ABSTRACT

Title of dissertation: A HOLISTIC APPROACH TO STRUCTURE FROM MOTION

Hui Ji, Doctor of Philosophy, 2006

Dissertation directed by: Professor Yiannis Aloimonos
Department of Computer Science

This dissertation investigates the general structure from motion problem. That is, how to compute in an unconstrained environment 3D scene structure, camera motion and moving objects from video sequences. We present a framework which uses concatenated feed-back loops to overcome the main difficulty in the structure from motion problem: the chicken-and-egg dilemma between scene segmentation and structure recovery. The idea is that we compute structure and motion in stages by gradually computing 3D scene information of increasing complexity and using processes which operate on increasingly large spatial image areas. Within this framework, we developed three modules. First, we introduce a new constraint for the estimation of shape using image features from multiple views. We analyze this constraint and show that noise leads to unavoidable mis-estimation of the shape, which also predicts the erroneous shape perception in human. This insight provides a clear argument for the need for feed-back loops. Second, a novel constraint on shape is developed which allows us to connect multiple frames in the estimation of camera motion by matching only small image patches. Third, we present

a texture descriptor for matching areas of extended sizes. The advantage of this texture descriptor, which is based on fractal geometry, lies in its invariance to any smooth mapping (Bi-Lipschitz transform) including changes of viewpoint, illumination and surface distortion. Finally, we apply our framework to the problem of super-resolution imaging. We use the 3D motion estimation together with a novel wavelet-based reconstruction scheme to reconstruct a high-resolution image from a sequence of low-resolution images.

A HOLISTIC APPROACH TO STRURCTURE FROM MOTION

by

Hui Ji

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2006

Advisory Committee:
Professor Yiannis Aloimonos, Chair/Advisor
Dr. Cornelia Fermüller, Co-Advisor
Professor Ralph Dubayah, Dean's Representative
Professor Larry Davis
Associate Professor David Jacobs

ACKNOWLEDGMENTS

I owe my gratitude to all the people who have made this thesis possible and because of whom my graduate experience has been one that I will cherish forever. First I'd like to thank my advisor, Professor Yiannis Aloimonos for giving me an invaluable opportunity to work on challenging and extremely interesting research projects over the past five years. He has always made himself available for help and advice on personal and research questions. It has been a pleasure to work with and learn from such an extraordinary individual.

I would also like to thank my co-advisor, Dr. Cornelia Fermüller. Without her extraordinary theoretical ideas and integral view on research, this thesis would have been a distant dream. I owe her lots of gratitude for her guidance and inspiration through my graduate study at University of Maryland. Life would have been very difficult without her.

Thanks are due to Professor Ralph Dubayah, Professor Larry Davis and Professor David Jacobs for agreeing to serve on my thesis committee and for sparing their invaluable time reviewing the manuscript.

My colleagues at the computer vision laboratory have enriched my graduate life in many ways and deserve a special mention. Abhijit S. Ogale, Jan Neumann, Patrick Baker, Justin Domke, Gutemberg Guerra, Alap Karapurkar. I would also want to thank my friends Richard Kollar, David Widemann, Ying-San Yeung, Ke-

Xue Liu and Zhi-Hui Tang for their friendship and companionship.

Finally, I would like to thank my family for their life-long love and support. I especially owe much to my wife, Ying, without whose love, encouragement and support, I would not have finished this thesis.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

1.1 Background

The field of Computer Vision has made great progress over the last few years. This is due to theories and algorithms as well as working applications related to the computation of scene models. A large amount of research has been devoted to the problem of structure from motion and the stereo correspondence problem. Solutions have been proposed to the segmentation problem using a variety of cues such as motion, texture, intensity or color. Other studies have developed constraints for the reconstruction of the scene from single images, so-called shape from X modules. Some of the techniques have been implemented in systems also, which have proven to work in constrained environments. The general reconstruction problem, that is the computation of scene models from image data acquired by a system (robot) moving in unconstrained environments, however, cannot be solved yet.

First, let us give a short discussion of the issues involved in *structure from motion*. The term has been used for the computations related to the reconstruction of the geometry from multiple images. These include the estimation of the structure of the scene and the 3D motion of the camera relative to the world from image sequences or multiple views. In a broader meaning, the problem also encompasses the estimation of the calibration parameters and the segmentation of the scene on

the basis of motion and structure. However, in the most common formulation only one rigid motion is considered; either the camera is moving in a static world or the whole scene is moving rigidly with the same motion. The literature distinguishes between two approaches, the discrete and the continuous ones, with major advances in different areas of application.

The so-called discrete techniques use views of the scene which are significantly separated in space [5, 6, 7, 8] and require corresponding image features in the different views, usually salient points [9] and lines [10] . Using the correspondences, the discrete rigid displacement of the camera between the views is computed. Then, the intersection of viewing rays provides the structure. This approach has been shown to be successful in many applications of 3D model reconstruction, where easily distinguishable features can be identified, for example in man-made structures, or when markers can be used, or correspondence can be established off-line. The limitations to this approach arise from the correspondence problem, which cannot be completely automated (without any prior knowledge of motion and structure). Without doubt, significant advances have been made on the correspondence problem using robust statistical techniques, most prominently, the RANSAC technique. But a general robust solution that would allow accurate estimation of structure and motion for general scenes does not seem to be possible.

The continuous techniques use as input video, that is sequences of images with small changes in viewing geometry between consecutive views [11, 12, 13]. From the time-varying image brightness pattern the image motion between pairs of views, or at least the image motion field of intensity gradients can be estimated rather easily,

although not very accurately. Using this, the 3D velocities and the structure of the scene (up to a scale factor) are estimated. Since biological systems use image motion, there is evidence for the success of such an approach in real world environments. Thus, naturally, image motion has been used in navigation tasks, such as 3D motion estimation [14], tracking [15, 16], segmentation and obstacle avoidance for robotic systems [17, 18].

Despite all the tremendous progress, neither image motion by itself nor correspondence by itself is sufficient to develop accurate human-like model building capabilities. It is clear that dense correspondence cannot be solved for general scenes and image motion cannot be estimated very accurately and does not allow for accurate localization of the discontinuities. Even worse, the structure from motion problem, which considers one rigid motion, is a simplification. Moving systems deal with a world of moving objects, and thus they also have to segment the moving objects from the static world. Detection of independently moving objects has to be solved together with structure and motion estimation, making the problem even harder. We call all these components together the "general structure from motion problem".

The classical approach to solving problems in Computer Vision, is much like in Engineering, a modular one. Given a complex problem, one breaks up the larger problem into multiple modules, which are then connected together to solve the original problem. In this spirit, the structure from motion problem has been addressed in three modules. First, the correspondence or image flow between frames is computed. In a second step, the geometric transformation, i.e. rigid motion between the

views is estimated. Then, in a last step, the camera views are placed in the world on the basis of the rigid motion estimate, and using the correspondence estimates, scene points are obtained through triangulation. In this way, a 3D model of the world is built.

Researchers in other vision fields have long pointed out that biological vision systems are not modular. They do not compute in a purely feed-forward fashion. Instead, it has been found in the Neurosciences that many neural connections are lateral and feed-back; modules higher in the hierarchy feed information to lower, earlier modules. Since nature is working rather well, this appears to be a better approach, and we should adapt it in Computer Vision. But the question still remains, what are these processes? To gain a better understanding, we will first take a look at the nature of the difficulties.

The main reason for structure reconstruction being difficult is that we have to segment the scene while we recover it. In terms of computations this means, in order to recover the scene structure, that is to estimate the parameters of a scene surface we need to know where the surfaces which belong to the same model are, or where the discontinuities are. But finding the discontinuities, that is segmenting the scene, requires some knowledge about the scene models. It is clear that there is an intricate interplay in the recovery and segmentation processes.

1.2 Overview of the thesis

Inspired from biological vision systems, a framework of great potential is solving the structure recovery and scene segmentation through computational loops (forward and backward processes). More specifically, 3D scene structure and segmentation are refined over stages. In every stage, using the information of the scene structure and camera motion computed in the previous stage, the 3D scene structure is re-estimated employing a more sophisticated scene model, and the segmentation is refined on the basis of information over larger spatial areas. A rough outline of the architecture of such a vision system for solving the general structure from motion problem is as follows:

1. Image motion from single flow fields and static cues (texture, intensity, color, edges) provides a first segmentation of the images. Then, image motion and matching over three (or more) frames provides occlusion and ordinal depth information, which is used to detect and locate independently moving objects.

2. Excluding the background from the independently moving objects in the scene, an initial estimate of the camera motion in the background is obtained. Multiple flow fields are then combined. The combination only requires the matching of image patches as opposed to pixel-wise matching. Then, from the matched frames, accurate camera motion and better 3D structure is estimated.

3. After obtaining accurate 3D motion and fairly good structure over multiple frames, we can now use many frames or frames significantly separated by

baseline to better segment, and refine the 3D structure estimation employing more sophisticated surface representations. The reason is that now, with some preliminary models of the scene, we can employ larger spatial areas and develop global spatial constraints. Thus, better matching can be achieved and more complicated structure models can be implemented.

In this thesis, we developed an elementary architecture with feed-back loops for solving the structure from motion in unconstrained scenes. The whole process begins with an "over" segmentation based on color information. Then, scene structure and segmentation are gradually improved over several stages. More specifically, an initial "over" segmentation, based on local constant flow, provides sufficient information for further refinement. At this stage we model the scene structure as a piece-wise planar surface. After that, a better 3D model is estimated by using multiple motion fields. Then, a better segmentation is obtained using the improved estimation of scene structure and camera motion. Thus in each stage, we obtain a more sophisticated scene model and better segmentation based on the input from previous stages.

In our implementation, accurate camera motion estimation and fairly good scene structure recovery are crucial to the performance of such a system. However, the accurate estimation of camera movement and scene structure is not feasible on the basis of two frames or consecutive frames with small baseline displacement only. On one hand, many researchers showed that camera translation is confused with camera rotation [19, 20, 14, 21]. Furthermore, the recovered scene structure is very sensitive to errors in camera motion estimation. On the other hand, our

studies (Chapter 1) on visual estimation process, through which humans perceive structure, shape and motion, show that statistical bias caused by noise in low level signal processing is unavoidable. All these arguments make the point that we have to do ego-motion estimation over multiple frames.

In order to combine the ego-motion estimation of multiple frames, we developed a novel constraint which allows simultaneous estimation of structure and motion from multiple frames. Instead of following the current approach of enforcing the structure to be the same, which is not computationally feasible, we enforce a weaker constraint. We require that the 3D shapes (surface normals) don't change under the moving camera. The computational advantage is that the shape of the scene is only related to the camera rotation, not to the camera translation. Thus, much more robust numerical algorithms can be employed to enforce this constraint.

Since multiple frames have to be linked together in the ego-motion estimation, some matching process is necessary. Luckily, our framework doesn't require matching of pixels, but only requires a matching of patches. Although matching patches is much easier than matching points, it is by no means an easy task, especially for two frames with large displacement. Thus, we developed the so-called MFS texture descriptor to match large image patches. Based on Fractal Geometry theory, the MFS (multifractal spectrum vector) texture descriptor is a framework which combines global statistics and local image features. The invariance of the MFS descriptor to various environmental changes, including changes of view-point, illumination and surface distortion, makes it a good candidate for the patch matching process, and also makes it a useful tool for many applications including texture retrieval and

classification.

A robust solution to the general structure from motion problem not only has important applications in visual navigation, but also has many other applications in AI, Graphics and Video Processing. The last part of the thesis is devoted to one important vision application called *super-resolution* imaging. Super-resolution imaging is the process of enhancing the image resolution of an image by utilizing a video sequence. Essentially such a process can be broken into two parts. One is the estimation of the sub-pixel displacement between multiple frames, so multiple low-resolution images can be aligned in the same coordinate system. The other is the signal processing problem, that is how to reconstruct the high resolution image from multiple aligned low-resolution images.

