthogonal to the normal space of the second order vote, and also be tangent to the circular arc connecting the voter and receiver, it can be directly computed as the first order vote cast by the stick voter $\vec{b}_1$ of Eq. 8.3.2. This vote satisfies all the requirements in terms of orientation and its magnitude decays with both the length and curvature of the arc, unless the receiver belongs to the tangent space of the voter and θ is zero.

8.3.4 Vote Analysis

Vote analysis is a direct generalization of the original formulation, with the only difference being that $N + 1$ structure types are possible in an N-D space. Each point casts a vote to all neighbors within the distance at which vote magnitude attenuates to 3% of the maximum. The votes are accumulated at each point by tensor addition. The eigensystem of the resulting tensor is computed and the tensor is decomposed as in Eq. 8.2. The estimate of local intrinsic dimensionality is given by the maximum gap in the eigenvalues. For instance, if $\lambda_1 - \lambda_2$ is the maximum difference between two successive eigenvalues, the dominant component of the tensor is the one that has one normal. Quantitative results in dimensionality estimation are presented in the next chapter. In general, if the maximum eigenvalue spread is $\lambda_d - \lambda_{d+1}$, the estimated local intrinsic dimensionality is $N - d$, and the manifold has d normals and $N - d$ tangents. Moreover, the first d eigenvectors that correspond to the largest eigenvalues are the normals to the manifold, and the remaining eigenvectors are the tangents. First order information is utilized, as in Chapter 3 to infer the boundaries of structures, the type of which



(a) Old implementation (b) New implementation

Figure 8.3: Visualization of the ball voting field using the old and the new implementation. Shown are the curve normals, as well as the tangents that represent the ball component. The ball component in the old implementation has been exaggerated for visualization purposes, while it is zero with the new implementation.

is given after analysis of the second order tensors. In the next chapter, we present quantitative results in local structure estimation, even in the presence of outliers that outnumber the inliers.

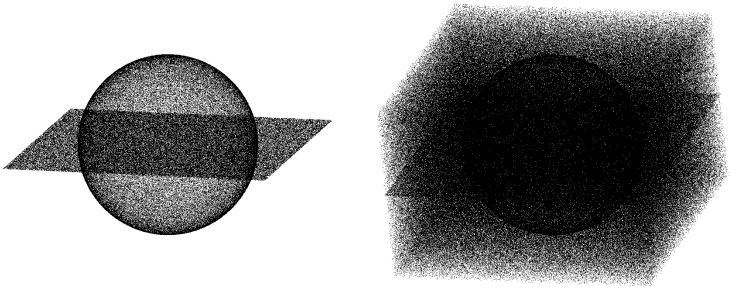
8.4 Comparison against the old Tensor Voting Implementation

In this section, we evaluate the effectiveness of the new implementation by performing the same experiments in 3-D using both the old and new vote generation schemes. Figure 8.3 shows a cut, containing the voter, of the 3-D ball voting field computed using the old and the new implementation. Shown are the projections of the eigenvectors on the selected plane after voting by a ball voter placed on the plane.

We do not attempt to directly compare vote magnitudes, since they are defined differently. In the old implementation, the total energy of the ball field is normalized to be equal to that of the stick field. In the new implementation, the magnitude of the ball vote is the same as that of the stick vote. This makes the total energy of the new field higher than that of the stick field,

since there is no attenuation due to curvature at any orientation, and voting takes place at all orientations, since the 45^o cut-off only applies to stick voters.

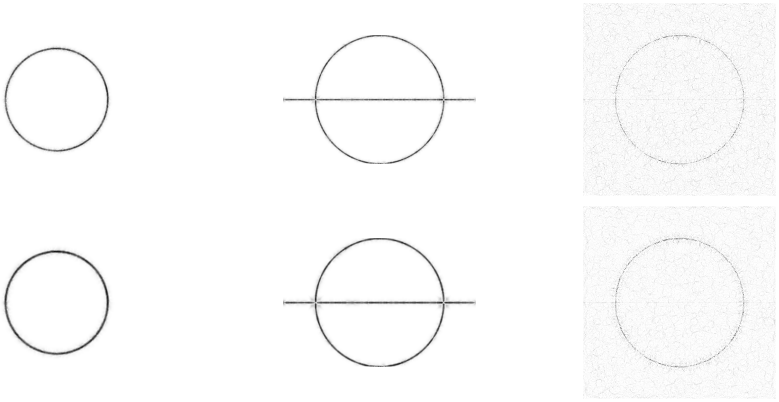
A test that captures the accuracy of the orientation conveyed by the vote is a comparison between the tangent of the ball vote and ground truth, which should be along the line connecting the voter and receiver. The old implementation was off by 1.25×10^{-5} degrees, while the new one was off by 1.38×10^{-5} degrees.



(a) Noise free (b) 4:1 outliers:inliers

Figure 8.4: Sphere and plane inputs used for comparing the old and new implementation of tensor voting

We also compared the two implementations in a simple experiment of local structure estimation. We sampled unoriented points from a sphere and a plane in 3-D, and compared the estimated surface normals against the ground truth. Note that the coordinates of the points are quantized to make the evaluation fair for the old implementation, which only operates with integer coordinates, due to the data structure that stores the tokens. The inputs, which are encoded as unit balls, can be seen in Fig. 8.4(a). Surface and curve saliency maps for horizontal and vertical cuts of the data using both implementations are shown in Fig. 8.5. The saliency maps are normalized so that their brightness covers the entire range from black to white. They are qualitatively almost identical for both cases. Quantitative results are presented in Table 8.1.



(a) Surface saliency z=120 (a) Surface saliency y=0 (a) Curve saliency z=0

Figure 8.5: Cuts of surface and curve saliency maps of the sphere and plane data. Darker areas correspond to higher saliency while white corresponds to zero. The top row was generated with the old implementation and the bottom row with the new one. The cut of the curve saliency maps contain the plane and show that the intersection of the sphere and the plane is the most salient curve in the data.

σ^2	Old TV	New TV
50	2.24	1.60
100	1.47	1.18
200	1.09	0.98
500	0.87	0.93

Table 8.1: Results on the sphere and plane dataset: average error rate in degrees for normal orientation estimation using the implementation of [92] and the one proposed here.

The accuracy in surface orientation estimation is similar in both cases, with a slight edge in favor of the new implementation. The main source of inaccuracy in the old implementation is the need for linear interpolation using the entries of the look-up table. It turns out that its effects on performance are similar to those of computing an approximate vote at the exact receiver position using the new approach. Since the difference in ball vote magnitudes becomes important only when not all voters are ball tensors, we performed a second pass of tensor voting

using the accumulated tensors form the first pass as voters. For $\sigma^2 = 100$, the error was 1.31^o with the old implementation and 1.20^o with the new one.

We, then, added noise to the data and repeated the experiment to test whether noise robustness is affected by the proposed approximation. The results are shown in the following table and are very similar to the noise-free case. A safe conclusion is that noise robustness is not compromised by the new approximation.

Outliers:Inliers	σ^2	Old TV	New TV
1:1	50	3.03	2.18
	100	2.19	1.73
	200	1.75	1.48
	500	1.44	1.38
2:1	50	3.59	2.53
	100	2.61	2.02
	200	2.10	1.74
	500	1.74	1.62
5:1	50	4.92	3.36
	100	3.59	2.71
	200	2.90	2.33
	500	2.39	2.15
8:1	50	5.98	3.98
	100	4.33	3.20
	200	3.49	2.77
	500	2.89	2.62

Table 8.2: Results on the sphere and plane dataset: average error rate in degrees for normal orientation estimation using the implementation of [92] and the one proposed here.

It should also be noted here that the efficiency benefits of the new implementation become more apparent as the dimensionality increases. The inapplicability of the original implementation to high-dimensional datasets does not allow us to demonstrate the improvement quantitatively.

8.5 Discussion

In this chapter we have presented arguably the most significant contribution of our work in terms of the tensor voting framework. It allows us to apply our methodology to problems where the properties of tensor voting, such as its noise robustness and lack of global computations, seemed appropriate and desirable, but the faithful adherence to exact vote generation made computational and storage requirements impractical. Experiments on data with ground truth show that the new approximation is equally as effective in orientation estimation, and also maintains the noise robustness of the original implementation. These results suggest that the main useful source of uncertainty in the tensors comes as a result of the tensor addition of votes at each token, and not from the uncertainty component of each vote, the computation of which we bypass here.

In the following chapters, we demonstrate outstanding results in problems such as dimensionality estimation, manifold learning, nonlinear interpolation, geodesic distance measurement and function approximation. We anticipate that the work presented here will serve as the groundwork for research in domains that include instance-based learning, pattern recognition, classification, data mining and kinematics.

Chapter 9

Dimensionality Estimation, Manifold Learning and Function Approximation

Equipped with the new N-D implementation of tensor voting, we address instance-based learning, a branch of machine learning that aims at learning in continuous domains from observations. The observations are vectors in an N-D space, such as the positions, velocities and motor commands of the robotic arm of Fig. 1.4. The goal of instance-based learning is to learn the relationships between these variables, and, eventually, the forward kinematic model of the arm. The assumption is that the observations are constrained by the kinematic model to lie in a limited part of the N-D space, typically a manifold, the dimensionality of which is an indication of the degrees of freedom of the underlying system. We pose the problem of instance-based learning in an equivalent form: manifold learning from observations.

In this chapter, we address manifold learning from unorganized points from a perceptual organization standpoint. Unlike traditional instance-based learning approaches, we do not perform dimensionality reduction, which would limit the class of datasets we are able to process, but instead perform all operations in the original input space. We are able to estimate local dimensionality and structure, measure geodesic distances and interpolate the data nonlinearly to obtain new samples for classes of datasets that cannot be handled by current state-of-the-art approaches. We first show that we can obtain reliable dimensionality estimates at each point, instead of a global average estimate for the entire dataset, which is a major advantage over competing methods. We then present a quantitative evaluation of our results in the estimation of local manifold structure using synthetic datasets with known ground truth. In addition, we compare our approach with a number of leading methods for manifold learning at the task of measuring distances between points on the manifold. We also present results on data with varying dimensionality and intersections under severe noise corruption.

Finally, we propose a non-iterative, local, nonparametric approach that can successfully approximate nonlinear functions in high-dimensional spaces in the presence of noise. The distinction between low and high-dimensional spaces is necessary, since highly specialized methods for low-dimensional cases exist in the literature. We pose the problem as manifold learning from unorganized points and address it using the tools developed in the first sections of the chapter. We present quantitative results on data with varying density, outliers and perturbation. We also show promising preliminary results on real data.

9.1 Introduction

Machine learning is a research area in artificial intelligence that aims at the improvement of the behavior of agents through diligent study of observations [124]. It deals with the development of algorithms that analyze the observed data to identify patterns and relationships, in order to predict unseen data. Here, we address a subfield of machine learning that operates in continuous domains and learns from observations that are represented as points in a Euclidean space. This type of learning is termed *instance-based* or *memory-based* learning [99]. Learning in discrete domains, which attempts to infer the decisions and strategies that maximize a utility function, or the states, transitions and rules that govern a system, is out of the scope of our research.

The problem of learning a target function based on instances is equivalent to learning a manifold formed by a set of points, and thus being able to predict the positions of other points on the manifold. The first task, given a set of observations, is to determine the intrinsic dimensionality of the data. This can provide insight for the complexity of the model that generates the data, the type of model needed to describe them, as well as the degrees of freedom of the system. We then attempt to estimate the orientation at each point. Both of these tasks are actually accomplished simultaneously by encoding the observations as tensors and analyzing the results of tensor voting. Since processing, that estimates dimensionality and orientation, is performed on the inputs, our approach falls under the "eager learning" category according to Mitchell [99]. Unlike other eager approaches, however, ours is not global. This offers considerable advantages when the data become more complex, or when the number of instances is large. The main characteristic of eager, instance-based learning methods is that the query points are not taken into account when decisions are made. In other words, the orientation of each training sample

is estimated during the training stage, and the fact that a previously unseen point also belongs to the manifold does not alter these estimates. Assuming that the distribution of the data is stationary, this does not cause any difficulties. If stationarity does not hold, however, learning has to be accompanied by an update and a forgetting mechanism. This is also out of the scope of this chapter.

Instance-based learning has recently received renewed interest from the machine learning community, due to its many applications in the fields of pattern recognition, data mining, kinematics, function approximation and visualization, among others. This interest was sparked by a wave of new algorithms that advanced the state of the art and are capable of learning nonlinear manifolds in spaces of very high dimensionality. These include kernel PCA [141], locally linear embedding (LLE) [122], Isomap [158] and charting [13], which are described in Section 9.2. They aim at reducing the dimensionality of the input space in a way that preserves certain geometric or statistical properties of the data. Isomap, for instance, attempts to preserve the geodesic distances between all points as the manifold is "unfolded" and mapped to a space of lower dimension. A common assumption is that the desired manifold consists of locally linear patches. We relax this assumption by only requiring that manifolds be smooth almost everywhere. Smoothness in this context is the property of the manifold's orientation to vary gradually as one moves from point to point. This property is encoded in the votes that each point casts to its neighbors.

We take a different path to learning low dimensional manifolds from sample points in a high dimensional space. Traditional methods for manifold learning address the problem as one of dimensionality reduction. We propose an approach for the unsupervised learning of manifold

structure in a way that is useful for tasks such as geodesic distance estimation and nonlinear interpolation, that does not embed the data in a lower dimensional space. We compute local dimensionality estimates, but instead of performing dimensionality reduction, we perform all operations in the original input space, taking into account the estimated dimensionality of the data. This allows us to process datasets that are not manifolds globally, or ones with varying intrinsic dimensionality. The latter ones pose no additional difficulties, since we do not use a global estimate for the dimensionality of the data. Moreover, the presence of outliers, boundaries, intersections or multiple manifolds pose no additional difficulties. Non-manifolds, such as hyper-spheres, can be processed without any modifications of the algorithm since we do not attempt to estimate a global "unfolding". Quantitative results for the robustness against outliers that outnumber the inliers are presented in Sections 9.4 and 9.5.

Manifold learning serves as the basis for the last part of the chapter, which deals with function approximation. The approximation of an unknown function based on observations is critical for predicting the responses of both natural and artificial systems. The main assumption is that some form of smoothness exists in the data [116] and unobserved outputs can be predicted from previously observed outputs for similar inputs. Highly specialized methods for univariate and bivariate functions exist in the literature, but are outside the scope of this chapter. Here, we address the approximation of multivariate functions, and thus employ methods that can be generalized to high dimensions. A common practice is to treat functions with multiple outputs as multiple single-output functions. We adopt this scheme here, even though nothing prohibits us from directly approximating multiple-input multiple-output functions.

As suggested by Poggio and Girosi [116], function approximation from samples and hypersurface inference are equivalent. Here, we propose a local, non-parametric approach that is based on the assumption of smoothness, but can easily handle discontinuities, intersections and noise in the data. The fact that no model other than local constant curvature connections between the voter and receiver is used allows us to handle a broad range of functions. Also contributing to this is the absence of global computations and parameters, such as the number of local models in an ensemble, that need to be selected. The inevitable trade-off between over-smoothing and over-fitting is regulated by the selection of the scale. Small values reduce the size of the voting neighborhood and preserve details better, but are more vulnerable to noise and over-fitting. Large values produce smoother approximations that are more robust to noise. As shown in Section 9.7, the results are very stable with respect to the scale. As most of the local methods reviewed in the next section, our algorithm is memory based. This increases flexibility, since we can process data that do not conform to any model, but also increases memory requirements, since all samples are kept in memory.

This chapter is organized as follows: related work including the algorithms that are compared against ours are presented in the next section; results in dimensionality estimation are presented in Section 9.3, while results in local structure estimation are presented in Section 9.4; our algorithm for measuring distances on the manifold and a quantitative comparison with state of the art methods is presented in Section 9.5; an algorithm for generating outputs for unobserved inputs is described in Section 9.6; followed by experimental results in function approximation in Section 9.7; finally, Section 9.8 concludes the chapter.

9.2 Related Work

In this section, we present related work in the domains of dimensionality estimation, manifold learning and multivariate function approximation. We begin with approaches whose ultimate goal is dimensionality estimation, and proceed with approaches that attempt to learn the structure of a manifold, some of which also address dimensionality estimation. In the last paragraph of the section, we briefly describe representative approaches to function approximation that can be applied in high-dimensional spaces.

Dimensionality Estimation An approach for dimensionality estimation was presented by Bruske and Sommer [16]. An optimally topology preserving map (OTPM) is constructed for a subset of the data, which is produced after vector quantization. Principal Component Analysis (PCA) is then performed for each node of the OTPM under the assumption that the underlying structure of the data is locally linear. The average of the number of significant singular values at the nodes is the estimate of the intrinsic dimensionality.

Kégl [65] estimates the capacity dimension of the manifold, which does not depend on the distribution of the data, and is equal to the topological dimension. An efficient approximation based on packing numbers is proposed. The algorithm takes into account dimensionality variations with scale, and is based on a geometric property of the data, rather than successive projections to increasingly higher-dimensional subspaces until a certain percentage of the data is explained.

Costa and Hero [19] estimate the intrinsic dimension of a manifold and the entropy of the samples using geodesic-minimal-spanning trees. The method is similar to Isomap and thus produces a single global estimate. Levina and Bickel [80] compute maximum likelihood estimates of dimensionality by examining the number of neighbors included in spheres whose radius is selected in such a way that the density of the data can be assumed constant, and enough neighbors are included. These requirements cause an underestimation of the dimensionality when it is very high.

Manifold Learning Here, we briefly present recent approaches for learning low dimensional embeddings from points in high dimensional spaces. Most of them are extensions of linear techniques, such as Principal Component Analysis (PCA) [59] and Multi-Dimensional Scaling (MDS) [21], based on the assumption that nonlinear manifolds can be approximated by locally linear patches. In contrast to other methods, Schölkopf *et al.* [141] propose kernel PCA that attempts to find linear patches using PCA in a space of typically higher dimensionality than the input space. Correct kernel selection can reveal the low dimensional structure of the input data. For instance a second order polynomial kernel can "flatten" quadratic manifolds.

Locally Linear Embedding (LLE) was presented by Roweis and Saul in [122] and [132]. The underlying assumption is that if data lie on a locally linear, low-dimensional manifold, then each point can be reconstructed from its neighbors with appropriate weights. These weights should be the same in a low-dimensional space whose dimensionality is greater or equal to the intrinsic dimensionality of the manifold, as long as the manifold is locally linear. The LLE algorithm computes the basis of such a low-dimensional space. The dimensionality of the

embedding, however, has to be given as a parameter, since it cannot always be estimated from the data [132]. Moreover, the output is an embedding of the given data, but not a mapping from the ambient to the embedding space. Global coordination of the local embeddings, and thus a mapping, can be computed according to [157]. LLE is not isometric and often fails by mapping distant points close to each other.

Tenenbaum *et al.* [158] proposed Isomap, which is an extension of multi-dimensional scaling (MDS) that uses geodesic instead of Euclidean distances. This allows Isomap to handle nonlinear manifolds, whereas MDS is limited to linear data. The geodesic distances between points are approximated by graph distances. Then, MDS is applied on the geodesic distances to compute an embedding that preserves the property of points to be close or far away from each other. Due to its global formulation, Isomap's computational cost is considerably higher than that of LLE and other local methods. On the other hand, it can handle points not in the original dataset, and perform interpolation. C-Isomap, a variation of Isomap that can be applied to data with intrinsic curvature, but known distribution, and L-Isomap, a faster alternative that only uses a few landmark point for distance computations have also been proposed in [22]. Isomap and its variants are limited to convex datasets.

Laplacian Eigenmaps were proposed by Belkin and Nuyogi [7] who compute the graph Laplacian of the adjacency graph of the input data, which is an approximation of the Laplace-Beltrami operator on the manifold, and exploit its locality preserving properties that were first observed in the field of clustering. The Laplacian eigenmaps algorithm can be viewed as a

generalization of LLE. Much like LLE, the dimensionality of the manifold also has to be provided, the computed embeddings are not isometric and a mapping between the two spaces is not produced. The latter is addressed in [46] where a variation of the algorithm is proposed.

Donoho and Grimes [24] propose Hessian LLE (HLLE), an approach similar to the above, which computes the Hessian instead of the Laplacian. The authors claim that the Hessian is better suited than the Laplacian in detecting linear patches on the manifold. The major contribution of this approach is that it proposes a global, isometric method, which, unlike Isomap, can be applied to non-convex datasets. The need to estimate second derivatives from possibly noisy, discrete data makes the algorithm more sensitive to noise than the others reviewed here.

Semidefinite Embedding (SDE) was proposed by Weinberger and Saul [168] who address the problem of manifold learning by enforcing local isometry. The lengths of the sides of triangles formed by neighboring points are preserved during the embedding. These constraints can be expressed in terms of pairwise distances and the optimal embedding can be found by semidefinite programming. The method is the most computationally demanding reviewed here, but can reliably estimate the underlying dimensionality of the inputs.

In Section 9.5, we compare our results with those from the five approaches described in this paragraph.

Other research related to ours includes the charting algorithm of Brand [13]. It computes a pseudo-invertible mapping of the data, as well as the intrinsic dimensionality of the manifold, which is estimated by examining the rate of growth of the number of points contained in hyper-spheres as a function of the radius. Linear patches, areas of curvature and noise can be discriminated using the proposed measure. Affine transformations that align the coordinate

systems of the linear patches are computed at the second stage. This defines a global coordinate system for the embedding and thus a mapping between the input space and the embedding space.

Wang *et al.* [165] propose an adaptive version of the local tangent space alignment (LTSA) of Zhang and Zha [179], a local dimensionality reduction method that is a variation of LLE. Wang *et al.* address a limitation of most of the approaches presented in this section, which is the use of a fixed number of neighbors for all points in the data. This causes serious problems if that number is not selected properly, near boundaries, or if the density of the data is not approximately constant. The authors propose a method to adapt the neighborhood size according to local criteria and demonstrate its efficacy on datasets of varying distribution.

The difference between our approach and those of [13, 16, 19, 65, 80, 168] is that ours produces reliable dimensionality estimates at the point level, which do not have to be averaged over the entire dataset. While this is not important for datasets with constant dimensionality, the ability to estimate local dimensionality reliably becomes a key factor when dealing with data generated by different unknown processes.