Our work on ego-motion estimation from multiple frames provides a good tool for accurate sub-pixel alignment. The remaining task is then a robust reconstruction of a high-resolution image from aligned low-resolution images. By modeling the image formation process using filter bank theory, we discovered that in general we couldn't perfectly reconstruct the high-resolution image. Instead only a blurred version can be perfectly recovered. Based on this discovery, we then designed an iterative reconstruction scheme which reconstructs the least- blurred high-resolution image under this theoretical limit. Therefore some un-necessary blurring operator is avoided. Our algorithm is based on perfect reconstruction filter bank theory (wavelet theory). A powerful wavelet-based denoising operator is then built into the algorithm with little extra computational expense. The new reconstruction algorithm not only is more robust to various noises (it reduces the blurring process),

but it also avoids the smoothing effect inherent to regular interpolation methods.

1.3 Organization of the thesis

Chapter 2 examines a problem inherent to 3D shape perception from multiple views. Noise in lower level processes causes bias in the estimation of higher level information. This bias phenomena predicts the underestimation of slant found in psychophysical and computational experiments. We first present a mathematical analysis of the estimation of 3D shape from motion and stereo using orientation disparity. This analysis shows that bias predicts the anisotropy in the perception of horizontal and vertical slant. Then, we demonstrate the bias by means of a new illusory display based on the analysis for the differential motion case. In the last part of Chapter 2, we discuss statistically optimal strategies for the estimation problem, which leads to possible avenues for visual systems to deal with noises.

In Chapter 3, we discuss a framework for solving the structure from motion problem in feed-back loops, with interplay between segmentation and ego-motion estimation. Our main technical contribution is a new module for the estimation of 3D motion. The new technique combines the information from multiple motion fields by enforcing a constraint on the surface normals (3D shape) of the scene in view. The fact that the shape vectors in the different views are related only by rotation can be formulated as a rank = 3 constraint. This constraint is implemented in an algorithm which solves 3D motion and structure estimation is a practical constrained minimization.

Chapter 4 introduces a new texture signature, based on fractal geometry theory, which is called the multifractal spectrum (MFS). It provides an efficient framework for combining global spatial invariance and local robust measurements. The MFS is invariant under the bi-Lipschitz map, which includes view-point changes and non-rigid deformations of the texture surface, as well as local affine illumination changes. The MFS constitutes a useful tool for patch matching over large displacements, which is necessary in the framework presented in Chapter 3 and also is useful for other applications such as texture retrieval and classification. Moreover, it is a useful tool for "place recognition" in robotics, where one needs to recognize places which previously have been seen under different viewpoints and different illuminations. We demonstrate its robustness to environmental changes in real settings, as well as its performance in comparison to other top methods in texture classification, which have much larger feature dimension.

In Chapter 5, the ego-motion estimation method over multiple frames is applied to the problem of super-resolution imaging, which requires multiple images to be aligned in the same coordinate system. Furthermore, we develop a complete super-resolution imaging system by introducing a new reconstruction scheme which reconstructs the high-resolution image from an aligned low-resolution image sequence. A new analysis of modeling image formation from the viewpoint of filter bank theory tells us that the best we can reconstruct is a HR image blurred by a specific low- pass filter. Based on this analysis we present a new wavelet-based iterative reconstruction algorithm which is very robust to noise without much extra computational expense.

Finally, Chapter 6 explores open problems and possible avenues of further research.

Chapter 2

Bias in shape perception

2.1 Overview

In this chapter we will examine the computations that allow us to recover the 3D shape of scene surfaces. These computations are often referred to as shape from X, because cues such as motion [9, 22, 23], stereo [24], texture [25, 26, 27, 28], shading [29, 30] and contours [31, 32] encode information from which the the shape of scene surfaces can be obtained. The recovery of 3D shape is difficult. The main reason is that we have to segment the scene while we recover it. It is clear that there is an intricate interplay in the recovery and segmentation processes, which we do not fully understand yet. But we have a good understanding of the computations of the inverse geometric image formation allowing for the recovery of shape [11]. That is, if we know where the continuous surfaces are (i.e. if we know the segmentation) and we know the surface property parameters, then we can obtain the shape.

However, computational experiments indicate that often the shape cannot be estimated correctly. For example when shape is estimated from multiple views, and even when the 3D viewing geometry is estimated correctly, the shape often is estimated incorrectly. It is known also in the psychophysical literature that human shape estimation is not veridical [31, 33]. For a variety of conditions and from a number of cues there is an underestimation of slant. Planar surface patches esti-

mated from texture [34, 27], contour [35], stereopsis [1, 36], and motion of various parameters [37] have been found to be estimated with smaller slant, that is, closer in orientation to a front-parallel plane than they are. In this chapter we are asking whether there are computational reasons for the mis-estimation.

In previous work it has been shown that there is a statistical problem with the estimation of image features [38, 39]. Here we extend these concepts to the visual shape recovery processes. We show that there is bias and thus consistent erroneous mis-estimation in the estimation of shape. The underlying cause is the well known statistical dilemma. Since image data is noisy, in order to estimate well, we would need to obtain the statistics of the noise. However, because of the complexity of the computations it is most often not possible to accurately estimate the noise parameters. The result is that bias cannot be avoided.

We investigated the effects of bias and found that it is consistent with the empirical findings. In particular, we show in this chapter that in the case of shape from motion for many 3D motions and for shape from stereo the bias causes an underestimation of slant. In [40, 41], we have demonstrated that bias also causes underestimation in shape from texture. Thus, we find, that one of the reasons for inaccuracy in shape estimation is systematic estimation error, i.e. bias, which affects machine vision as well as biological vision.

2.1.1 The main concept and what this thesis is about

The concepts underlying the statistical analysis are simple. The constraints in the recovery of shape can be formulated as linear equations in the unknown parameters. Thus we need to find the "best" solution to an over-determined equation system. We have equations in two unknown shape parameters $(x,\ y)$ of the form

$$a_{1_i}x + a_{2_i}y = b_i. \tag{2.1}$$

In our problem (x, y) encode the two components of the surface normal vector $N = (N_1, N_2, 1) = (x, y, 1)$. a_{1_i} and a_{2_i} are the observations, which are composed of multiple components. These components are the parameters of the image texture, that is the lines (edges) in the image and in the case of motion in addition the rotational parameters.

Consider, that we have n such equations, which we write in matrix for as

$$Ax = b \tag{2.2}$$

with A an n by 2 matrix, b an n dimensional vector and $x = (x, y)$ the 2 dimensional vector of unknowns. The observations a_{1_i}, a_{2_i} and b_i are always corrupted by errors. In the sequel unprimed letters are used to denote estimates, primed letters to denote the actual values, and δ's to denote errors, where $A = A' + \delta_A$ and $b = b' + \delta_b$. Thus (2.2) can be written as

$$(A' + \delta_A)x = (b' + \delta_b). \tag{2.3}$$

The most common choice to solving the system is by means of Least Squares (LS) estimation. Denoting by superscripts ^{T} the transpose and by $^{-1}$ the inverse of

a matrix, the solution of the LS estimator x_{LS} is characterized by

$$x_{LS} = (A^T A)^{-1} A^T b. \tag{2.4}$$

However, it is well known, that under noisy conditions this estimator generally is biased [42, 43].

What does this mean? Consider a problem for which you have a set of noisy measurements and you make an estimate. Then you choose another set of measurements and make another estimate. Continue many times. The expected value of your estimate is the average of your estimates in the limit. This expected value is not the true value. This is what we call statistical bias. Notice, there are no particular assumptions on the noise; it only needs to be symmetric around the true value.

Consider the simple case where all elements in δ_A and δ_b are independent identically distributed (i.i.d.) random variables with zero mean and variance σ^2. Then under quite general conditions the expected value $E(x_{LS})$ of the estimate amounts to [42]

$$E(x_{LS}) = x' - \sigma^2 (\lim_{n\to\infty} (\frac{1}{n} A'^T A'))^{-1} x', \tag{2.5}$$

which implies that x_{LS} is asymptotically biased [43, 44].

The bias here is $\sigma^2(\lim_{n\to\infty}(\frac{1}{n}A'^T A'))^{-1}x'$. Please note, the bias does not depend on n, the number of measurements, which only shows up for the purpose of normalization, because $A'^T A'$ is proportional to n.

An analysis of the bias term allows us to understand the errors in estimation. In general we can conclude that large variance in δ_A, an ill-conditioned A', or an

x' which is oriented close to the eigenvector of the smallest eigenvalue of $A'^T A'$ all could increase the bias and push the LS solution x_{LS} away from the real solution. Generally it leads to an underestimation of vector x.

The main reason for the bias are the errors in the explanatory variables A. If there are no errors in A, and there are errors only in b, least squares estimation is unbiased. Basically the bias originates from the quadratic term $(A'^T A')^{-1} = ((A+\delta A)^T(A+\delta A))^{-1}$ and can be obtained from a second order expansion of (2.4).

To give the reader an intuition about the bias, we illustrate it by means of the line equations. Referring to Figure 2.1, consider two lines $a_{1_1}x + a_{2_1}y = b_1$ and $a_{1_2}x + a_{2_2}y = b_2$ (the solid lines), which intersect in an acute angle. We want to find the intersection point. We don't have the exact lines, but we have noisy observations of these lines (the dashed lines). That is, the observed lines have orientations $(a_{1_1} + \delta_a, a_{2_1} + \delta_a)$ and $(a_{2_1} + \delta_a, a_{2_2} + \delta_a)$ and intercepts ($b_1 + \delta_b$ and $b_2 + \delta_b$) with δ_a and δ_b i.i.d. random variables of zero mean. (However, note δ_b, the error in the intersections with the y-axis, does not contribute to the bias.) The least squares solution to the intersection of the noisy lines is found as the point closest in distance (L^2 norm) to all the lines. This is not the correct intersection, but a point closer to the origin, because $\|(x, y)\|$ is underestimated.

The main assumption underlying our explanation is that our vision system uses least squares estimation. One may argue then that the bias is just an artifact of linear estimation. In Section 2.6.1 we discuss that this is not the case by taking a short digression into the statistical literature. In general classical estimation techniques will not be able to alleviate the bias significantly. The main reason is that usually

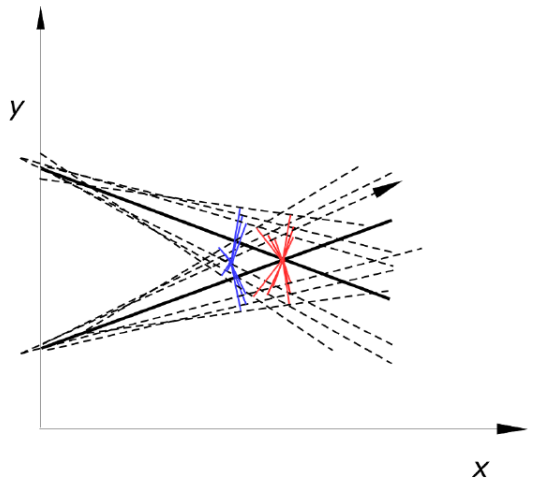


Figure 2.1: Illustration of the bias by means of line equations: The solid lines denote the true constraints, and the dashed lines denote the noisy observations. The LS solution to the intersection of the noisy lines is the point with closest distance (smallest sum of squares distance) to all the dashed lines. The distances to the true solution (denoted by red lines) are larger than the distances to the estimated solution (denoted by blue lines). Thus the estimated intersection point is the blue point, and not the correct red one. It is closer to the origin, because the bias leads to underestimation.

there is not enough data available. If in some cases there is data to extract statistics, the best thing to do is to partially correct the bias, and this does not change the form of the bias. In section 2.6.2 we present our hypothesis that the human visual system actually does partial correction and we show experiments that support this point of view.