Function Approximation Neural networks are often employed as global methods for function approximation. Poggio and Girosi [116] addressed function approximation in a regularization framework implemented as a three-layer neural network. They view the problem as hypersurface reconstruction, where the only reasonable assumption is that of smoothness. The emphasis is on the selection of the appropriate approximating functions and optimization algorithm. Sanger [127] proposed a tree-structured neural network for function approximation in

high-dimensional spaces. It does not suffer from the exponential growth with dimensionality of the number of models and parameters that plagued previous approaches. It does, however, require the selection of an appropriate set of basis functions to approximate a given function. Neural network based approaches with pre-specified types of models were proposed by: Barron [4] who derived the bounds for approximation using a superposition of sigmoidal functions; Breiman [14] who proposed a simpler and faster model based on hinging hyperplanes and Saha *et al.* [125] who used radial basis functions.

Xu *et al.* [173] modified the training scheme for the mixture of experts architecture so that a single-loop EM algorithm is sufficient for optimization. Mitaim and Kosko [98] approached the problem within the fuzzy systems framework. They investigated the selection of the shape of fuzzy sets for an adaptive fuzzy system and concluded that no shape emerges as the best choice. All these approaches are global and model based. They can achieve good performance, but they require all the inputs to be available at the same time for training and the selection of an appropriate model that matches the unknown function. If the latter is complex, the resulting model may have an impractically large number of parameters.

Lawrence *et al.* [74] compared a global approach using a multi-layer perceptron neural network with a linear local approximation model. They found that the local model performed better when the density of the input data deviated a lot from being uniform. Furthermore, the local model allowed for incremental learning and cross-validation. On the other hand, it showed poorer generalization, slower performance after training and required more memory, since all input data had to be kept. The global model performed better in higher dimensions, where data sparsity becomes a serious problem for the local model.

Schaal and Atkenson [137] proposed a nonparametric, local, incremental learning approach based on receptive field weighted regression. The approach is truly local since the parameters for each model and the size and shape of each receptive field are learned independently. The provided mechanisms for the addition and pruning of local models enable incremental learning as new data points become available. This would have been impossible otherwise, since the entire computation would have to be repeated if the number of local models changed. Atkenson *et al.* [3] survey local weighted learning methods and identify the issues that must be taken into account. These include the selection of the distance metric, the weighting function, prediction assessment and robustness to noise. The authors argue that in certain cases no values of the parameters of a global model can provide a good approximation of the true function. In these cases, a local approximation using a simpler, even linear model, is a better approach than increasing the complexity of the global model. Along these lines, Vijaykumar and Schaal [164] proposed locally weighted projection regression, an algorithm based on successive univariate regressions along projections of the data in directions given by the gradient of the underlying function.

We also opt for a local approach and address the problem as manifold learning. Note, however, that we are not limited to functions that are strictly manifolds. The recent group of manifold learning algorithms based on dimensionality reduction is not applicable for function approximation, since they compute neighborhood relationships in the form of a graph, but do not compute the geometric structure of the samples, with the exception of the adaptive local tangent space alignment method [165]. It proposes a way to adaptively select the number of neighbors k used at each point and computes its local tangent space. The adaptive number of

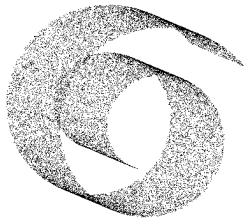


Figure 9.1: The "Swiss Roll" dataset in 3-D

neighbors at each point is an important contribution in the field of manifold learning, since the contributions of each neighbor are typically not weighted, making the algorithms very sensitive to the selection of k. Using tensor voting, we are able to reliably estimate the normal and tangent space at each sample, as described in the previous chapter. These estimates allow us to perform nonlinear interpolation and generate outputs for unobserved inputs, even under severe noise corruption. Since the votes are weighted, sensitivity to the scale of voting and outliers is small, as demonstrated by the experiments in the remainder of the chapter.

9.3 Dimensionality Estimation

In this section, we present experimental results in dimensionality estimation. As shown in the previous chapter, the dimensionality at each point can be found as the maximum gap in the eigenvalues of the tensor after votes from its neighboring points have been collected. All inputs consist of unoriented points and are encoded as ball tensors.

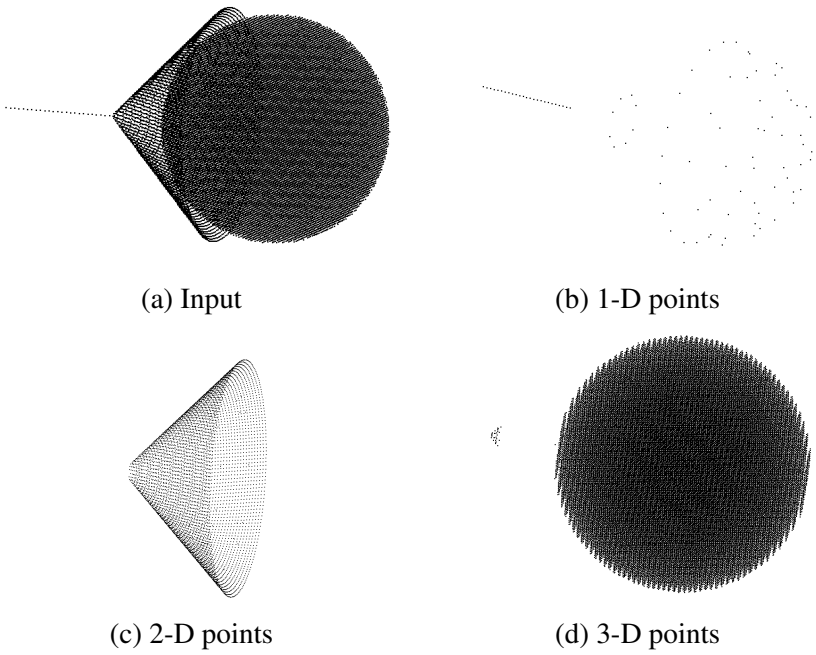
Swiss Roll The first experiment is on the "Swiss Roll" dataset, which is available online at http://isomap.stanford.edu/. It contains $20,000$ points on a 2-D manifold in 3-D (Fig. 9.1). We perform a simple evaluation of the quality of the orientation estimates by projecting the nearest

neighbors of each point on the estimated tangent space and measuring the percentage of the distance that has been recovered. This is a simple measure of the accuracy of the local linear approximation of the nonlinear manifold. The percentage of points with correct dimensionality estimates and the percentage of recovered distances for the 8 nearest neighbors as a function of σ, can be seen in Table 9.1. The performance is the same at boundaries. The number of votes cast by each point ranges from 187 for $\sigma^2 = 50$ to 5440 for $\sigma^2 = 1000$. The accuracy is high for a large range of σ.

σ^2	Correct Dim. Estimation (%)	Perc. of Dist. Recovered (%)	Time (sec)
50	99.25	93.07	7
100	99.91	93.21	13
200	99.95	93.19	30
300	99.92	93.16	47
500	99.68	93.03	79
700	99.23	92.82	112
1000	97.90	92.29	181

Table 9.1: Rate of correct dimensionality estimation and execution times as functions of σ for the "Swiss Roll" dataset.

Structures with varying dimensionality The second dataset is in 4-D and contains points sampled from three structures: a line, a 2-D cone and a 3-D hyper-sphere. The hyper-sphere is a structure with three degrees of freedom. It cannot be unfolded unless we remove a small part from it. Figure 9.2(a) shows the first three dimensions of the data. The dataset contains a total $135,864$ points, voting is performed with $\sigma^2 = 200$ and takes 51 minutes and 14 seconds. Figures 9.2(c-d) show the points classified according to their dimensionality. Performing the same analysis as above for the accuracy of the tangent space estimation, 91.04% of the distances



(a) Input (b) 1-D points

(c) 2-D points (d) 3-D points

Figure 9.2: Data of varying dimensionality in 4-D. The first three axes of the input and the classified points are shown.

of the 8 nearest neighbors of each point lie on the tangent space, even though both the cone and the hyper-sphere have intrinsic curvature and cannot be accurately approximated by linear models. All the methods presented in Sec. 9.2 would fail for this dataset because of the presence of structures with different dimensionalities and because the hyper-sphere cannot be unfolded.

Data in high dimensions The datasets for this experiment were generated by sampling a few thousand points with low intrinsic dimensionality (3 or 4) and mapping them to a medium dimensional space (14- to 16-D) using linear and quadratic functions. The generated points were then rotated and embedded in a 50- to 150-D space while uniform noise was added to all

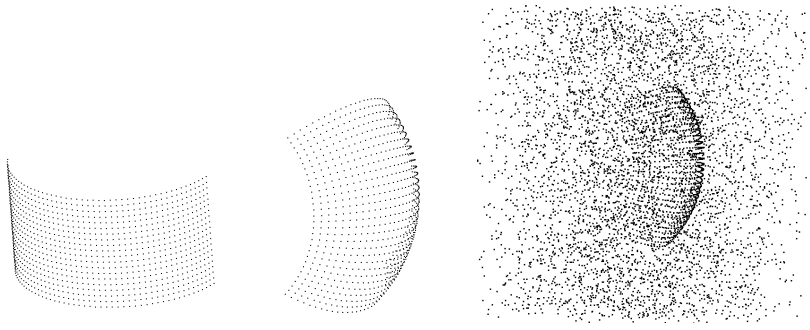
dimensions. The percentage of correct point-wise dimensionality estimates after tensor voting can be seen in Table 9.2.

Intrinsic Dim.	Linear Mappings	Quadratic Mappings	Space Dim.	Dim. Est. (%)
4	10	6	50	93.6
3	8	6	100	97.4
4	10	6	100	93.9
3	8	6	150	97.3

Table 9.2: Rate of correct dimensionality estimation for high dimensional data

9.4 Manifold Learning

In this section, we present quantitative results on simple datasets in 3-D for which ground truth can be analytically computed. In Section 9.5, we process the same data with state of the art manifold learning algorithms and compare against our results. The two datasets are a section of a cylinder and a section of a sphere shown in Fig. 9.3. The cylindrical section spans 150^o and consists of 1000 points. The spherical section spans $90^o \times 90^o$ and consists of 900 points. Both



(a) Cylinder (b) Sphere (c) Noisy sphere

Figure 9.3: Datasets used in Sections 9.4 and 9.5

are approximately uniformly sampled. The points are represented by ball tensors, assuming no information about their orientation. In the first part of the experiment, we compute local dimensionality and normal orientation as a function of scale. The results are presented in Tables 9.3 and 9.4. The results show that if the scale is not too small, dimensionality estimation is very reliable. For all scales the angular errors are below 0.4^o. Similar results are obtained for a large range of scales.

σ^2	Neighbors	Angular Error	Dim. Estim. (%)
10	5	0.06	4
20	9	0.07	90
30	9	0.08	90
40	12	0.09	90
50	20	0.10	100
60	20	0.11	100
70	23	0.12	100
80	25	0.12	100
90	30	0.13	100
100	34	0.14	100

Table 9.3: Results on the cylinder dataset. Shown in the first column is σ^2, in the second is the average number of neighbors that cast votes to each point, in the third the average error in degrees of the estimated normals, and in the fourth the accuracy of dimensionality estimation.

The same experiments were performed for the spherical section in the presence of outliers. Quantitative results are shown in the following tables for a number of outliers that ranges from 900 (equal to the inliers) to 5000. The latter dataset is shown in Fig. 9.3(c). Note that performance was evaluated only on the points that belong on the sphere. Larger values of the scale prove to be more robust to noise, as expected. The smallest values of the scale result in voting neighborhoods that include less than 10 points, which are insufficient. Taking that into account, performance is still good even with wrong parameter selection. Also note that one could reject

σ^2	Neighbors	Angular Error	Dim. Estim. (%)
10	5	0.20	44
20	9	0.23	65
30	11	0.24	93
40	20	0.26	94
50	21	0.27	94
60	23	0.29	94
70	26	0.31	94
80	32	0.34	94
90	36	0.36	94
100	39	0.38	97

Table 9.4: Results on the sphere dataset. The columns are the same as in Table 9.3.

the outliers, which have smaller eigenvalues than the inliers, by thresholding and vote again to obtain better estimates of structure and dimensionality. Even a single pass of tensor voting, however, turns out to be very effective, especially considering that no other method can handle such a large number of outliers. Foregoing the embedding computation is a main reason that allows our method to perform well in the presence of noise.

Outliers	900		3000		5000	
σ^2	AE	DE	AE	DE	AE	DE
10	1.15	44	3.68	41	6.04	39
20	0.93	65	2.95	52	4.73	59
30	0.88	92	2.63	88	4.15	85
40	0.88	93	2.49	90	3.85	88
50	0.90	93	2.41	92	3.63	91
60	0.93	94	2.38	93	3.50	93
70	0.97	94	2.38	93	3.43	93
80	1.00	94	2.38	94	3.38	94
90	1.04	95	2.38	95	3.34	94
100	1.07	97	2.39	95	3.31	95

Table 9.5: Results on the sphere dataset contaminated by noise. AE: error in normal angle estimation in degrees, DE: correct dimensionality estimation (%).

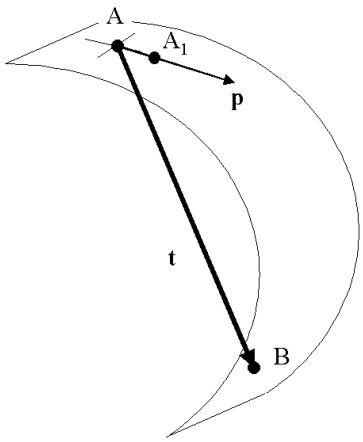


Figure 9.4: Nonlinear interpolation on the tangent space of a manifold

9.5 Manifold Distances and Nonlinear Interpolation

Learning the manifold structure from samples is an interesting problem. The ability to evaluate distances between points and to interpolate on the manifold are more useful for many applications. Here, we show how to compute the distance between any two points on a manifold essentially by taking small steps on the manifold, collecting votes, estimating the local tangent space and advancing on it until the destination is reached.

Processing begins by learning the manifold structure, as in the previous section, starting from unoriented points that are represented by ball tensors. We, then, select a starting point that has to be on the manifold and a target point or a desired direction (the vector from the origin to the target). At each step, we can project the desired direction on the tangent space of the current point and create a new point at a small distance. The tangent space of a new point is computed by collecting votes from the neighboring points, as in regular tensor voting. Note that the tensors used here are no longer balls, but the ones resulting from the previous pass of voting.

For instance, in Fig. 9.4, we start from point A and wish to reach B. We project $\vec{t}$, the vector from A to B, on the tangent space of A and obtain its projection $\vec{p}$. Then, we take a small step along $\vec{p}$ to reach point A_1, on which we collect votes to obtain an estimate of its tangent space. The desired direction is then projected on the tangent space of the new point and so forth until the destination is reached within ϵ. The manifold distance between two points is approximated by measuring the length of the path.

For the experiments reported in Table 9.6, we learned the manifolds of the previous section using tensor voting. We also computed embeddings using LLE [122], Isomap [158], Laplacian eigenmaps [7], HLLE [24] and SDE [168]. Matlab implementations for these methods were downloaded from the following internet locations.

- LLE from http://www.cs.toronto.edu/~roweis/lle/code.html
- Isomap from http://isomap.stanford.edu/
- Laplacian Eigenmaps from http://people.cs.uchicago.edu/~misha/ManifoldLearning/index.html
- HLLE from http://basis.stanford.edu/HLLE
- and SDE from http://www.seas.upenn.edu/~kilianw/sde/download.htm.

We are grateful to the authors for making the core of their methods available to the community. We intend to make our software publicly available as well.

The experiment was performed as follows. We randomly selected 5000 pairs of points on each manifold and attempted to measure the distance between the points of each pair in

the input space using tensor voting and in the embedding space using the other five methods. The estimated distances were compared to the ground truth: $r\Delta\theta$ for the sphere and $\sqrt{(r\Delta\theta)^2 + (\Delta z)^2}$ for the cylinder. Among these approaches, only Isomap and SDE produce isometric embeddings, and only Isomap preserves the absolute distances between the input and the embedding space. To make the evaluation fair, we compute a uniform scale that minimizes the error between the computed distances and the ground truth for all methods, except Isomap. Thus, perfect distances ratios would be awarded a perfect rating in the evaluation, even if the absolute magnitudes of the distances are meaningless in the embedding space. For all the algorithms, we tried a wide range of values for K. In some cases, we were not able to produce good embeddings of the data for any value of K, especially for the cylinder.

The evaluation of the quality of manifold learning based on the computation of pairwise distances is a fair measure for the performance of all algorithms, since high quality manifold learning should minimize distortions. In addition, it does not require operations not supported by all algorithms, such as the processing of points not included in the training dataset. Quantitative results are presented in the following table along with the value of K that was used. In the case of tensor voting, the same scale was used for both learning the manifold and computing distances.

We also applied our approach in the presence of 900, 3000 and 5000 outliers. Keep in mind that the sphere and the cylinder datasets consist of 900 and 1000 points respectively. The error rates for the sphere are 0.39, 0.47 and 0.53% respectively. The rates for the cylinder are 0.77, 1.17 and 1.22%. Compared with the noise free case, these results demonstrate that our approach degrades slowly in the presence of outliers. The best performance achieved by any

Dataset	Sphere		Cylinder	
	K	Err(%)	K	Err(%)
LLE	18	5.08	6	26.52
Isomap	6	1.98	30	0.35
Laplacian	16	11.03	10	29.36
HLLE	12	3.89	40	26.81
SDE	2	5.14	6	25.57
TV (σ^2)	60	0.34	50	0.62

Table 9.6: Error rates in distance measurements between pairs of points on the manifolds. The best result of each method is reported along with the number of neighbors used for the embedding (K), or the scale σ^2 in the case of tensor voting (TV).

other method is 3.54% on the sphere dataset with 900 outliers by Isomap. Complete results are shown in Table 9.7. In many cases, we were unable to achieve useful embeddings for datasets with outliers. Taking the scaling that is applied to the distances into account, error rates in the order of 25% are clear indications of poor performance.

Dataset	Sphere 900 outliers		Cylinder 900 outliers	
	K	Err(%)	K	Err(%)
LLE	40	60.74	6	15.40
Isomap	18	3.54	14	11.41
Laplacian	6	13.97	14	27.98
HLLE	30	8.73	30	23.67
SDE		N/A		N/A
TV (σ^2)	70	0.39	100	0.77

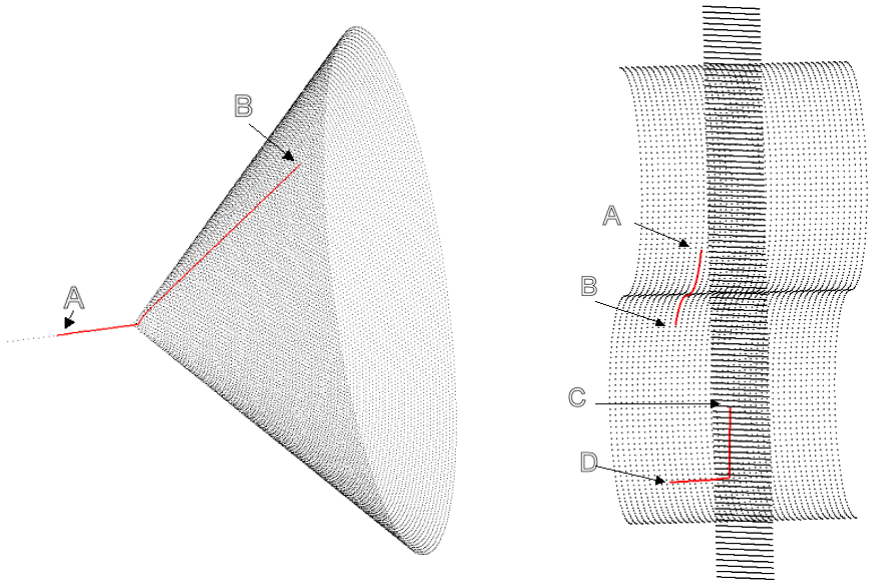
Table 9.7: Error rates in distance measurements between pairs of points on the manifolds under outlier corruption. The best result of each method is reported along with the number of neighbors used for the embedding (K), or the scale σ^2 in the case of tensor voting (TV). Note that HLLE fails to compute an embedding for small values of K, while SDE fails at both examples for all choices of K

Datasets with varying dimensionality and intersections For the final experiment of this section, we create synthetic data in 3-D that were embedded in higher dimensions. The first

Dataset	Sphere 3000 outliers		Cylinder 3000 outliers	
	K	Err(%)	K	Err(%)
TV (σ^2)	80	0.47	100	1.17
Dataset	Sphere 5000 outliers		Cylinder 5000 outliers	
	K	Err(%)	K	Err(%)
TV (σ^2)	100	0.53	100	1.22

Table 9.8: Error rates for our approach in the presence of 3000 and 5000 outliers

dataset consists of a line and a cone. The points are embedded in 50-D by three orthonormal 50-D vectors and initialized as ball tensors. Tensor voting is performed in the 50-D space and a path from point A on the line to point B on the cone is interpolated as in the previous experiment, making sure that it belongs in the local tangent space, which changes dimensionality from one to two. The data is re-projected back to 3-D for visualization in Fig. 9.5(a). In the second part of the experiment, we generate an intersecting S-shaped surface and a plane (a total of 11,000 points) and 30,000 outliers, and embed them in a 30-D space. Without explicitly removing the noise, we interpolate between two points on the S (A and B) and a point on the S and a point on the plane (C and D) and create the paths shown in Fig. 9.5(b) re-projected in 3-D. The first path is curved, while the second jumps from manifold to manifold still keeping the optimal path. (The outliers are not shown for clarity.) Processing time for 41,000 points in 30-D is 2 min. and 40 sec. on a Pentium 4 at 2.8 MHz using voting neighborhoods that included an average of 44 points.



(a) Line and cone (b) S and plane

Figure 9.5: Nonlinear interpolation in 50-D with varying dimensionality (a) and 30-D with intersecting manifolds under noise corruption (b).

9.6 Generation of Unobserved Samples

In this section, we build upon the results of the previous section to address function approximation. As before, observations of inputs and outputs are available for training. The difference with the examples previous sections is that the queries are given as input vectors with unknown output values, and thus are of lower dimension than the voting space. What is missing is a procedure to find a point on the manifold that corresponds to an input similar to the query. Then, in order to predict the output y of the function for an unknown input $\vec{x}$, under the assumption of local smoothness, we move on the manifold formed by the training samples until we reach the point corresponding to the given input coordinates. To ensure that we always remain on the manifold, we need to start from a point on it and proceed as in the previous section.