The question is then, does this insight about the bias tell us anything about the way the human system estimates or about how theoretically machine vision should estimate shape? Section 2.6.3 discusses these issues. First, we could do better in the geometrical estimation problems by using color information. Three color pixels usually do not contain more information than one gray value pixel from a geometrical point of view. They do, however, contain statistical information, which we could exploit to improve the estimation. Second, in order to do good statistics, whatever we do, we do it better with larger amounts of data. We can only use large amounts of data if we have models of the scene. Thus the bias makes a computational argument for the need to solve the estimation of image features, motion, structure, shape, and the segmentation in feed-back loops. Having estimates about the 3D motion and the structure, we can segment the scene and apply our methods to larger amounts of data.

A number of previous studies have analyzed the statistics of visual processes. In particular, [45] discussed bias for some visual recovery processes. A few studies analyzed the statistics of structure from motion [46, 19, 12, 47]. However, these analyses stayed at the general level of parameter estimation; no one has shown before the effects on the estimated shape.

2.2 Shape from multiple views

The 3D shape of a surface patch is described by the surface normal. In the literature on multiple view geometry, 3D shape is considered a by-product of the estimation of structure (the 3D coordinates of the scene). 3D shape is obtained as the spatial derivative of structure. However, one could estimate the 3D shape directly from the image texture without estimating the structure and without knowing the displacement (translation) between the cameras. Let us describe a surface patch by its local tangent plane. Estimating the structure of the patch means estimating three parameters: two parameters for the surface normal and one parameter for the depth (how far is the surface patch?). There is reference to this idea in the psychophysical stereo literature, where it is referred to as estimation from *orientation disparity*. The geometric constraints are explained next.

Consider two views of a planar patch separated only by translation T as in stereo (Fig. 2.2). Assume the patch contains a line L (an edge due to texture). The projections of this line on the two views with centers O and $\tilde{O}$ are then the two image lines ℓ and $\tilde{\ell}$. Usually a line in the image is described by an equation of the form $ax + by + c = 0$, which we can also write as $(a, b, c) \cdot (x, y, 1) = 0$. This means, that the vector (a, b, c) is perpendicular to any vector $(x, y, 1)$ representing the points on the line, assuming that the image is at distance one from the center O. Thus, such a line can be represented by the vector (a, b, c), and we normalize it to have a unit z-component as $\ell = (\frac{a}{c}, \frac{b}{c}, 1)$. Geometrically, vector ℓ is perpendicular to the plane through O and L. Similarly, vector $\tilde{\ell}$ is perpendicular to the plane through $\tilde{O}$ and

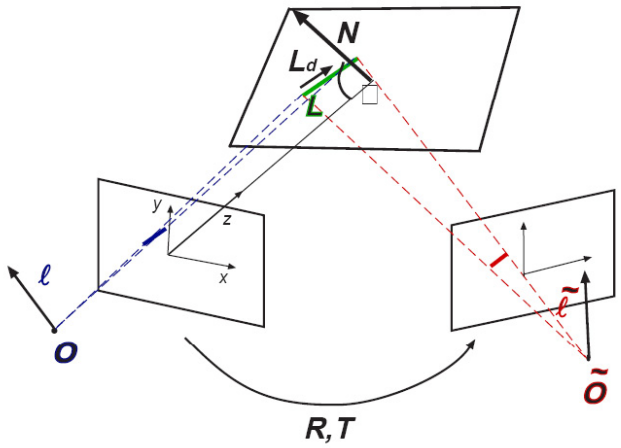


Figure 2.2: Orientation disparity constraint in stereo: A line L in space is projected on the two views as ℓ and $\tilde{\ell}$. The representation for ℓ and $\tilde{\ell}$ are the vectors normal to the planes defined by the line in space and each of the centers O and $\tilde{O}$. Since ℓ and $\tilde{\ell}$ are both perpendicular to the line L, it follows that $\ell \times \tilde{\ell}$ is parallel to the line L. N is normal to the plane containing L. Thus we have $(\ell \times \tilde{\ell}) \cdot N = 0$.

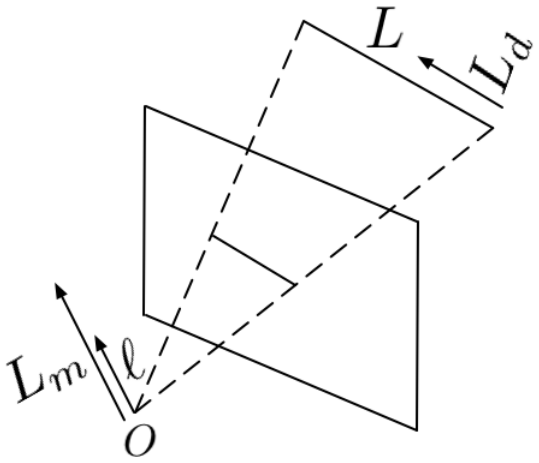


Figure 2.3: Plücker representation of a line L as (L_d, L_m) and its image ℓ.

L. The line L is perpendicular to vector ℓ, and it is also perpendicular to vector $\tilde{\ell}$. Thus the cross-product vector $\ell \times \tilde{\ell}$ is parallel to the line L. Since the surface normal N is perpendicular to the patch, which includes the line L, we obtain

$$(\ell \times \tilde{\ell}) \cdot N = 0, \tag{2.6}$$

where "$\cdot$" denotes the scalar product. This is the linear equation that we use to estimate N.

2.2.1 Analysis of Equation (2.6)

In order to analyze the bias and predict parametric influences we need to relate the image lines to the line in 3D. To facilitate the analysis in the following discussion, we present 3D lines by their *Plücker coordinates*. A line L in 3D space, which is a four-dimensional object, is then represented by the two vectors (L_d, L_m). L_d is a unit vector parallel to the line L in space, and thus it denotes its orientation. L_m, which is called the moment of the line, is a vector perpendicular to the plane through L and the origin O with value the distance of L from O. L_m is parallel to ℓ and perpendicular to L_d (Fig. 2.3). Then the line coordinates in the two views are related as (see Appendix 2.7.1)

$$\begin{aligned} \widetilde{L_d} &= L_d \\ \widetilde{L_m} &= L_m + T \times L_d, \end{aligned}$$

where T is the translation between two views. Thus

$$\ell \times \tilde{\ell} = \frac{L_m}{\hat{z} \cdot L_m} \times \frac{\widetilde{L_m}}{\hat{z} \cdot \widetilde{L_m}} = \frac{L_m}{\hat{z} \cdot L_m} \times \frac{(L_m + T \times L_d)}{\hat{z} \cdot \widetilde{L_m}} = -\frac{(L_m \cdot T) L_d}{(\hat{z} \cdot L_m)(\hat{z} \cdot \widetilde{L_m})}.$$

For comparing the different configurations, we will simply use the projective relation (with $\sim$ denoting equality up to a scale factor)

$$\ell \times \tilde{\ell} \sim (T \cdot L_m) L_d. \tag{2.7}$$

2.3 Shape from stereo

Let us write the estimation equation (2.6) as

$$e \cdot N = 0,$$

with $e = \ell \times \tilde{\ell}$. Let $N = (N_1, N_2, 1)$ then be the surface normal, and let $\{\ell_i = (a_i, b_i, 1)\}$ denote the lines in the left image and $\{\widetilde{\ell_i} = (\tilde{a}_i, \tilde{b}_i, 1)\}$ the corresponding lines in the right image. Substituting these coordinates into equation (2.6), we obtain for every observed line an equation in the two parameters (N_1, N_2) of the form

$$(e_{1_i}, e_{2_i}) \cdot (N_1, N_2) = -e_{3_i}, \tag{2.8}$$

where

$$\begin{cases} e_{1_i} = b_i - \tilde{b}_i \\ e_{2_i} = -a_i + \tilde{a}_i \\ e_{3_i} = a_i \tilde{b}_i - \tilde{a}_i b_i \end{cases}$$

The line measurements (a_i, b_i) are always corrupted by noise in practice. Let the noise $\delta a_i = a_i - a_i', \delta b_i = b_i - b_i'$ and $\delta \tilde{a}_i = \tilde{a}_i - \tilde{a}_i', \delta \tilde{b}_i = \tilde{b}_i - \tilde{b}_i'$ be independent random variables with zero mean and covariance δ^2. Thus $e_{k_i}, k = 1, 2, 3$ are also corrupted by noise. Then we have:

$$(e_{1_i}' + \delta e_{1_i}) N_1 + (e_{2_i}' + \delta e_{2_i}) N_2 = -(e_{3_i}' + \delta e_{3_i}). \tag{2.9}$$

Let E, E' and δE denote the $n \times 2$ matrices incorporating the n measurements e_{1_i} and e_{2_i} and G, G' and δG denote the $n \times 1$ matrices incorporating the e_{3_i}. Then the estimation of $x = (N_1, N_2)$ is obtained by solving the equation

$$\begin{aligned} Ex &= G \quad \text{or} \\ (E' + \delta E)x &= G' + \delta G. \end{aligned} \tag{2.10}$$

Assuming that the errors are much smaller then the real values, we develop the LS solution of x in a second order Taylor expansion and obtain as an approximation for the estimate of x (see appendix 2.7.2):

$$E(x) = x' - 2n\delta^2 M'^{-1}x' = (I - 2n\delta^2 M'^{-1})x' \quad \text{with} \quad M' = E'^T E'. \tag{2.11}$$

Since M' is a positive definite matrix, so is M'^{-1}. Considering the perturbation $2n\delta^2 M'^{-1}$ being small, we have $\|I - 2n\delta^2 M'^{-1}\|_2 < 1$. Then we conclude that

$$\|E(x)\| = \|(I - 2n\delta^2 M' x^{-1} x'\| < \|Ix'\| = \|x'\|, \tag{2.12}$$

i.e. generally vector x is underestimated, and the degree of underestimation highly depends on the structure of matrix M'.

2.3.1 The effects on slant

The slant σ is the angle between the surface normal and the negative z-axis (0^o slant corresponds to a plane parallel to the image plane, 90^o to a plane that contains the optical axis) and the tilt τ is the angle between the direction of the projection of the surface normal onto the image plane and the x-axis (see Figure 2.4). Using these coordinates $N = (\cos\tau \tan\sigma, \sin\tau \tan\sigma, 1)$.

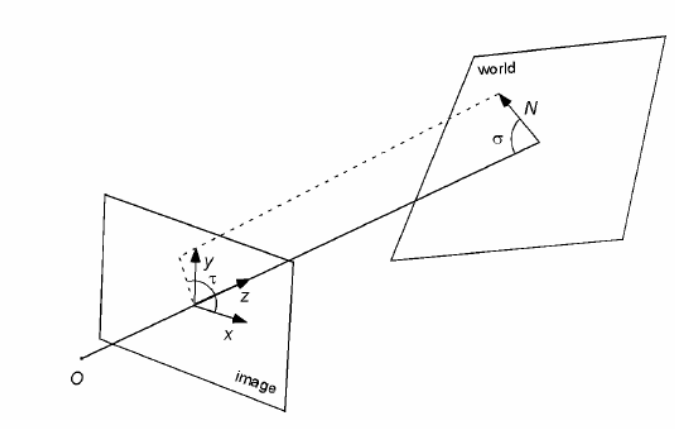


Figure 2.4: The slant of a surface is the angle σ between the negative z-axis and the surface normal, the tilt τ is the angle between the projection of the surface normal onto the image plane and the x-axis.

We know from the previous section (eq. (2.12)) that $\|x\|$ is underestimated. Since $\sigma = \cos^{-1}(1 + \|x\|)^{-\frac{1}{2}}$ is a strictly increasing function of $\|x\|$, by linear approximation, the slant σ is also underestimated. That is,

$$E(\sigma) < \sigma', \tag{2.13}$$

i.e. the expected value of the estimated slant is smaller than the actual value.

The degree of underestimation can be found by analyzing matrix M', or more specifically, the inverse of matrix M'. The inverse of a matrix can be written as its adjoint over its determinant, i.e. $M'^{-1} = \frac{adj(M')}{det(M)}$, which shows that large bias results from a small determinant and an x' close to the smaller eigenvalue of M'.

2.3.2 Anisotropy in the perception of stereoscopic slant

An interesting phenomenon in stereoscopic vision is the anisotropy in the perception of slanted (or tilted) planes. A surface slanted about the horizontal axis is estimated much easier and more accurately than a surface slanted about the vertical axis [48, 1, 49]. In both cases there is an underestimation of slant, but it is much larger for slant about the vertical. [48] argued that this effect is due to orientation disparity, which generally (assuming the texture lines to be mostly vertical and horizontal) is smaller for surfaces slanting about the vertical. However, as shown in [1], the effect also exists, even though in weaker form, when the texture is made up of lines oriented at 45°. For such a configuration the orientation disparity in the two differently slanted planes should be the same. From this result, the authors argue that orientation disparity can not be the cause. We now show that bias in orientation disparity can account for this anisotropy.

2.3.3 Analysis of stereoscopic slant

Consider a front-parallel plane which is textured with two sets of orthogonal line segments of orientation θ and $\frac{\pi}{2} + \theta$ (see Fig. 2.5a). When we slant this plane with angle σ about the vertical axis (Fig. 2.5b) the corresponding surface normal $N_v = (x_v, 1) = (\tan\sigma, 0, 1)$. When we slant the front-parallel plane about the horizontal axis (Fig. 2.5c), the surface normal is $N_h = (x_h, 1) = (0, \tan\sigma, 1)$. Substituting the components of e into eq. (2.11) and relating x to the slant σ we obtain the estimated slant of the vertical and horizontal tilted plane, σ_v and σ_h, (

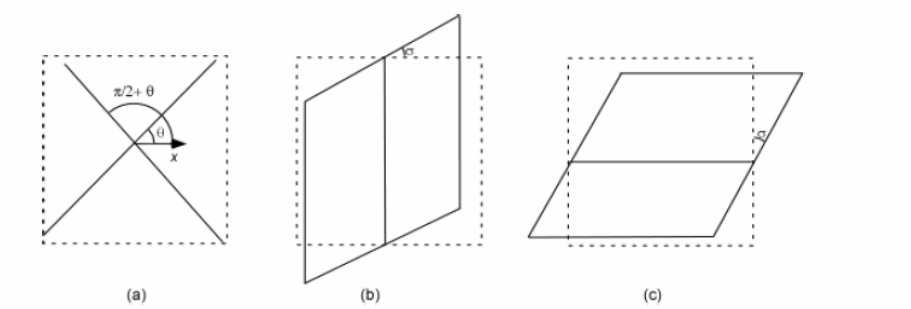


Figure 2.5: (a) In the fronto-parallel setting the orientation of the line elements on the plane is θ. The plane is rotated (slanted) by angle σ about the (b) vertical axis and (c) about the horizontal axis.

see appendix 2.7.3):

$$E(\sigma_v) = \sigma - \delta^2 \frac{\sin 2\sigma}{2} \frac{nE(\sum e_{2_v}'^2)}{E(det(M_v'))} = \sigma - \delta^2 \frac{\sin 2\sigma}{2} C_v \tag{2.14}$$

$$E(\sigma_h) = \sigma - \delta^2 \frac{n \sin 2\sigma}{2} \frac{E(\sum e_{1_h}'^2)}{E(det(M_h'))} = \sigma - \delta^2 \frac{\sin 2\sigma}{2} C_h. \tag{2.15}$$

In the equations above the effects of $M'^{-1}x' = \frac{adj(M')x'}{det(M')}$ show up as the terms C_v and C_h. The terms $E(\sum e'^2_{2_v})$ and $E(\sum e'^1_{2_h})$ originate from $adj(M')x'$, and the terms $E(det(M_v')$ and $E(det(M_h')$ are the determinants of M' . C_v and C_h determine the degree of underestimation. The larger they are the more the underestimation will be, and their ratio is a measure of the relative error.

Denoting the displacement between the cameras as t in appendix 2.7.3 we derive:

$$C_v = \frac{1}{t^2} \frac{(\sin^4\theta + \cos^4\theta)}{\cos^4\sigma(\sin^2\theta\cos^2\theta)} \tag{2.16}$$

$$C_h = \frac{2}{t^2 \cos^2\sigma}, \tag{2.17}$$

and thus their ratio amounts to:

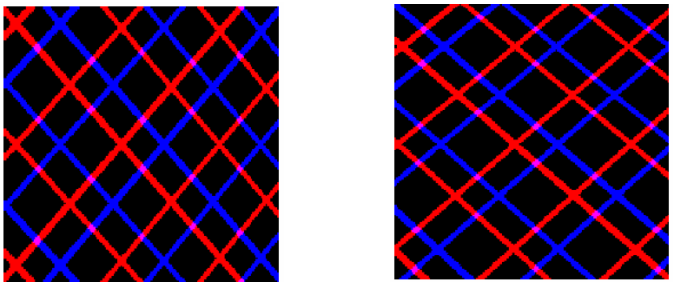
$$\frac{C_v}{C_h} = \frac{1}{\cos^2\sigma}\frac{\tan^2\theta + \cot^2\theta}{2} > 1. \tag{2.18}$$

Let us get an intuition for the above equations. The differences in the bias can be understood from the relation

$$e \sim (T \cdot L_m)L_d.$$

The two parameters involved in eq. (2.18) are σ, the slant (or rotation) of the surface and θ, which defines the orientation of the image lines (Figure 2.5). The effect of σ enters through the determinants. A slant about the vertical axis shortens the x component of L_m and L_d by a multiplicative factor of $\cos(\sigma)$. A slant about the horizontal axis effects in the same way the y component of L_m and L_d. Since the translation is parallel to the x-axis, i.e. $T = (t, 0, 0)$ and the product $(T \cdot L_m)$ has only the x-component of L_m, there is more shortening effect in the determinant of the vertically slanted plane, and the ratio of $\frac{C_v}{C_h}$ is $\frac{1}{\cos^2\sigma}$.

The effect of θ enters due to $adj(M')x'$: The components involving θ are the same for both matrix M_v and M_h. (Only the components due to slant are different). Because of the product $T \cdot L_m$, for both matrices there is a smaller θ component parallel to the x-axis than parallel to the y-axis (smaller e'^2_1 than e'^2_2. In other words, the smaller eigenvalue component is closer to the x axis than to the y axis,). $x_v = (\tan\sigma, 0)$ is parallel to the x-component and $x_h = (0, \tan\sigma)$ is parallel to the y-component, and thus there is larger bias for vertical slant. The ratio of the two terms amounts to $\frac{\tan^2\theta+\cot^2\theta}{2}$.



(a) (b)

Figure 2.6: Stereoscopic images of a plane slanted by 30° about the vertical (a) and horizontal (b) axes. View the images with red-blue glasses with the red on the right.

Now, using (2.18) we can predict the perception of slanted planes [48, 1]. For a pattern with vertical and horizontal line segments ($\theta \approx 90^o$) the ratio $\frac{C_v}{C_h} >> 1$. This predicts the estimation for the plane slanted about the vertical to be significantly worse than for the plane slanted about the horizontal. When the line segments are at 45° the bias is still larger for the slant about the vertical since $\frac{C_v}{C_h} = \frac{1}{\cos^2 \sigma} > 1$. Thus the perception for the slant is still predicted to be more erroneous, but with much less anisotropy.

2.3.4 Experiments and predictions

We created stereograms on a computer display as follows: A plane textured with lines in two orthogonal directions was slanted in space about the vertical and horizontal axes with the slant in the range of 0° to 55°, and its images were created by projection. In order to keep the number of lines constant (between 4 and 5 lines in one direction) we zoomed in. We tested two line orientations, a pair with 45° and 135° degrees and a pair with 30° and 120°. Three observers were shown first

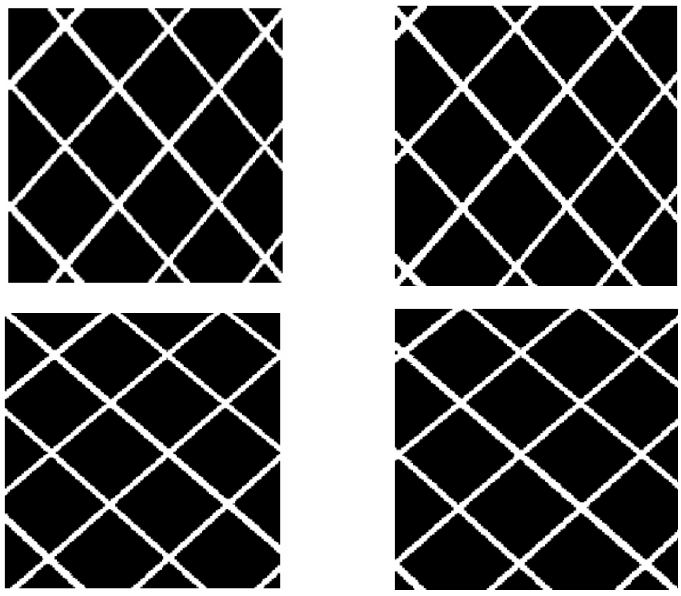
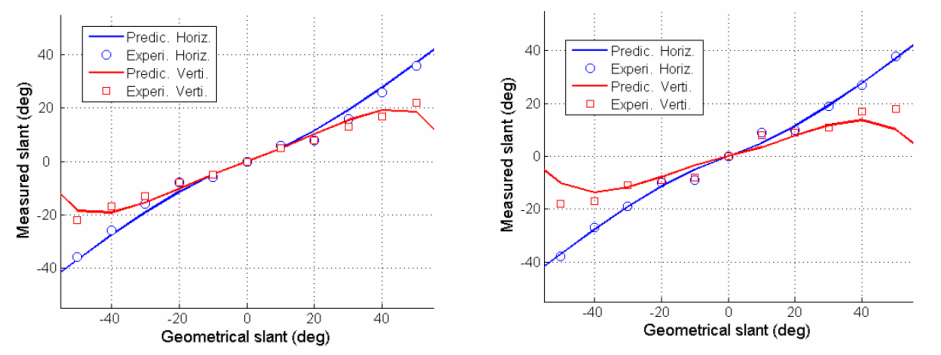


Figure 2.7: Free fusion versions of 2.6 for uncrossed viewing.

a random sequence of horizontally, then a sequence of vertically slanted planes and asked to adjust a cardboard on the desk next to the screen to denote the perceived slant. Figures 2.6 and 2.7 show the stereograms for 30° slant and the lines oriented at 45° (and 135°).

Figure 2.8 plots the measurement along with our predictions. The data points shown are the mean values over all all trials and all three subjects. The standard deviation was about 20%. Mostly one of our subjects was much more accurate in his estimates than the the other two. Fig. (2.9) shows the data of the individual subjects for one configuration.

As can be seen from Fig. 2.8 the predictions model the data very well within this range of angles. We should note that we do not attempt to model larger slants. First, we think that because at larger angles texture is very compressed, additional information about the texture distortion and the vanishing lines becomes important,



(a) (b)

Figure 2.8: Our experiments: Predictions and measurements for textures oriented at (a) 45° and (b) 30°.

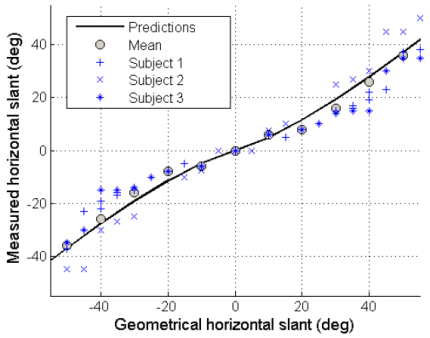


Figure 2.9: Measurements from the three subjects for a plane slanted about the horizontal axis and textured with lines of 45° orientation.

which has more influence on the perception than the bias. Second, our equations are approximations which are valid only for smaller angles. In particular, the Taylor expansion in (2.46) is not a justified for larger angles. Our measurements indicate the same general behavior as found in [48, 1]. To allow for comparison we show data of [1] for one of their subjects. Referring to Figure 2.10, one can see that Mitchison and McKee found a worse estimation for vertically slanted planes, and they found a nearly linear measurement function.