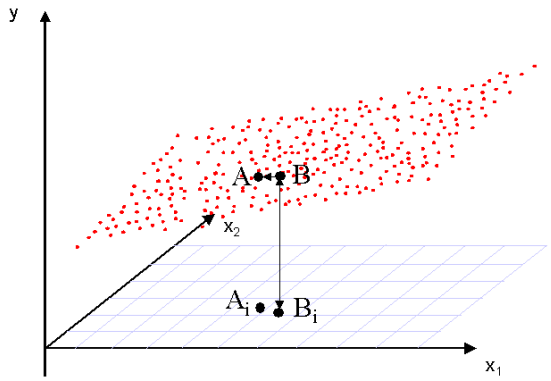
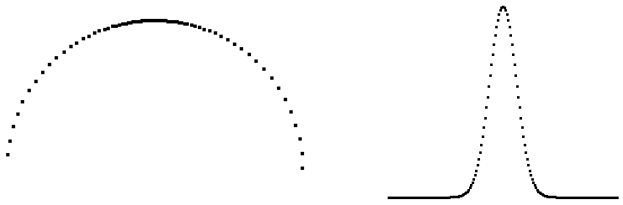


Figure 9.6: Interpolation to obtain output value for unknown input point A_i

One way to find a suitable starting point is to find the nearest neighbor of $\vec{x}$ in the input space, which has fewer dimensions than the joint input-output (voting) space. Then, we can compute the desired direction in the low dimensional space and project it to the input-output space. If many outputs are possible for a given input (if the data do not form a function in the strict sense), we have to either find neighbors at each branch of the function and produce multiple outputs, or use other information, such as the previous state of the system, to pursue only one of the alternatives. Figure 9.6 provides a simple illustration. We begin with a point A_i in the input space. We proceed by finding its nearest neighbor in the input space B_i. (Even if B_i is not the nearest neighbor the scheme still works but possibly requires more steps.) The sample B in the input-output space that corresponds to B_i is the starting point on the manifold. The desired direction is the projection of the A_iB_i vector on the tangent space of B. Now, we are in the case described in Section 9.5, where the starting point and the desired direction are known. Processing stops when the input coordinates of the point on the path from B are within ϵ of A_i. The corresponding point A in the input-output space is the desired interpolated sample.



(a) Samples from Eq. 9.1 (b) Samples from Eq. 9.2

Figure 9.7: Input data for the two experiments proposed by [165]

9.7 Nonparametric Function Approximation

As in all the experiments of this chapter, for all the results presented in this section, the input points were encoded as ball tensors, since we assume that we have no knowledge of their orientation. The first two experiments we conducted were on functions proposed in [165]. The key difficulty with these functions is the non-uniform density of the data. In the first example we attempt to approximate:

$$x_i = [cos(t_i), sin(t_i)]^T \quad t_i \in [0, \pi], t_{i+1} - t_i = 0.1(0.001 + |cos(t_i)|) \quad (9.1)$$

where the distance between consecutive samples is far from uniform. See Fig. 9.7(a) for the inputs and the second column of Table 9.9 for quantitative results on tangent estimation for 152 points as a function of scale.

In the second example, 180 points were uniformly sampled on the t-axis from the [-6, 6] interval. The output is produced by the following function:

$$x_i = [t_i, 10e^{-t_i^2}] \quad (9.2)$$

The points, as can be seen in Fig. 9.7(b), are not uniformly spaced. The quantitative results on tangent estimation accuracy for 180 and 360 samples from the same interval are reported in the last two columns of Table 9.9. Naturally, as the sampling becomes denser, the quality of the approximation increases. What should be emphasized here is the stability of the results as a function of σ. Even with as few as 5 or 6 neighbors included in the voting neighborhood, the tangent at each point is estimated rather accurately.

σ^2	Eq. 9.1 152 pts	Eq. 9.2 180 pts	Eq. 9.2 360 pts
10	0.60	4.52	2.45
20	0.32	3.37	1.89
30	0.36	2.92	1.61
40	0.40	2.68	1.43
50	0.44	2.48	1.22
60	0.48	2.48	1.08
70	0.51	2.18	0.95
80	0.54	2.18	0.83
90	0.58	2.02	0.68
100	0.61	2.03	0.57

Table 9.9: Error in degrees for tangent estimation for the functions of Eq. 9.1 and Eq. 9.2

For the next experiment we approximated the following function, proposed by Schaal and Atkenson in [137]:

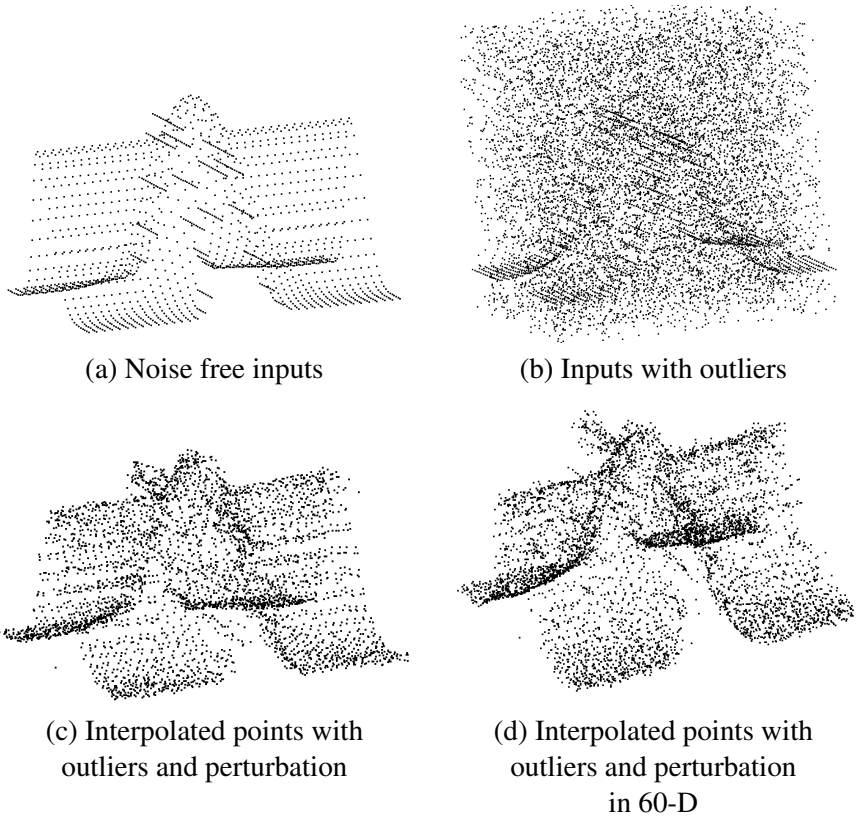
$$y = max\{e^{-10x_1^2}, e^{-50x_2^2}, 1.25e^{-5(x_1^2+x_2^2)}\} \tag{9.3}$$

1681 samples of y were generated by uniformly sampling the $[-1, 1] \times [-1, 1]$ square. We performed four experiments with increasing degree of difficulty. In all cases, after voting on the given inputs, we generated new samples by interpolating between the input points. The four configurations and noise conditions were:

- In the first experiment we performed all operations with noise free data in 3-D.
- For the second experiment we added 8405 outliers (five times more than the inliers) in a $2 \times 2 \times 2$ cube containing the data.
- For the third experiment we added Gaussian noise with variance 0.01 to the coordinates of all points.
- Finally, we embedded the perturbed data (and the outliers) in a 60-D space, before voting and nonlinear interpolation.

The noise-free and noisy input, as well as the generated points can be seen in Fig. 9.8. We computed the mean square error between the outputs generated by our method and Eq. 9.3 normalized by the variance of the data. The NMSE for all cases is reported in Table 9.10. Robustness against outliers is due to the fact that the inliers form a consistent surface and thus receive votes that support the correct local structure from other inliers. Outliers, on the other hand, are random and do not form any structure. They cast and receive inconsistent votes and therefore neither develop a preference for a certain manifold nor significantly disrupt the structure estimates at the inliers. They can be removed by simple thresholding since all their eigenvalues are small and almost equal. Note that performance in 60-D is actually better since the interference by outliers is reduced as the dimensionality of the space increases.

Results on real data The final experiment is on real data taken from the University of California at Irvine Machine Learning Repository, which is available online at: http://www.ics.uci.edu/ ˜mlearn/MLRepository.html. We used the "Auto-Mpg Database" that



(a) Noise free inputs

(b) Inputs with outliers

(c) Interpolated points with outliers and perturbation

(d) Interpolated points with outliers and perturbation in 60-D

Figure 9.8: Inputs and interpolated points for Eq. 9.3. The top row shows the noise-free inputs and the noisy input set where only 20% of the points are inliers. The bottom row shows the points generated in 3-D and 60-D respectively. In both cases the inputs were contaminated with outliers and Gaussian noise.

contains 392 samples of mileage per gallon (MPG) for automobiles as a function of seven variables: cylinders, displacement, horsepower, weight, acceleration, model year and origin. Due to the large differences in the range of values for each variable, we re-scaled the data so that the ratio of maximum to minimum standard deviation of the variables was $10:1$, instead of the original $1000:1$. We randomly selected 314 samples, approximately 80%, for training and 78 for testing. Since the variables are correlated, the samples do not form a manifold with seven degrees of freedom and do not cover the entire input domain. In fact, the estimated intrinsic

Experiment	NMSE
Noise-free	0.0041
Outliers	0.0170
Outliers & N(0, 0.01)	0.0349
Outliers & N(0, 0.01) in 60-D	0.0241

Table 9.10: Normalized MSE for the interpolated points of Eq. 9.3 under different noise conditions

dimensionality at each point by tensor voting ranges between one and three. After performing tensor voting on the 314 training samples, we attempted to estimate the MPG for the testing samples. We found the nearest neighbor in the 7-D input space for each testing sample and followed a path on the estimated manifold until the desired coordinates were reached. The average error of our estimates with respect to the ground truth was 10.45%. Stable performance between 10.45% and 10.48% was achieved even as the average voting neighborhood ranges between 35 and 294 points. Considering that 314 points are hardly sufficient for inferring a description of a complex 8-D space, the performance of our algorithm is promising. In fact the average error we achieve is 2.67 miles per gallon, which we consider acceptable given the sparsity of the data.

9.8 Conclusions

We have presented an approach to manifold learning that offers certain advantages over the state of the art. First, we are able to obtain accurate estimates of the intrinsic dimensionality at the point level. Moreover, since the dimensionality is found as the maximum gap in the eigenvalues of the tensor at the point, no thresholds are needed. In most other approaches, the

dimensionality has to be provided, or, at best, an average intrinsic dimensionality is estimated for the entire dataset, as in [13, 16, 19, 65, 168].

Second, even though tensor voting on the surface looks similar to other local, instance-based learning algorithms that propagate information from point to point, the fact that the votes are tensors and not scalars allows considerably more information to be conveyed by them. The properties of the tensor representation, which can handle the simultaneous presence of multiple orientations, allow the reliable inference of the normal and tangent space at each point. Due to its unsupervised nature, tensor voting is very robust against outliers, as demonstrated in previous chapters for the 2-D and 3-D case. This property holds in higher dimensions, where random noise is even more scattered. It should also be noted that the votes attenuate with distance and curvature. This is a more intuitive formulation than using the K nearest neighbors with equal weights, since some of them may be too far, or belong to a different part of the structure. For both the ϵ distance and the K nearest neighbor formulations, however, the distance metric in the input space has to be meaningful. This is also the case for all the methods presented in Sec. 9.2. The only free parameter in our approach is σ, the scale of voting. Small values tend to preserve details better, while large values are more robust against noise. An effective way to set the scale is to use a value that ensures that a sufficient number of points are included in each point's neighborhood. Since the votes are weighted, sensitivity to the choice of scale is rather low, as nearby points always contribute more, and similar results are obtained for a large range of scales, as shown by the experimental results.

The novelty of our approach to manifold learning is that it is not based on dimensionality reduction, in the form of an embedding or mapping between a high and a low dimensional space.

Instead, we perform tasks such as manifold distance measurement and nonlinear interpolation in the input space. Experimental results show that we can perform these tasks in the presence of outlier noise at high accuracy, even without explicitly removing the outliers from the data. This is due to the fact that the accumulated tensors at the outliers do not develop any preference for a particular structure and do not outweigh the contributions of the inliers. Moreover, the fact that we do not attempt an embedding broadens the range of datasets we can process. While isometric embeddings can be achieved for a certain class of manifolds, we are able to process non-flat manifolds and even non-manifolds. The last experiment of Section 9.5 demonstrates our ability to work with datasets of varying dimensionality or with intersecting manifolds. To the best of our knowledge, this is impossible with any other method.

We have also presented a local nonparametric approach to function approximation that combines the advantages of other local methods with the efficient representation and information propagation of tensor voting. We have shown that we can process challenging examples from the literature under very adverse noise conditions. As shown in the example of Eq. 9.3, even when 80% of the samples are outliers and the inliers are corrupted by noise in the form of perturbation, we are still able to correctly predict unobserved outputs. Outliers, even in large numbers, do not cause severe errors in the estimation of local structure since they do not align to form salient structures. Perturbation of the coordinates of the inliers by noise can lead to errors in the estimates, especially at small scales, but robustness against this type of noise is still rather high.

The proposed method is model-free, non-iterative and has only one free parameter: the scale of voting. Our results show that sensitivity with respect to scale is small, as shown in

Tables 9.1, 9.3-9.5 and 9.9. The same can be observed in the results for the Auto-Mpg Dataset, where the error fluctuates by 0.03% as the average voting neighborhood ranges between 35 and 294 points. The scale can be selected automatically by randomly sampling a few points before voting and making sure that enough points are included in their voting neighborhoods. The number of points that can be considered sufficient is a function of the dimensionality of the space as well as the intrinsic dimensionality of the data. A full investigation of data sufficiency is among the objectives of our future research.

Another important advantage of tensor voting is the absence of global computations, which makes time complexity $O(NMlogM)$, where N is the dimension of the space and M is the number of points. This property enables us to process datasets with very large number of points. Computation time scales linearly with the number of points assuming that more points are added to the dataset in a way that the density remains constant. In this case, the number of votes cast per point remains constant and time requirements grow linearly. If points are added in a way that increases the average density, voting can be performed with a smaller σ, thus maintaining the number of votes cast and the computation time per point. Complexity is adversely affected by the dimensionality of the space N, since eigen-decompositions of $N \times N$ tensors have to be performed. For most practical purposes, however, the number of points has to be considerably larger than the dimensionality of the space $(M \gg N)$ to allow structure inference. Computational complexity, therefore, is reasonable with respect to the largest parameter, which is typically M.

An issue we do not address here is that of the selection of an appropriate distance metric. We assume that the Euclidean distance in the input coordinate system is a meaningful distance metric. This is not the case if the measurements are not of the same type, as for instance distances and angles. On the other hand, a metric such as the Mahalanobis distance is not necessarily appropriate in all cases, since the data may lie in low dimensional manifolds in the input space and giving equal weight to all dimensions, including the redundant ones, would be detrimental. For the experiments shown in this chapter, we apply heuristic scaling of the coordinates when necessary. We intend to develop a systematic way based on cross-validation that automatically scales the coordinates by maximizing prediction performance for the observations that have been left out.

Our future research will focus on applying our algorithm to challenging real problems, such as the study of direct and inverse kinematics. One can also view the proposed approach as learning data from a single class, which can serve as groundwork for an approach for supervised and unsupervised classification.

Chapter 10

Conclusion

This chapter presents a brief summary of our contributions and discusses some possible extensions of this research. In the previous chapters, we described both a general perceptual organization approach as well as its application to a number of computer vision and machine learning problems. The cornerstone of our work is the tensor voting framework which provides a powerful and flexible way to infer the saliency of structures formed by elementary primitives. The primitives may differ from problem to problem, but the philosophy behind the manner in which we address them is the same. In all cases, we arrive at solutions which receive maximal support from the primitives as the most coherent and smooth structures. Throughout this work, we strove to maintain the desired properties that we described in the introduction, that the approach should be local, data-driven, unsupervised, robust to noise and able to represent all structure types simultaneously. These principles make our approach general and flexible, while allowing to incorporate problem specific constraints, such as uniqueness for stereo.

10.1 Contributions

The contributions of our research can be divided in three main categories: contributions to the tensor voting framework, to computer vision problems and to machine learning problems.

The contributions to the tensor voting framework are:

- the augmentation of the framework with first order representation and voting that allow the detection of boundaries and terminations;
- the direct application of tensor voting on the responses of the filter bank without the need for thresholding;
- special fields that facilitate modal and amodal completion; and
- a novel N-D implementation that made tackling problems in high-dimensional spaces feasible.

The main contributions to computer vision are:

- the integration of a filter bank as a module that can directly generate tensors suitable for initializing the voting process;
- the inference of salient contours and junctions from challenging natural images;
- an automatic mechanism for selecting between modal and amodal completion that can produce satisfactory interpretations of challenging visual inputs;
- a flexible way to combine pixel correspondences generated by a variety of possibly incompatible methods;

- a method to integrate monocular cues in stereo processing that does not rely on image segmentation, combines geometric and photometric smoothness and addresses the systematic noise sources inherent in stereo; and
- a framework for dense multiple view reconstruction with general camera placement that does not require prior foreground segmentation.

Finally, in terms of machine learning, the contributions are:

- the capability to produce dimensionality estimates at the point level, that are reliable enough to be used without having to resort to using a global average;
- a manifold learning approach that is applicable to datasets that cannot be addressed by other methods, such as non-manifolds, intersecting manifolds and structures with varying dimensionality;
- a nonlinear interpolation mechanism that does not assume local linearity and can be applied to the same wide range of data described above;
- an algorithm to estimate geodesic distances; and
- an approach to nonlinear, multi-variate function approximation.

10.2 Future Work

Our future work for the single image case will aim at bridging the gap between the work presented in Chapter 4 and Chapter 5. The inference of features from images in such a way that they can be useful for further symbolic processing remains a largely unsolved problem [136].

On one hand, images are discrete, limited-resolution, 2-D representations of a continuous, 3-D world through a nonlinear projection process. Given these limitations, one can design filters and filter banks that extract the maximal amount of low level information from pixel intensities. On the other hand, given satisfactory features, such as T-junctions and region boundaries, higher level processes can draw important conclusions about the configuration of the depicted scene and the depth ordering of the objects. What is missing is a mid level processing framework that converts filter outputs to symbolic descriptors. We have made efforts on both sides of the gap, but the problem is far from being solved, especially for images of complex, natural scenes.

Once enough progress has been made in the monocular case, the inference of descriptions from two or more images may be revisited. The work of Tao *et al.* [154] shows that in cases of images, where the lack of texture causes binocular algorithms to fail, outstanding results can be achieved by matching image regions, which can be easily segmented, rather than individual pixels. The algorithm fails in cases where texture information is abundant and traditional binocular algorithms excel. Ideally, one can use both monocular and binocular cues, balancing between them, to utilize the most reliable information for each case. In a generalization of the idea presented in Chapter 6, where binocular matching and consistency in color distribution are combined, the problem of stereo can be preceded by monocular analysis of each image. Evidence for the existence of layers or occlusion relationships, as could be implied by T-junctions for instance, can be used to aid the establishment of correspondences.

In terms of multiple view stereo, we are interested in taking advantage of the fact that we can process all data simultaneously and aim at super-resolution reconstructions. Even at its current state, our approach should be able to reconstruct parts of the surfaces that are visible by a large

number of cameras at a resolution that is higher than that of any single camera. Since our main assumption, that correct pixel correspondences align to form the scene surfaces, holds, then the density of points on these surfaces is larger than in the binocular case. This is an aspect of our approach that we have not fully investigated of explored.

We anticipate that the main focus of our future work will be in the field of machine learning. The research presented in Chapters 8 and 9 has only scratched the surface of the capabilities of our approach and will serve as the groundwork for research in domains that include pattern recognition, classification, data mining and kinematics. Unlike competing approaches, tensor voting scales well as the number of samples increases since it involves only local computations. This property is crucial in a world where information is generated and transmitted a lot faster than it can be processed. Our experiments have demonstrated that we can attain excellent performance levels given sufficient samples, and the latter abound in many cases. The development of on-line and lazy implementations of our approach, which would not require complete datasets to begin processing, is a direct and immediate extension.

Bibliography

[1] M. Agrawal and L.S. Davis. Window-based, discontinuity preserving stereo. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 66–73, 2004.

[2] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching. *Journ. of the ACM*, 45:891–923, 1998.

[3] C.G. Atkeson, A.W. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11(1-5):11–73, 1997.

[4] A.R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans. on Information Theory*, 39(3):930–945, 1993.

[5] P.N. Belhumeur. A bayesian-approach to binocular stereopsis. *Int. Journ. of Computer Vision*, 19(3):237–260, August 1996.

[6] P.N. Belhumeur and D. Mumford. A bayesian treatment of the stereo correspondence problem using half-occluded regions. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages 506–512, 1992.

[7] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[8] S. Birchfield and C. Tomasi. Depth discontinuities by pixel-to-pixel stereo. In *Int. Conf. on Computer Vision*, pages 1073–1080, 1998.

[9] S. Birchfield and C. Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(4):401–406, April 1998.

[10] S. Birchfield and C. Tomasi. Multiway cut for stereo and motion with slanted surfaces. In *Int. Conf. on Computer Vision*, pages 489–495, 1999.

[11] A.F. Bobick and S.S. Intille. Large occlusion stereo. *Int. Journ. of Computer Vision*, 33(3):1–20, September 1999.

[12] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, November 2001.

[13] M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems 15*, pages 961–968. MIT Press, Cambridge, MA, 2003.

[14] L. Breiman. Hinging hyperplanes for regression, classification, and function approximation. *IEEE Trans. on Information Theory*, 39(3):999–1013, 1993.

[15] M.Z. Brown, D. Burschka, and G.D. Hager. Advances in computational stereo. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(8):993–1008, August 2003.

[16] J. Bruske and G. Sommer. Intrinsic dimensionality estimation with optimally topology preserving maps. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(5):572–575, May 1998.

[17] H. Chen, P. Meer, and D.E. Tyler. Robust regression for data with multiple structures. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I:1069–1075, 2001.

[18] K.J. Cho and P. Meer. Image segmentation from consensus information. *Computer Vision and Image Understanding*, 68(1):72–89, October 1997.

[19] J. Costa and A.O. Hero. Geodesic entropic graphs for dimension and entropy estimation in manifold learning. *IEEE Trans. on Signal Process*, 52(8):2210–2221, Aug. 2004.

[20] I.J. Cox, S.L. Hingorani, S.B. Rao, and B.M. Maggs. A maximum-likelihood stereo algorithm. *Computer Vision and Image Understanding*, 63(3):542–567, May 1996.

[21] T. Cox and M. Cox. *Multidimensional Scaling*. Chapman & Hall, London, 1994.

[22] V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in Neural Information Processing Systems 15*, pages 705–712. MIT Press, Cambridge, MA, 2003.

[23] J. Dolan and E.M. Riseman. Computing curvilinear structure by token-based grouping. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages 264–270, 1992.