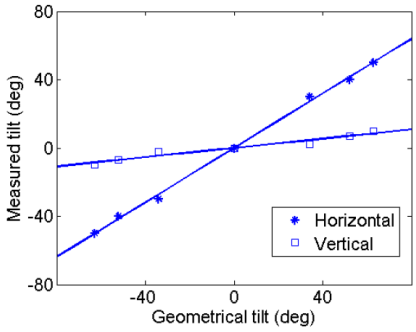


Figure 2.10: Slant perceived by one of the subjects in [1] for 45° oriented texture lines.

2.4 Shape from motion

In the case of differential motion, when the camera (or eye) moves with instantaneous translational velocity t and rotational velocity ω, (2.6) takes a similar form. If N is the normal of a plane containing a line with image ℓ and temporal derivative $\dot{\ell}$ the estimation equation (2.6) becomes (see appendix 2.7.1)

$$\begin{aligned} (\ell \times (\dot{\ell} - \omega \times \ell)) \cdot N &= 0 \quad \text{or} \\ e \cdot N &= 0 \end{aligned} \tag{2.19}$$

with $e = \ell \times (\dot{\ell} - \omega \times \ell)$. (see appendix 2.7.1).

Let $\{\ell_i = (a_i, b_i, 1)\}$ denote the lines on the plane, and $\{\dot{\ell}_i = (\dot{a}_i, \dot{b}_i, 0)\}$ denote the motion parameters of the lines ℓ_i. Then in the prime equations

$$(e_{1_i}, e_{2_i}) \cdot (N_1, N_2) = -e_{3_i}, \tag{2.20}$$

the parameters are:

$$\begin{cases} e_{1_i} = -\dot{b}_i + (-(1+b_i^2)\omega_1 + a_i b_i \omega_2 + a_i \omega_3) \\ e_{2_i} = \dot{a}_i + (a_i b_i \omega_1 - (1+a_i^2)\omega_2 + b_i \omega_3) \\ e_{3_i} = -(\dot{a}_i b_i - \dot{b}_i a_i) + (a_i \omega_1 + b_i \omega_2 - (a_i^2 + b_i^2)\omega_3). \end{cases}$$

There is noise in the measurements of the line locations and the measurements of the line movement. Let the error random variables be $\delta a_i = a_i - a_i'$ and $\delta b_i = b_i - b_i'$ with expected value 0 and variance δ_1^2 and $\delta \dot{a}_i = \dot{a}_i - \dot{a}_i'$ and $\delta \dot{b}_i = \dot{b}_i - \dot{b}_i'$ with expected value 0 and variance δ_2^2. Then from the second order Taylor expansion of the LS solution we obtain the expected value of $x = (N_1, N_2)$ (see appendix 2.7.2) as

$$E(x) = x' - M'^{-1}(\delta_2^2 D' + \delta_1^2 F')x' - M'^{-1}\delta_1^2 H', \tag{2.21}$$

where

$$D' = \begin{pmatrix} n & 0 \\ 0 & n \end{pmatrix}, \quad H' = \omega_3 \sum_i^n \begin{pmatrix} \omega_1(6b_i'^2 + c_i' + 3) \\ \omega_2(6a_i'^2 + c_i' + 3) \end{pmatrix},$$

$$F' = \sum_i^n \begin{pmatrix} 6b_i'^2\omega_1^2 + c_i'\omega_2^2 + \omega_3^2 + 2\omega_1^2 & 2c_i'\omega_1\omega_2 + 2\omega_1\omega_2 \\ 2c_i'\omega_1\omega_2 + 2\omega_1\omega_2 & c_i'\omega_1^2 + 6a_i'^2\omega_2^2 + \omega_3^2 + 2\omega_2^2 \end{pmatrix}$$

with $c_i' = a_i'^2 + b_i'^2$.

For the case when rotation around the z-axis can be ignored (i.e, $\omega_3 = 0$) equation (2.21) simplifies to

$$E(x) = (I - M'^{-1}(\delta_2^2 D' + \delta_1^2 F'))x' = (I - \delta_A)x'. \tag{2.22}$$

D' and F' are positive definite matrices and the perturbations δ_1 and δ_2 are small. Thus δ_A is also a positive definite matrix, and by the same arguments as in the case of stereo, the slant can shown to be underestimated.

To show the degree of underestimation, next we will analyze the determinant of matrix M'; the smaller the determinant the larger the underestimation. The velocity of rotation also contributes to the magnitude of the bias as can be seen from matrix F'; larger velocity more bias.

2.4.1 Predictions and illusory display

To say more about the dependence of slant estimation on the texture distribution we use the relation (2.31):

$$e \sim (t \cdot \ell) L_d.$$

Let us consider a slanted plane with a texture of two major directional components. Let the directional components be $L_{d_1} = (\cos\tau_1 \sin\sigma_1, \sin\tau_1 \sin\sigma_1, \cos\sigma_1)$ and $L_{d_2} = (\cos\tau_2 \sin\sigma_2, \sin\tau_2 \sin\sigma_2, \cos\sigma_2)$. That is σ_1 and σ_2 are the angles between the texture lines on the world and the negative z-axis, and τ_1 and τ_2 are the angles between the projections of the texture lines on the image plane and the x-axis (see Figure 2.11). The determinant $det(M)$ of M amounts to (appendix 2.7.4)

$$det(M) \;=\; \sum (t \cdot \ell_{1_i})^2 \sum (t \cdot \ell_{2_i})^2 (\sin\sigma_1 \sin\sigma_2 \sin(\tau_1 - \tau_2))^2. \qquad (2.23)$$

In this equation the term $\sum (t \cdot \ell_{1_i})^2 \sum (t \cdot \ell_{2_i})^2$ is due to the products $t \cdot \ell_1$ and $t \cdot \ell_2$ in the e's and the term $(\sin\sigma_1 \sin\sigma_2 \sin(\tau_1 - \tau_2))^2$ is due to L_{d_1} and L_{d_2}.

Using our model we can predict the findings from experiments in the literature. In ([37]) it has been observed that an increase in the slant of a rotating surface causes increased underestimation of the slant. This can be understood from the change of

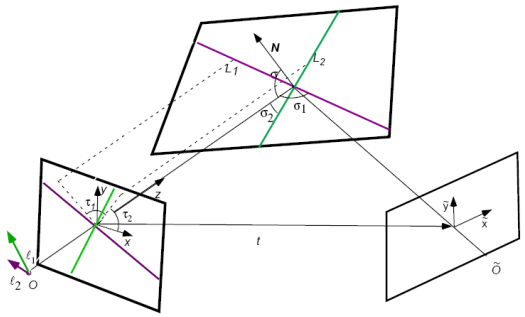


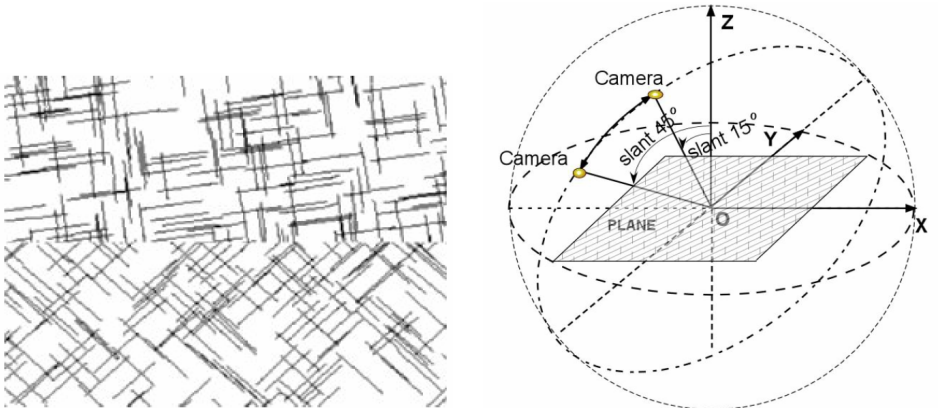
Figure 2.11: A plane with a texture of two orientations L_{d_1} and L_{d_2} is imaged under motion. σ_1 and σ_2 are the angles between the texture lines on the world and the negative z-axis, and τ_1 and τ_2 are the angles between the projections of the texture lines on the image plane and the x-axis.

L_d in eq. (2.23). Intuitively, larger slant causes an increase in the z- and decrease in the x- and y-component of L_d and thus a smaller $det(M)$. By our formula in eq. (2.23) this is manifested in the factor $\sin(\sigma_1)\sin(\sigma_2)$, where σ_1 and σ_2 are the the angles between the directions of the line in space and the negative z-axis. Unless, they are 0 degree, these values decrease with an increase of the slant of the plane, and this leads to a smaller $det(M)$. Hence, we get a larger error towards underestimation of the slant.

To demonstrate the predictive power of the model we created two illusory displays. In the first one, the scene consists of a plane with two textures, one in the upper half, the other in the lower half. Figure 2.12a shows the plane when it is parallel to the screen. The texture in the upper part consists of two line clusters with slope 8^o and 98^o. The lower part has two lines clusters with slope 45^o and 135^o. A video was created for the camera orbiting the sphere along a great circle in the yz-plane as

shown in Figure 2.12b – that is the camera translates and rotates such that it keeps fixating at the center. At the beginning of the motion, the slant of the plane with respect to the camera is 15^o, at the end it is 45^o. The image sequence can be seen from the website (http://www.cfar.umd.edu/users/fer/optical/Newsite/shape/video.avi). As can be experienced, it creates the perception of the plane being segmented into two parts, with the upper part having a much smaller slant. This is predicted by the biases in the different textures. For the upper texture the bias is much larger, thus producing larger underestimation of the slant, and the underestimation gets worse as the slant increases. The difference in the values of the bias can be understood from the difference in the values for $(t \cdot \ell)$ (the term $\sum(t \cdot \ell_{1_i})^2 \sum(t \cdot \ell_{2_i})^2$ in (2.23)). t is nearly parallel to the y-axis, and thus $t \cdot \ell$ is close to zero for vertical texture lines, making the determinant very small. In a second display (www.cfar.umd.edu/users/fer/optical/Newsite/shape/video4.avi) the plane is divided into multiple segments with two alternating textures. In every other segment there is large bias, and this gives rise to the perception of the plane folding as if it were a staircase.

Before going on, let us note that the underestimation of slant is not due to the particular constraints we employed. We may instead compute structure from normal flow (the component of optical flow perpendicular to edges) by fitting a plane to the flow data. The surface normal vector obtained this way (from the structure estimates) will have a qualitatively similar behavior.



(a) (b)

Figure 2.12: (a) The plane in view. (b) Scene geometry in the shape from motion demonstration.

2.5 Summary of the parametric influences on the bias

Table 1 summarizes the findings of the last two sections. It displays the effects of the viewing geometry parameters and the texture on the bias for general 3D motion between the cameras and for stereo in particular. T and R denote the translation and rotation, and $|T|$ and $|R|$ their absolute value. ℓ_1 and ℓ_2 are are two orthogonal image lines expressed in projective coordinates. Notice that any texture may be represented by two orthogonal directions corresponding to the eigenvectors of the matrix M'.

	large bias	small bias
Two view geometry (motion and stereo)	small $\|T\|$ large $\|R\|$ large slant small $(T \cdot \ell_1)(T \cdot \ell_2)$	large $\|T\|$ small $\|R\|$ small slant large $(T \cdot \ell_1)(T \cdot \ell_2)$
Stereo	horizontal and vertical texture lines surface slanted about vertical axis	texture lines at 45° surface slanted about horizontal axis

Table 2.1: Influence of viewing geometry and texture on the size of the bias.

2.6 Discussion

2.6.1 Why is estimation so difficult

The statistical model used to describe the data in our equation $Ax = b$ is the errors-in-variable model, which is defined as:

Definition 1 *(Errors-In-Variable Model)*

$$\begin{aligned} b' &= A'x + \epsilon \\ b &= b' + \delta_b \\ A &= A' + \delta_A \end{aligned}$$

x' are the true but unknown parameters. $A = (A_{i,j})$ and $b = (b_i)$ are observations of

the true but unknown values A' and b'. $\delta_{A_{i,j}}, \delta b_j$ are the measurement errors and ϵ is the modeling error. This is the error due to our model assumptions.

It is well known that for this model Least Squares (LS) estimation is biased. The main reason is that it does not consider errors in the explanatory variables, that is the δ_A. Let us then investigate the theoretical question. Are there better ways to estimate? Are there better statistical estimators that do not suffer from bias? The answer is that in general we cannot avoid bias. We may be able to reduce the bias, if there is enough data available to obtain reasonable error statistics. In this case the bias will still be of the same form, only smaller. Simply weighting the data to make it uniform is statistically not justified as it would increase the variance in the direction of lesser data significantly. Choosing a nonlinear estimator (such as Total Least Squares), which would give a different form of bias, does not appear to give better results because of the noise to be expected. A short discussion summarizing the main arguments is given next.

The so-called **Corrected Least Squares (CLS)** estimator is the classical technique to address bias. If the statistics of the noise, that is the covariance matrix of δ_A, is known, an asymptotically unbiased linear estimator could be constructed. The problem is that for small amounts of data, accurate estimation of the variance of the noise is a very difficult problem, and has high variance itself, and this leads to higher variance for the CLS estimation. It is well known that the scale of the error variance is difficult to obtain in practice.

In the computational vision literature more attention has been given to the non-linear technique of **Total Least Square (TLS)**, which deals with the errors in A and b symmetrically and only requires the ratio of the error variances. If all the errors $\delta_{A_{i,j}}$ and δb_j are identical and independent, or their ratio can be obtained, then TLS estimation is asymptotically unbiased. Estimation of the ratio of errors is not easy either. However, the main problem for TLS is modeling error (or also called system error [42]). Theoretically one can use multiple tests to obtain the measurement errors, like re-measuring or re-sampling; but unless the exact parameters of the model are known, one cannot test for the modeling error. The noise we expect is actually much more complicated than simple i.i.d. additive noise. It is correlated, and this would cause further problems for TLS, causing convergence problems for the corresponding nonlinear non-convex objective function to be minimized [50]. TLS is attractive in the sense that it has an obvious geometrical explanation, but it does not appear advantageous for the vision applications discussed. Its improvement over usual least squares in the statistical sense would be offset with more complicated error models or if mis-modeling the error.

We can classify the errors into two categories: *measurement noise* and *modeling error*. In the problem at hand the measurements are the line parameters $\{a_i, b_i\}$, and the image motion parameters of the lines $\{\dot{a}_i, \dot{b}_i\}$. We can expect *measurement errors* due to *sensor noise* which effects the measurements of image intensity $I(x, y, t)$. It seems reasonable to approximate the sensor noise as i.i.d. But we have to consider dependencies when the images are smoothed. Other errors in measurement are due to *bad fitting*, when estimating the line parameters with edge detectors and

discretization due to the edge detectors and difference operators computing the derivatives.

Modeling errors are due to erroneous assumptions. When computing the motion of lines, we assume that the image intensity is constant between frames. Significant errors occur at specular components. We use first order expansions when deriving velocities. Thus, errors are expected for large local velocities. Furthermore, the modeling of the scene as consisting of planar patches is an approximation to the actual surface of the scene.

Sensor noise may be considered i.i.d. and is easier to deal with ([51, 52]). But other errors could be more significant, and they are more elaborate, making the statistics rather complicated. It is too difficult to estimate the statistics of the combined noise, which is necessary to apply the classical techniques.

There is another technique widely known in Economics which theoretically may be well suited for vision, the **technique of instrumental variables (IV technique)**, which deals with the errors in the explanatory variables but does not require the error variance as a priori. This techniques uses additional variables, the instrumental variables, which could be additional measurements of the explanatory variables. Let these variables be called W. If the errors in the measurements of the two methods can be treated as independent, an asymptotically unbiased estimator [42] can be created, whose variance is close to the variance of the CLS estimator, by solving the replaced equation system

$$(W^T A)x = (W^T)b, \tag{2.24}$$

with standard least square estimation. But even, if the errors are not fully independent, but not completely related, the technique can help reduce the bias.

Possible ways to obtain instrumental variables are by taking multiple measurements of the explanatory variables, for example by using multiple edge detections, fitting schemes, or difference operators.

Using color image sequences we could create even better instrumental variables. We may use one color channel as instrumental variables to another color channel. It is quite reasonable to assume that the sensor noise components are independent in the different color channels. The approximation errors in the image gradients would not be completely independent since there is a similarity in the structure of the color intensity functions. This means, we could not completely remove the bias from approximation error, but we could partially correct the bias caused by this error. We cannot correct the bias from the modeling error. But this is the advantage of this technique despite the presence of modeling error, it still can deal with the other errors.

2.6.2 Is our vision system doing the best?

Bias is only one component of estimation, the other is variance; and there is a trade-off between the two. Generally an estimator correcting for bias increases the variance while decreasing the bias. In statistics, the performance of an estimator is evaluated by a risk function. Usually the mean squared error (MSE) is used as performance criterion. It is the expected value of the square of the difference

between the estimated and the true value. If x' is used to denote the actual value, $\hat{x}$ to denote the estimate, and $E(\cdot)$ to denote the expected values, the MSE is defined as

$$\begin{aligned} MSE(\hat{x}) &= E((\hat{x} - x')^2) \\ &= (E(\hat{x}) - x')^2 + E(x' - E(\hat{x}))^2 \\ &= bias^2(\hat{x}) + cov(\hat{x}), \end{aligned} \tag{2.25}$$

that is, as the sum of the square of the bias (denoted as $bias(\hat{x})$) and the variance (denoted as $cov(\hat{x})$).

Let us assume we know the variance of the error as well as the covariance of the LS estimate exactly. In appendix 2.7.5 we derive an expression for the best linear estimator. This would be a partial correction using CLS. How much to correct depends on the covariance of the LS estimate. The larger the covariance, the less the correction.

Thus, theoretically the best we could do is to partially correct. If there is a sufficient amount of data we should be able to somewhat correct the bias. A good choice for doing so, would be a conservative, that is slight correction using CLS or the IV method. Such a correction would lead to an estimation with bias qualitatively of the same form as LS but smaller in value.

What about the human vision system? We assume that it is doing the best it can. If it has enough data, it should be able to perform some correction. Two observations make us believe that it does.

Partially corrected estimation would explain why the illusory perception in

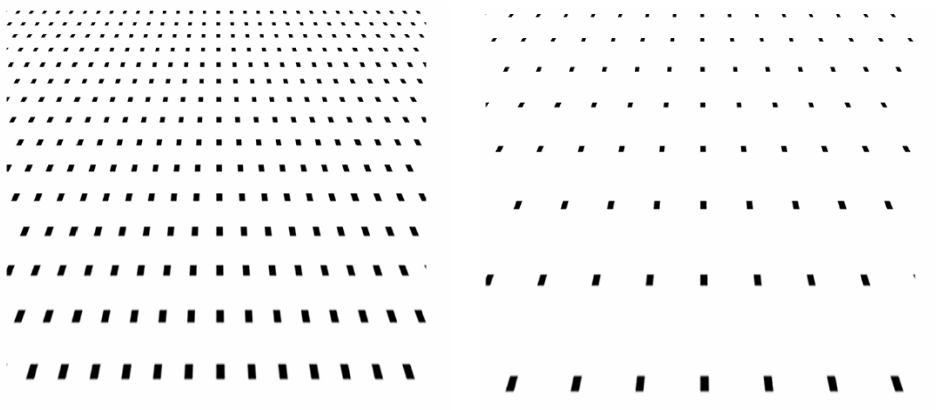


Figure 2.13: Influence of texture density on estimation. The plane is slanted with an angle of 45°. The denser pattern appears to be estimated closer to the veridical than the sparser pattern.

many optical illusions weakens after extended viewing, in particular when subjects are asked to fixate ([53]). In these cases, we can assume that the noise parameters stay fixed, and the visual system can reasonably well estimate them.

We can draw conclusions by varying the covariance (of the estimator) in a pattern. Two patterns created by line segments of same orientation but different density give rise to the same bias, but different covariance. The smaller the covariance of the estimator, the more the correction and thus the less the bias should be. Thus, a pattern with higher density and smaller covariance should be estimated better. We tested different textures and found our perception to be consistent with the hypothesis. An example is shown in Figure 2.13. Patterns of larger density and thus smaller covariance of the estimator appear to result in better estimation.

2.6.3 Structure and motion in the feed-back loop

We have analyzed the effects of noise on the estimation of shape and found that there is bias. We showed that this bias predicts the underestimation of slant, which is known from computational and psychophysical experiments. Our analysis was based on LS estimation, but we showed that other estimators suffer from bias too, and for the most appropriate estimators the bias is of the same form as for LS estimation. The main reason for the inability to correct for the bias lies in the difficulty to obtain good estimates of the statistics of the data.

Vision scientists have long realized that a large part of the visual processes are carried out by feeding information from higher processing areas to lower processing areas. The Computer Vision literature creating algorithms, however, has not embraced this view yet. That is nobody will argue that recognition requires top-down process, but the problems of reconstruction discussed here have been studied in a pure feed-forward fashion. This brings us to the question: Is there a need for computational feed-back? It is clear that the problems of model parameter estimation and segmentation are computationally antagonistic to each other. The idea was that it is possible set up a large minimization which includes discontinuity localization and parameter estimation that will solve the problems. However the approach has not been proven to be successful. It appears that the information one can extract from the signal without having knowledge of the scene model is not sufficient to perform good reconstruction.

Better estimation techniques are not the answer to bias. Thus we have to use

the data such that bias does (mostly) not effect the goal, that is, what we want to do with the data. First, we should use the data selectively. Since we understand how the the different parameters influence the bias, we can choose data that is not effected much by the bias. For example in computing shape from motion we can avoid patches with textures corresponding to a badly conditioned matrix M. Second, we should use as discussed above, the data globally. Large amounts of data usually are not directionally biased, and thus the bias in estimation will be small. Third, the statistics becomes easier if the data available is less correlated. For example, structure (or shape) from motion is easier for the discrete case than the continuous case, since in the discrete case the errors in the image measurements are expected to be less correlated. Thus, it is advantageous to estimate shape from views far apart. Of course, using far away views we run into the difficulty of finding good correspondence. The way to address structure from motion then is to use continuous motion to obtain a preliminary estimate of the 3D motion and shape, and subsequently use these estimates to obtain shape from views far apart.

2.7 Appendix

2.7.1 Shape from lines in multiple views: The constraint

Consider the general stereo configuration of two cameras displaced by a rigid motion with translation T and rotation R. Let the scene be a textured plane with surface normal N. The texture is described by the lines on the plane. A line L in 3D space is a four-dimensional object and can be elegantly described by Plücker

coordinates. Let P_1 and P_2 be two points with unit distance and P any point on $L = (L_d, L_m)$. Then

$$\begin{cases} L_d = P_1 - P_2; \\ L_m = P \times L_d = P_2 \times P_1. \end{cases} \tag{2.26}$$

L_d denotes the direction of the line in space, and L_m its moment. Geometrically L_m is a vector perpendicular to the plane through L and the coordinate center O with value the distance of L from O (Figure 2.3). L_d and L_m are perpendicular, that is $L_d \cdot L_m = 0$. The projection ℓ of the 3D line L on the image is just L_m normalized, i.e to have the third coordinate 1, it is $\ell = \frac{L_m}{\hat{z} \cdot L_m}$, where $\hat{z}$ is a unit vector parallel to the z-axis.