[24] D. Donoho and C. Grimes. Hessian eigenmaps: new tools for nonlinear dimensionality reduction. In *Proceedings of National Academy of Science*, pages 5591–5596, 2003.

[25] C.R. Dyer. Volumetric scene reconstruction from multiple views. In L.S. Davis, editor, *Foundations of Image Understanding*, pages 469–489. Kluwer, Dordrecht, 2001.

[26] J.H. Elder and S.W. Zucker. Computing contour closure. In *European Conf. on Computer Vision*, pages I:399–412, 1996.

[27] O.D. Faugeras and M. Berthod. Improving consistency and reducing ambiguity in stochastic labeling: An optimization approach. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 3(4):412–424, July 1981.

[28] O.D. Faugeras and R. Keriven. Variational principles, surface evolution, pdes, level set methods, and the stereo problem. *IEEE Trans. on Image Processing*, 7(3):336–344, March 1998.

[29] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM*, 24(6):381–395, 1981.

[30] W. Förstner. A framework for low level feature extraction. In *European Conf. on Computer Vision*, pages B:383–394, 1994.

[31] P.V. Fua. From multiple stereo views to multiple 3-d surfaces. *Int. Journ. of Computer Vision*, 24(1):19–35, August 1997.

[32] Y. Gdalyahu, D. Weinshall, and M. Werman. Self-organization in vision: Stochastic clustering for image segmentation, perceptual grouping, and image database organization. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(10):1053–1074, October 2001.

[33] D. Geiger, B. Ladendorf, and A. Yuille. Occlusions and binocular stereo. *Int. Journ. of Computer Vision*, 14(3):211–226, April 1995.

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

[35] J. Gluckman and S.K. Nayar. Rectifying transformations that minimize resampling effects. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I:111–117, 2001.

[36] J.Y. Goulermas and P. Liatsis. A collective-based adaptive symbiotic model for surface reconstruction in area-based stereo. *IEEE Trans. on Evolutionary Computation*, 7(5):482–502, 2003.

[37] A. Gove, S. Grossberg, and E. Mingolla. Brightness perception, illusory contours, and corticogeniculate feedback. *Visual Neuroscience*, 12:1027–1052, 1995.

[38] G.H. Granlund and H. Knutsson. *Signal Processing for Computer Vision*. Kluwer, Dordrecht,, December 1995.

[39] S. Grossberg and E. Mingolla. Neural dynamics of form perception: Boundary completion. *Psychological Review*, 92(2):173–211, 1985.

[40] S. Grossberg and D. Todorovic. Neural dynamics of 1-d and 2-d brightness perception: A unified model of classical and recent phenomena. *Perception and Psychophysics*, 43:723–742, 1988.

[41] G. Guy. *Inference of Multiple Curves and Surfaces from Sparse Data*. Ph.D. Thesis, University of Southern California, 1995.

[42] G. Guy and G. Medioni. Inferring global perceptual contours from local features. *Int. Journ. of Computer Vision*, 20(1/2):113–133, 1996.

[43] G. Guy and G. Medioni. Inference of surfaces, 3d curves, and junctions from sparse, noisy, 3d data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(11):1265–1277, November 1997.

[44] R.M. Haralick and L.G. Shapiro. The consistent labeling problem: Part i. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 1(2):173–184, April 1979.

[45] R.M. Haralick and L.G. Shapiro. The consistent labeling problem: Part ii. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2(3):193–203, May 1980.

[46] X. He and P. Niyogi. Locality preserving projections. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.

[47] F. Heitger and R. von der Heydt. A computational model of neural contour processing: Figure-ground segregation and illusory contours. In *Int. Conf. on Computer Vision*, pages 32–40, 1993.

[48] L. Herault and R. Horaud. Figure-ground discrimination: A combinatorial approach. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(9):899–914, September 1993.

[49] W. Hoff and N. Ahuja. Surfaces from stereo: Integrating feature matching, disparity estimation, and contour detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(2):121–136, February 1989.

[50] L. Hong and G. Chen. Segment-based stereo matching using graph cuts. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 74–81, 2004.

[51] B.K.P. Horn and B.G. Schunck. Determining optical flow. *AI*, 17(1-3):185–203, August 1981.

[52] R.A. Hummel and S.W. Zucker. On the foundations of relaxation labeling processes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 5(3):267–287, May 1983.

[53] S.S. Intille and A.F. Bobick. Disparity-space images and large occlusion stereo. In *European Conf. on Computer Vision*, pages B:179–186, 1994.

[54] H. Ishikawa and D. Geiger. Occlusions, discontinuities, and epipolar lines in stereo. In *European Conf. on Computer Vision*, pages I: 232–248, 1998.

[55] L. Itti and P. Baldi. A principled approach to detecting surprising events in video. In *Int. Conf. on Computer Vision and Pattern Recognition*, Jun 2005.

[56] D.W. Jacobs. Robust and efficient detection of salient convex groups. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(1):23–37, January 1996.

[57] A.K. Jain and R.C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Englewood Cliffs, NJ, 1988.

[58] H. Jin, S. Soatto, and A.J. Yezzi. Multi-view stereo beyond lambert. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 171–178, 2003.

[59] I.T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986.

[60] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(9):920–932, September 1994.

[61] E.Y. Kang. *Inference of 2D Layers from Uncalibrated Images*. Ph.D. Thesis, University of Southern California, 2003.

[62] J. Kang. *Tracking Multiple Objects from Single, Multiple, or Moving Uncalibrated Cameras*. Ph.D. Thesis, University of Southern California, 2004.

[63] S.B. Kang, R. Szeliski, and J. Chai. Handling occlusions in dense multi-view stereo. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I:103–110, 2001.

[64] G. Kanizsa. *Organization in Vision*. Praeger, New York, 1979.

[65] B. Kégl. Intrinsic dimension estimation using packing numbers. In *Advances in Neural Information Processing Systems 15*, pages 681–688. MIT Press, Cambridge, MA, 2005.

[66] K. Koffka. *Principles of Gestalt Psychology*. Harcourt, Brace, New York, 1935.

[67] W. Köhler. Physical gestalten. *W.D. Ellis (ed), A source book of Gestalt psychology (1950)*, pages 17–54, 1920.

[68] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions via graph cuts. In *Int. Conf. on Computer Vision*, pages II: 508–515, 2001.

[69] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *European Conf. on Computer Vision*, pages III: 82–96, 2002.

[70] K. Koster and M. Spann. Mir: An approach to robust clustering-application to range image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(5):430–444, May 2000.

[71] U. Köthe. Integrated edge and junction detection with the boundary tensor. In *Int. Conf. on Computer Vision*, pages 424–431, 2003.

[72] K.N. Kutulakos and S.M. Seitz. A theory of shape by space carving. *Int. Journ. of Computer Vision*, 38(3):199–218, July 2000.

[73] A. Laurentini. How far 3d shapes can be understood from 2d silhouettes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(2):188–195, February 1995.

[74] S. Lawrence, A. C. Tsoi, and A. D. Back. Function approximation with neural networks and local methods: Bias, variance and smoothness. In *Australian Conference on Neural Networks*, pages 16–21, 1996.

[75] Y.G. Leclerc. Constructing simple stable descriptions for image partitioning. *Int. Journ. of Computer Vision*, 3(1):73–102, May 1989.

[76] K.M. Lee, P. Meer, and R.H. Park. Robust adaptive segmentation of range images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(2):200–205, February 1998.

[77] M.S. Lee. *Tensor Voting for Salient Feature Inference in Computer Vision*. Ph.D. Thesis, University of Southern California, 1998.

[78] M.S. Lee and G. Medioni. Inferring segmented surface description from stereo data. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages 346–352, 1998.

[79] M.S. Lee, G. Medioni, and P. Mordohai. Inference of segmented overlapping surfaces from binocular stereo. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(6):824–837, June 2002.

[80] E. Levina and P. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems 17*, pages 777–784. MIT Press, Cambridge, MA, 2005.

[81] M. Lhuillier and L. Quan. Quasi-dense reconstruction from image sequence. In *European Conf. on Computer Vision*, pages II: 125–139, 2002.

[82] Z. Li. A neural model of contour integration in the primary visual cortex. *Neural Computation*, 10:903–940, 1998.

[83] M.H. Lin and C. Tomasi. Surfaces with occlusions from layered stereo. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 710–717, 2003.

[84] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3d surface reconstruction algorithm. *Computer Graphics*, 21(4):163–169, 1987.

[85] L.M. Lorigo, O.D. Faugeras, W.E.L. Grimson, R. Keriven, R. Kikinis, A. Nabavi, and C.F. Westin. Codimension-two geodesic active contours for the segmentation of tubular structures. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 444–451, 2000.

[86] D.G. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer, Dordrecht, June 1985.

[87] A. Luo and H. Burkhardt. An intensity-based cooperative bidirectional stereo matching with simultaneous detection of discontinuities and occlusions. *Int. Journ. of Computer Vision*, 15(3):171–188, July 1995.

[88] S. Mahamud, L.R. Williams, K.K. Thornber, and K. Xu. Segmentation of multiple salient closed contours from real images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(4):433–444, April 2003.

[89] D. Marr. *Vision*. Freeman Press, San Francisco, 1982.

[90] D. Marr and T.A. Poggio. Cooperative computation of stereo disparity. *Science*, 194(4262):283–287, October 1976.

[91] A. Massad, M. Babos, and B. Mertsching. Application of the tensor voting technique for perceptual grouping to grey-level images. In *DAGM*, pages 306–313, 2002.

[92] G. Medioni, M.S. Lee, and C.K. Tang. *A Computational Framework for Segmentation and Grouping*. Elsevier, New York, NY, 2000.

[93] G. Medioni and P. Mordohai. The tensor voting framework. *in Emerging Topics in Computer Vision, S.B. Kang and G. Medioni, editors*, 2004.

[94] G. Medioni and P. Mordohai. Saliency in computer vision. *in Neurobiology of Attention, L. Itti, G. Rees, and J. Tsotsos, editors*, 2005.

[95] G. Medioni, P. Mordohai, and M. Nicolescu. The tensor voting framework. *to appear in the Handbook of Computational Geometry for Pattern Recognition, Computer Vision, Neurocomputing and Robotics, E. Bayro-Corrochano, editor*, 2005.

[96] P. Meer and B. Georgescu. Edge detection with embedded confidence. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(12):1351–1365, December 2001.

[97] E. Mingolla, W.D. Ross, and S. Grossberg. A neural network for enhancing boundaries and surfaces in synthetic aperture radar images. *Neural Networks*, 12:499–511, 1999.

[98] S. Mitaim and B. Kosko. The shape of fuzzy sets in adaptive function approximation. *IEEE Trans. on Fuzzy Systems*, 9(4):637–656, 2001.

[99] T.M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[100] R. Mohan and R. Nevatia. Perceptual organization for scene segmentation and description. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(6):616–635, June 1992.

[101] P. Mordohai, M.S. Lee, and G. Medioni. Inference of segmented overlapping surfaces from binocular and multiple-view stereo. In *Third Workshop on Perceptual Organization in Computer Vision*, 2001.

[102] P. Mordohai and G. Medioni. Perceptual grouping for multiple view stereo using tensor voting. In *Int. Conf. on Pattern Recognition*, pages III: 639–644, 2002.

[103] P. Mordohai and G. Medioni. Dense multiple view stereo with general camera placement using tensor voting. In *Second International Symposium on 3-D Data Processing, Visualization and Transmission*, pages 725–732, 2004.