Since points in the two views are related as $\tilde{P} = RP + T$, the line parameters in the two views are related as

$$\begin{cases} \widetilde{L_d} = \tilde{P}_1 - \tilde{P}_2 = R(P_1 - P_2) = RL_d; \\ \widetilde{L_m} = \tilde{P}_2 \times \tilde{P}_1 = (RP_2 + T) \times (RP_1 + T) = RL_m + T \times RL_d. \end{cases} \tag{2.27}$$

Thus, the orientation of the line can be obtained as

$$\begin{aligned} \ell \times (R^T \tilde{\ell}) &= \frac{L_m}{\hat{z} \cdot L_m} \times \frac{L_m + R^T T \times L_d}{\hat{z} \cdot \widetilde{L_m}} \\ &= \frac{-(L_m \cdot R^T T) L_d}{(\hat{z} \cdot L_m)(\hat{z} \cdot \widetilde{L_m})} = \frac{-(\ell \cdot R^T T)}{\hat{z} \cdot \widetilde{L_m}} L_d. \end{aligned} \tag{2.28}$$

Since L_d is perpendicular to the surface normal N we have that

$$(\ell \times R^T \tilde{\ell}) \cdot N = 0. \tag{2.29}$$

In the case of differential motion, where the motion of a point in space has velocity Hence

$$\dot{\ell} = \frac{\dot{L_m}}{(\hat{z} \cdot L_m)} - \frac{(\dot{L_m} \cdot \hat{z})}{(\hat{z} \cdot L_m)} \frac{L_m}{(\hat{z} \cdot L_m)} = \frac{1}{(\hat{z} \cdot L_m)} t \times L_d + \omega \times \ell - \frac{(\dot{L_m} \cdot \hat{z})}{(\hat{z} \cdot L_m)} \ell, \tag{2.30}$$

and the constraint in (2.2.1) takes the form

$$\ell \times (\dot{\ell} - \omega \times \ell) = -\frac{t \cdot \ell}{\hat{z} \cdot L_m} L_d. \tag{2.31}$$

Thus if the 3D line is on the plane with normal vector N, its image ℓ must obey

$$N \cdot (\ell \times (\dot{\ell} - \omega \times \ell)) = 0. \tag{2.32}$$

To clarify the use of these equations; slant is estimated using equations (2.29) and (2.32). The relations (2.28) and (2.31) are used to analyze the bias.

2.7.2 Expected value of Least Squares solution

Consider the equation system

$$(e_{1_i}, e_{2_i}) \cdot (N_1, N_2) = -e_{3_i}, \tag{2.33}$$

with corrupted measurements

$$e_{k_i} = e'_{k_i} + \delta e_{k_i}, \quad k = 1, 2, 3.$$

The Least Square solution amounts to $x = (E^T E)^{-1} E^T G$, where E and G are the $n \times 2$ and $n \times 1$ matrices $E = E' + \delta_E = (e_{1_i}, e_{2_i})_n$, $G = G' + \delta_G = (-e_{3_i})_n$. Assuming that the errors are much smaller than the real values, we develop the LS solution of x in a second order Taylor expansion. Since the noise terms are considered i.i.d. with mean 0, we obtain as an approximation for the expected value $E(\cdot)$:

$$E(x) \approx x' + \sum_i \sum_{\delta t_i \in \delta_V} \frac{\partial^2 E(x)}{\partial \delta {t_i}^2}\bigg|_{\delta t_i = 0} \frac{E(\delta {t_i}^2)}{2}, \tag{2.34}$$

where in the case of motion δ_V is the set of all variables $\{\delta_{a'_i}, \delta_{b'_i}, \delta_{\dot{a}'_i}, \delta_{\dot{b}'_i}\}$, and in the case of stereo δ_V is the set $\{\delta_{a'_i}, \delta_{b'_i}, \delta_{\tilde{a}'_i}, \delta_{\tilde{b}'_i}\}$. Let M' denote $E'^T E'$. Using the fact that for any arbitrary matrix Q

$$\frac{-\partial Q^{-1}}{\partial x} = Q^{-1}\frac{\partial Q}{\partial x}Q^{-1} \tag{2.35}$$

the expected value of x is approximated as

$$E(x) = x' - \sum_i \sum_{t_i \in V} \frac{\delta t_i^2}{2}\Big(M'^{-1}\begin{pmatrix} \frac{\partial^2 e'^2_{1_i}}{\partial t_i^2} & \frac{\partial^2 e'_{1_i} e'_{2_i}}{\partial t_i^2} \\ \frac{\partial^2 e'_{1_i} e'_{2_i}}{\partial t_i^2} & \frac{\partial^2 e'^2_{2_i}}{\partial t_i^2} \end{pmatrix} x' + M'^{-1}\begin{pmatrix} \frac{\partial^2 e'_{1_i} e'_{3_i}}{\partial t_i^2} \\ \frac{\partial^2 e'_{2_i} e'_{3_i}}{\partial t_i^2} \end{pmatrix}\Big), \tag{2.36}$$

with V the set $\{a'_i, b'_i, \dot{a}'_i, \dot{b}'_i\}$, or $\{a'_i, b'_i, \tilde{a}'_i, \tilde{b}'_i\}$.

2.7.2.1 Motion analysis

Let the variances of δa_i and δb_i be δ_1^2, and the variance of $\delta\dot{a}_i$ and $\delta\dot{b}_i$ be δ_2^2. We then substitute for e_i in (2.36) from (2.20) and write out the derivatives piecemeal:

$$\begin{aligned}
\frac{\partial^2 e'^2_{1_i}}{\partial a_i'^2} + \frac{\partial^2 e'^2_{1_i}}{\partial b_i'^2} &= 2(b'_i\omega_2 + \omega_3)^2 + 2(a'_i\omega_2 - 2b'_i\omega_1)^2 \\
&\quad +4\omega_1(\dot{b}'_i + (1+b'_i)^2\omega_1 - a'_i b'_i\omega_2 - a'_i\omega_3) \\
\frac{\partial^2 e'_{1_i} e'_{2_i}}{\partial a_i'^2} + \frac{\partial^2 e'_{1_i} e'_{2_i}}{\partial b_i'^2} &= 2(b'_i\omega_2 + \omega_3)(b'_i\omega_1 - 2a'_i\omega_2) \\
&\quad +2\omega_2(\dot{b}'_i + (1+b_i'^2)\omega_1 - a'_i b'_i\omega_2 - a'_i\omega_3) \\
&\quad -2\omega_1(\dot{a}'_i + a'_i b'_i\omega_1 - (1+a_i'^2)\omega_2 + b'_i\omega_3) \\
&\quad +2(a'_i\omega_2 - 2b'_i\omega_1)(a'_i\omega_1 + \omega_3) \\
\frac{\partial^2 e'^2_{2_i}}{\partial a_i'^2} + \frac{\partial^2 e'^2_{2_i}}{\partial b_i'^2} &= 2(b'_i\omega_1 - 2a'_i\omega_2)^2 \\
&\quad -4(\dot{a}'_i + a'_i b'_i\omega_1 - (1+a_i'^2)\omega_2 + b'_i\omega_3)\omega_2 \\
&\quad +2(a'_i\omega_1 + \omega_3)^2
\end{aligned}$$

$$
\begin{aligned}
\frac{\partial^2 e'_{1_i} e'_{3_i}}{\partial a_i'^2} + \frac{\partial^2 e'_{1_i} e'_{3_i}}{\partial b_i'^2} &= 2(b'_i\omega_2 + \omega_3)(\dot{b}'_i + \omega_1 - 2a'_i\omega_3) \\
&\quad +4\omega_3(\dot{b}'_i + (1 + b_i'^2)\omega_1 - a'_i b'_i\omega_2 - a'_i\omega_3) \\
&\quad -2\omega_1(\dot{b}'_i a'_i - \dot{a}'_i b'_i + a'_i\omega_1 + b'_i\omega_2 - (a_i'^2 + b_i'^2)\omega_3) \\
&\quad +2(-2b'_i\omega_1 + a'_i\omega_2)(-\dot{a}'_i + \omega_2 - 2b'_i\omega_3) \\
\frac{\partial^2 e'_{2_i} e'_{3_i}}{\partial a_i'^2} + \frac{\partial^2 e'_{2_i} e'_{3_i}}{\partial b_i'^2} &= 2\omega_2(\dot{a}'_i b'_i - \dot{b}'_i a'_i - a'_i\omega_1 - b'_i\omega_2 + (a_i'^2 + b_i'^2)\omega_3) \\
&\quad +2(b'_i\omega_1 - 2a'_i\omega_2)(\dot{b}'_i + \omega_1 - 2a'_i\omega_3) \\
&\quad -4\omega_3(\dot{a}'_i + a'_i b'_i\omega_1 - (1 + a_i'^2)\omega_2 + b'_i\omega_3) \\
&\quad +2(a'_i\omega_1 + \omega_3)(-\dot{a}'_i + \omega_2 - 2b'_i\omega_3)
\end{aligned}
$$

$$
\begin{array}{llll}
\frac{\partial^2 e_{1_i}'^2}{\partial \dot{a}_i'^2} + \frac{\partial^2 e_{1_i}'^2}{\partial \dot{b}_i'^2} = 2 & & \frac{\partial^2 e_{2_i}'^2}{\partial \dot{a}_i'^2} + \frac{\partial^2 e_{2_i}'^2}{\partial \dot{b}_i'^2} = 2 & \\
\frac{\partial^2 e'_{1_i} e'_{3_i}}{\partial \dot{a}_i'^2} + \frac{\partial^2 e'_{1_i} e'_{3_i}}{\partial \dot{b}_i'^2} = -2a'_i & & \frac{\partial^2 e'_{2_i} e'_{3_i}}{\partial \dot{a}_i'^2} + \frac{\partial^2 e'_{2_i} e'_{3_i}}{\partial \dot{b}_i'^2} = -2b'_i &
\end{array} \tag{2.37}
$$

and all other second order derivatives are 0. For the simplicity of expression, we consider a_i and b_i to be independent random variables which are symmetric with respect to the center of the image coordinate system; in other words, $E(a_i^k) = E(b_i^k) = 0, k = 1, 3$. The $\dot{a}_i$ and $\dot{b}_i$ are very small and set to 0. Then with enough equations, the expected value for the LS solution of x is well approximated by

$$
E(x) = x' - M'^{-1}(\delta_2^2 D' + \delta_1^2 F')x' - M'^{-1}\delta_1^2 H', \tag{2.38}
$$

where

$$
D' = \begin{pmatrix} n & 0 \\ 0 & n \end{pmatrix}, \quad H' = \omega_3 \sum_i^n \begin{pmatrix} \omega_1(6b_i'^2 + c'_i + 3) \\ \omega_2(6a_i'^2 + c'_i + 3) \end{pmatrix}, \tag{2.39}
$$

$$
F' = \sum_i^n \begin{pmatrix} 6b_i'^2\omega_1^2 + c'_i\omega_2^2 + \omega_3^2 + 2\omega_1^2 & 2c'_i\omega_1\omega_2 + 2\omega_1\omega_2 \\ 2c'_i\omega_1\omega_2 + 2\omega_1\omega_2 & c'_i\omega_1^2 + 6a_i'^2\omega_2^2 + \omega_3^2 + 2\omega_2^2 \end{pmatrix} \tag{2.40}
$$

with $c'_i = a_i'^2 + b_i'^2$.