[104] P. Mordohai and G. Medioni. Junction inference and classification for figure completion using tensor voting. In *Fourth Workshop on Perceptual Organization in Computer Vision*, pages 56–56, 2004.

[105] P. Mordohai and G. Medioni. Stereo using monocular cues within the tensor voting framework. In *European Conf. on Computer Vision*, pages 588–601, 2004.

[106] P. Mordohai and G. Medioni. Unsupervised dimensionality estimation and manifold learning in high-dimensional spaces by tensor voting. *accepted at the Int. Joint Conf. on Artificial Intelligence*, 2005.

[107] J.M. Morel and S. Solimini. *Variational Methods in Image Segmentation*. Birkhauser, Boston, 1995.

[108] P.J. Narayanan, P.W. Rander, and T. Kanade. Constructing virtual worlds using dense stereo. In *Int. Conf. on Computer Vision*, pages 3–10, 1998.

[109] H. Neummann and E. Mingolla. Computational neural models of spatial integration in perceptual grouping. *From Fragments to Objects: Grouping and Segmentation in Vision, T.F.Shipley and P.J. Kellman, Editors*, pages 353–400, 2001.

[110] M. Nicolescu. *A Voting-Based Computational Framework for Visual Motion Analysis and Interpretation*. Ph.D. Thesis, University of Southern California, 2003.

[111] A.S. Ogale and Y. Aloimonos. Stereo correspondence with slanted surfaces: Critical implications of horizontal slant. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 568–573, 2004.

[112] Y. Ohta and T. Kanade. Stereo by intra- and inter-scanline search using dynamic programming. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 7(2):139–154, March 1985.

[113] M. Okutomi and T. Kanade. A multiple-baseline stereo. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(4):353–363, April 1993.

[114] S. Osher and R.P. Fedkiw. *The Level Set Method and Dynamic Implicit Surfaces*. Springer, Berlin Heidelberg New York, 2002.

[115] P. Parent and S.W. Zucker. Trace inference, curvature consistency, and curve detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(8):823–839, August 1989.

[116] T. Poggio and F. Girosi. Networks for approximation and learning. *Proc. of IEEE*, 78(9):1481–1497, 1990.

[117] T.A. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317:314–319, 1985.

[118] M. Pollefeys, R. Koch, and L.J. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Int. Conf. on Computer Vision*, pages 90–95, 1998.

[119] X. Ren and J. Malik. A probabilistic multi-scale model for contour completion based on image statistics. In *European Conf. on Computer Vision*, pages I: 312–327, 2002.

[120] L. Robert and R. Deriche. Dense depth map reconstruction: A minimization and regularization approach which preserves discontinuities. In *European Conf. on Computer Vision*, pages I:439–451, 1996.

[121] A. Robles-Kelly and E.R. Hancock. An expectation-maximisation framework for perceptual grouping. In *Inter. Workshop on Visual Form, LNCS 2059*, pages 594–605. Springer, Berlin Heidelberg New York, 2001.

[122] S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[123] S. Roy and I.J. Cox. A maximum-flow formulation of the n-camera stereo correspondence problem. In *Int. Conf. on Computer Vision*, pages 492–499, 1998.

[124] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2003.

[125] A. Saha, C.L. Wu, and D.S. Tang. Approximation, dimension reduction, and nonconvex optimization using linear superpositions of gaussians. *IEEE Trans. Computers*, 42(10):1222–1233, 1993.

[126] P.T. Sander and S.W. Zucker. Inferring surface trace and differential structure from 3-d images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(9):833–854, September 1990.

[127] T. D. Sanger. A tree-structured algorithm for reducing computation in networks with separable basis functions. *Neural Computation*, 3(1):67–78, 1991.

[128] R. Sara. Finding the largest unambiguous component of stereo matching. In *European Conf. on Computer Vision*, pages III: 900–914, 2002.

[129] R. Sara and R. Bajcsy. On occluding contour artifacts in stereo vision. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages 852–857, 1997.

[130] S. Sarkar and K.L. Boyer. A computational structure for preattentive perceptual organization: Graphical enumeration and voting methods. *IEEE Trans. on Systems, Man and Cybernetics*, 24:246–267, 1994.

[131] S. Sarkar and K.L. Boyer. Quantitative measures of change based on feature organization: Eigenvalues and eigenvectors. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages 478–483, 1996.

[132] L. K. Saul and S. T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

[133] E. Saund. Symbolic construction of a 2-d scale-space image. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(8):817–830, August 1990.

[134] E. Saund. Labeling of curvilinear structure across scales by token grouping. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages 257–263, 1992.

[135] E. Saund. Perceptual organization of occluding contours of opaque surfaces. *Computer Vision and Image Understanding*, 76(1):70–82, October 1999.

[136] E. Saund. Finding perceptually closed paths in sketches and drawings. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(4):475–491, April 2003.

[137] S. Schaal and C.G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084, 1998.

[138] D. Scharstein and R. Szeliski. Stereo matching with nonlinear diffusion. *Int. Journ. of Computer Vision*, 28(2):155–174, 1998.

[139] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. Journ. of Computer Vision*, 47(1-3):7–42, April 2002.

[140] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 195–202, 2003.

[141] B. Schölkopf, A.J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[142] S.M. Seitz and C.R. Dyer. Photorealistic scene reconstruction by voxel coloring. *Int. Journ. of Computer Vision*, 35(2):151–173, November 1999.

[143] J.A. Sethian. *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Materials Science*. Cambridge University Press, 1996.

[144] A. Shashua and S. Ullman. Structural saliency: The detection of globally salient structures using a locally connected network. In *Int. Conf. on Computer Vision*, pages 321–327, 1988.

[145] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000.

[146] G. Slabaugh, W. B. Culbertson, T. Malzbender, and R. Schafer. A survey of volumetric scene reconstruction methods from photographs. In *Volume Graphics 2001, Proc. of Joint IEEE TCVG and Eurographics Workshop*, pages 81–100. Springer, Berlin Heidelberg New York, 2001.

[147] J. Sun, H.Y. Shum, and N.N. Zheng. Stereo matching using belief propagation. In *European Conf. on Computer Vision*, pages II: 510–524, 2002.

[148] J. Sun, N.N. Zheng, and H.Y. Shum. Stereo matching using belief propagation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(7):787–800, July 2003.

[149] R. Szeliski and P. Golland. Stereo matching with transparency and matting. *Int. Journ. of Computer Vision*, 32(1):45–61, August 1999.

[150] R. Szeliski and D. Scharstein. Symmetric sub-pixel stereo matching. In *European Conf. on Computer Vision*, pages II: 525–540, 2002.

[151] C.K. Tang. *Tensor Voting in Computer Vision, Visualization, and Higher Dimensional Inferences*. Ph.D. Thesis, University of Southern California, 2000.

[152] C.K. Tang and G. Medioni. Inference of integrated surface, curve, and junction descriptions from sparse 3d data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(11):1206–1223, November 1998.

[153] C.K. Tang, G. Medioni, and M.S. Lee. N-dimensional tensor voting and application to epipolar geometry estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(8):829–844, August 2001.

[154] H. Tao, H.S. Sawhney, and R. Kumar. A global matching framework for stereo computation. In *Int. Conf. on Computer Vision*, pages I: 532–539, 2001.

[155] M.F. Tappen and W.T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. In *Int. Conf. on Computer Vision*, pages 900–907, 2003.

[156] C.J. Taylor. Surface reconstruction from feature based stereo. In *Int. Conf. on Computer Vision*, pages 184–190, 2003.

[157] Y.W. Teh and S. Roweis. Automatic alignment of local representations. In *Advances in Neural Information Processing Systems 15*, pages 841–848, Cambridge, MA, 2003. MIT Press.

[158] J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[159] D. Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(4):413–424, July 1986.

[160] W.S. Tong, C.K. Tang, and G. Medioni. Simultaneous two-view epipolar geometry estimation and motion segmentation by 4d tensor voting. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(9):1167–1184, September 2004.

[161] W.S. Tong, C.K. Tang, P. Mordohai, and G. Medioni. First order augmentation to tensor voting for boundary inference and multiscale analysis in 3d. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(5):594–611, May 2004.

[162] O. Veksler. Fast variable window for stereo correspondence using integral images. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 556–561, 2003.

[163] S. Vijayakumar, A. D'Souza, T. Shibata, J. Conradt, and S. Schaal. Statistical learning for humanoid robots. *Autonomous Robots*, 12(1):59–72, 2002.

[164] S. Vijayakumar and S. Schaal. Locally weighted projection regression: An o(n) algorithm for incremental real time learning in high dimensional space. In *Int. Conf. on Machine Learning*, pages I:288–293, 2000.

[165] J. Wang, Z. Zhang, and H. Zha. Adaptive manifold learning. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, Cambridge, MA, 2005. MIT Press.

[166] S. Wang, J. Wang, and T. Kubota. From fragments to salient closed boundaries: An in-depth study. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 291–298, 2004.

[167] Y. Wei and L. Quan. Region-based progressive stereo matching. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages I: 106–113, 2004.

[168] K.Q. Weinberger and L.K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition*, pages II: 988–995, 2004.

[169] M. Wertheimer. Laws of organization in perceptual forms. *Psycologische Forschung, Translation by W. Ellis, A source book of Gestalt psychology (1938)*, 4:301–350, 1923.

[170] L.R. Williams and D.W. Jacobs. Stochastic completion fields: A neural model of illusory contour shape and salience. *Neural Computation*, 9(4):837–858, 1997.

[171] L.R. Williams and K.K. Thornber. A comparison of measures for detecting natural shapes in cluttered backgrounds. *Int. Journ. of Computer Vision*, 34(2-3):81–96, August 1999.

[172] L.R. Williams and K.K. Thornber. Orientation, scale, and discontinuity as emergent properties of illusory contour shape. *Neural Computation*, 13(8):1683–1711, 2001.

[173] L. Xu, M. I. Jordan, and G. E. Hinton. An alternative model for mixtures of experts. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 633–640. MIT Press, Cambridge, MA, 1995.

[174] S.C. Yen and L.H. Finkel. Extraction of perceptually salient contours by striate cortical networks. *Vision Research*, 38(5):719–741, 1998.

[175] A.J. Yezzi and S. Soatto. Stereoscopic segmentation. In *Int. Conf. on Computer Vision*, pages I: 59–66, 2001.

[176] X.M. Yu, T.D. Bui, and A. Krzyzak. Robust estimation for range image segmentation and reconstruction. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(5):530–538, May 1994.

[177] Y. Zhang and C. Kambhamettu. Stereo matching with segmentation-based cooperation. In *European Conf. on Computer Vision*, pages II: 556–571, 2002.

[178] Z. Zhang and Y. Shan. A progressive scheme for stereo matching. In *Lecture Notes on Computer Science, 2018*, pages 68–85. Springer, Berlin Heidelberg New York, 2001.

[179] Z. Zhang and H. Zha. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM Journal of Scientific Computing*, 26(1):313–338, 2004.

[180] Z.Y. Zhang. Determining the epipolar geometry and its uncertainty: A review. *Int. Journ. of Computer Vision*, 27(2):161–195, March 1998.

[181] H.K. Zhao, S. Osher, and R. Fedkiw. Fast surface reconstruction using the level set method. In *UCLA Computational and Applied Mathematics Reports*, pages 32–40, 2001.

[182] C.L. Zitnick and T. Kanade. A cooperative algorithm for stereo matching and occlusion detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(7):675–684, July 2000.