2.7.2.2 Stereo

Substituting in (2.36) from (2.8) and setting the variance $E(\delta a_i^2) = E(\delta b_i^2) = E(\delta \tilde{a}_i^2) = E(\delta \tilde{b}_i^2) = \delta^2$ we obtain for the derivatives:

$$\begin{array}{lllllllll} \sum_{\delta t_i} \frac{\delta^2 e'_{1_i} e'_{2_i}}{\delta t_i^2} & = & 0, & \sum_{\delta t_i} \frac{\delta^2 e'_{1_i} e'_{3_i}}{\delta t_i^2} & = & -2(a+\tilde{a}), & \sum_{\delta t_i} \frac{\delta^2 e'_{2_i} e'_{3_i}}{\delta t_i^2} & = & -2(b+\tilde{b}), \\ \sum_{\delta t_i} \frac{\delta^2 e'^2_{1_i}}{\delta t_i^2} & = & 4, & \sum_{\delta t_i} \frac{\delta^2 e'^2_{2_i}}{\delta t_i^2} & = & 4. & & & \end{array} \tag{2.41}$$

To simplify, we align the image center such that $E(a_i + \tilde{a}_i) = 0$ and $E(b_i + \tilde{b}_i) = 0$, and we obtain for the expected value of x

$$E(x) = x' - 2n\delta^2 M'^{-1} x'. \tag{2.42}$$

2.7.3 Stereo: slant about the vertical and horizontal

Let us use $N_v = (x_v, 1) = (\tan\sigma, 0, 1)$ and $N_h = (x_h, 1) = (0, \tan\sigma, 1)$ to denote the surface normals in the planes slanted about the vertical and the horizontal axes respectively. Denoting the real slant as σ we derive from (2.8) the expected value of $x = (N_1, N_2)$ as:

$$\begin{array}{rcl} E(x) & = & \begin{pmatrix} N'_1 \\ N'_2 \end{pmatrix} - \delta^2 \begin{pmatrix} \frac{1}{n}\sum e'^2_{1_i} & \frac{1}{n}\sum e'_{1_i} e'_{2_i} \\ \frac{1}{n}\sum e'_{1_i} e'_{2_i} & \frac{1}{n}\sum e'^2_{2_i} \end{pmatrix}^{-1} \begin{pmatrix} N'_1 \\ N'_2 \end{pmatrix} \\ & = & \begin{pmatrix} N'_1 - n\delta^2 |E(M')|^{-1} (E(\sum e'^2_2) N'_1 - E(\sum e'_1 e'_2) N'_2) \\ N'_2 - n\delta^2 |E(M')|^{-1} (E(\sum e'^2_1) N'_2 - E(\sum e'_1 e'_2) N'_1) \end{pmatrix}. \end{array} \tag{2.43}$$

Then for the two settings above, omitting terms of $O(\delta^4)$, we have

$$E(||x_v||^2) = (1 - \epsilon_v)||x'||^2 = (1 - (2n\delta^2 \tfrac{E(\sum e'^2_{2_v})}{|E(M'_v)|}))||x'||^2, \tag{2.44}$$

$$E(||x_h||^2) = (1-\epsilon_h)||x'||^2 = (1-(2n\delta^2\frac{E(\sum e'^2_{1_h})}{|E(M'_h)|}))||x'||^2. \tag{2.45}$$

Consider the following Taylor expansion:

$$\begin{aligned} E(\sigma) &= \cos^{-1}(((1-\epsilon)||x'||^2+1)^{-\frac{1}{2}}) \\ &= \cos^{-1}(((1-\epsilon)\tan^2\sigma+1)^{-\frac{1}{2}}) \\ &= \sigma - (\frac{1}{4}\sin 2\sigma)\epsilon + O(\epsilon^2), \end{aligned} \tag{2.46}$$

which is a reasonable approximation only for smaller angles σ, because for larger angles ϵ becomes large. We then have

$$E(\sigma_v) = \sigma - n\delta^2\frac{\sin 2\sigma}{2}\frac{E(\sum e'^2_{2_v})}{|E(M'_v)|} = \sigma - n\delta^2\frac{\sin 2\sigma}{2}C_v, \tag{2.47}$$

$$E(\sigma_h) = \sigma - n\delta^2\frac{\sin 2\sigma}{2}\frac{E(\sum e'^2_{1_h})}{|E(M'_h)|} = \sigma - n\delta^2\frac{\sin 2\sigma}{2}C_h. \tag{2.48}$$

Denote the lines in the front-parallel view by

$$\begin{aligned} \hat{L}_{m_i} &= (-\sin\theta, \cos\theta, k_i), \hat{L}_{d_i} = (\cos\theta, \sin\theta, 0), \\ \hat{L}_{m_j} &= (-\cos\theta, -\sin\theta, k_i), \hat{L}_{d_j} = (-\sin\theta, \cos\theta, 0). \end{aligned} \tag{2.49}$$

Then through rotation we obtain the parameters in the vertically and horizontally tilted plane as

$$\begin{aligned} L_{m_v} &= R_y(\sigma)\hat{L}_m, \quad L_{d_v} = R_y(\sigma)L_{d_h}, \\ L_{m_h} &= R_x(\sigma)\hat{L}_m, \quad \hat{L}_d = R_x(\sigma)L_{d_h}, \end{aligned} \tag{2.50}$$

where

$$R_y(\sigma) = \begin{pmatrix} \cos\sigma & 0 & -\sin\sigma \\ 0 & 1 & 0 \\ \sin\sigma & 0 & \cos\sigma \end{pmatrix} \quad \text{and} \quad R_x(\sigma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\sigma & -\sin\sigma \\ 0 & \sin\sigma & \cos\sigma \end{pmatrix}.$$

We have

$$e \approx (T \cdot L_m)L_d.$$

Let $T = (t, 0, 0)$. Then under the assumption that the images have small fields of view, that is the magnitude of the k_is is small, we have that

$$\frac{1}{n}E(M'_v) \sim t^2\cos^2\sigma \begin{pmatrix} 2\cos^2\sigma\sin^2\theta\cos^2\theta & \cos\sigma\sin\theta\cos\theta(\sin^2\theta - \cos^2\theta) \\ \cos\sigma\sin\theta\cos\theta(\sin^2\theta - \cos^2\theta) & \sin^4\theta + \cos^4\theta \end{pmatrix},$$

$$\frac{1}{n}E(M'_h) \sim t^2 \begin{pmatrix} 2\sin^2\theta\cos^2\theta & \cos\sigma\sin\theta\cos\theta(\sin^2\theta - \cos^2\theta) \\ \cos\sigma\sin\theta\cos\theta(\sin^2\theta - \cos^2\theta) & \cos^2\sigma(\sin^4\theta + \cos^4\theta) \end{pmatrix}.$$

Therefore

$$C_v = n\frac{E(\sum e'^2_{2_v})}{|E(M'_v)|} = \frac{1}{t^2}\frac{(\sin^4\theta + \cos^4\theta)}{\cos^4\sigma(\sin^2\theta\cos^2\theta)}, \tag{2.51}$$

$$C_h = n\frac{E(\sum e'^2_{1_h})}{|E(M'_h)|} = \frac{2}{t^2\cos^2\sigma}. \tag{2.52}$$

and

$$\frac{C_v}{C_h} = \frac{\frac{E(\sum e'^2_{2_v})}{|E(M'_v)|}}{\frac{E(\sum e'^2_{1_h})}{|E(M'_h)|}} = \frac{1}{\cos^2\sigma}\frac{\sin^4\theta + \cos^4\theta}{2\sin^2\theta\cos^2\theta} = \frac{1}{\cos^2\sigma}\frac{\tan^2\theta + \cot^2\theta}{2} > 1. \tag{2.53}$$

2.7.4 Matrix M for motion

Consider a slanted plane with a texture of two major directional components. Let the directional components be $L_{d_1} = (\cos\tau_1\sin\sigma_1, \sin\tau_1\sin\sigma_1, \cos\sigma_1)$ and $L_{d_2} = (\cos\tau_2\sin\sigma_2, \sin\tau_2\sin\sigma_2, \cos\sigma_2)$. From (2.31) we have that

$$e = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} \sim (t \cdot \ell)L_d.$$

Thus

$$
\begin{aligned}
M = E^T E &= \begin{pmatrix} \sum e_{1_i}^2 & \sum e_{1_i} e_{2_i} \\ \sum e_{1_i} e_{2_i} & \sum e_{2_i}^2 \end{pmatrix} \\
&= \sum (t \cdot \ell_{1_i})^2 \sin^2 \sigma_1 \begin{pmatrix} \cos^2 \tau_1 & \sin \tau_1 \cos \tau_1 \\ \sin \tau_1 \cos \tau_1 & \sin^2 \tau_1 \end{pmatrix} \\
&+ \sum (t \cdot \ell_{2_i})^2 \sin^2 \sigma_2 \begin{pmatrix} \cos^2 \tau_2 & \sin \tau_2 \cos \tau_2 \\ \sin \tau_2 \cos \tau_2 & \sin^2 \tau_2 \end{pmatrix} \qquad (2.54)
\end{aligned}
$$

and the determinant $det(M)$ of M amounts to

$$
det(M) = \sum (t \cdot \ell_{1_i})^2 \sum (t \cdot \ell_{2_i})^2 (\sin \sigma_1 \sin \sigma_2 \sin(\tau_1 - \tau_2))^2. \qquad (2.55)
$$

2.7.5 Bias correction

Consider the equation system $Ax = b$. Let the errors be described by the errors in variable model. The bias of the LS estimator $\boldsymbol{x}_{LS}$ amounts to

$$
bias(\boldsymbol{x}_{LS}) = \lim_{n \to \infty} E(\boldsymbol{x}_{LS} - x') = -\sigma^2 (\lim_{n \to \infty} (\frac{1}{n} A'^T A'))^{-1} x'. \qquad (2.56)
$$

Assuming the variance of the error δA is known, the bias can be removed with the CLS estimator, which amounts to

$$
\boldsymbol{x}_{CLS} = (A^T A - n\sigma^2 I)^{-1} (A^T b), \qquad (2.57)
$$

and can be rewritten as

$$
\boldsymbol{x}_{CLS} = (I - n\sigma^2 (A^T A)^{-1})^{-1} \boldsymbol{x}_{LS}. \qquad (2.58)
$$

The variance, denoted as as $cov(\cdot)$, of $\boldsymbol{x}_{CLS}$ amounts to

$$
cov(\boldsymbol{x}_{CLS}) = (I - n\sigma^2 (A^T A)^{-1})^{-2} cov(\boldsymbol{x}_{LS}). \qquad (2.59)
$$

Let us now investigate what the theoretically best linear estimator should be. We have to adjust the corrected least squares estimator, such that we achieve the smallest MSE (as defined in (2.25)). Let $x_\alpha = \alpha \boldsymbol{x}_{CLS} + (1-\alpha)\boldsymbol{x}_{LS}$ denote the adjusted CLS estimator.

Then we have that

$$\begin{aligned} MSE(x_\alpha) &= \alpha^2 cov(\boldsymbol{x}_{CLS}) + (1-\alpha)^2(bias^2(\boldsymbol{x}_{LS}) + cov(\boldsymbol{x}_{LS})) \\ &= \alpha^2(1-n\sigma^2(A^TA)^{-1})^{-2}cov(\boldsymbol{x}_{LS}) + (1-\alpha)^2(bias^2(\boldsymbol{x}_{LS}) + cov(\boldsymbol{x}_{LS})). \end{aligned}$$

. (2.60)

which is a quadratic expression in α. Thus, the MSE minimum is achieved for

$$\begin{aligned} \alpha &= \frac{bias^2(\boldsymbol{x}_{LS}) + cov(\boldsymbol{x}_{LS})}{bias^2(\boldsymbol{x}_{LS}) + cov(\boldsymbol{x}_{LS}) + cov(\boldsymbol{x}_{LS})(1-n\sigma^2(A^TA)^{-1})^{-2}} \\ &= 1 - \frac{cov(\boldsymbol{x}_{LS})(1-n\sigma^2(A^TA)^{-1})^{-2}}{bias^2(\boldsymbol{x}_{LS}) + cov(\boldsymbol{x}_{LS}) + cov(\boldsymbol{x}_{LS})(1-n\sigma^2(A^TA))^{-2}} \\ &= 1 - \frac{(1-n\sigma^2(A^TA)^{-1})^{-2}}{(1+(1-n\sigma^2(A^TA)^{-1})^{-2}) + \frac{bias^2(\boldsymbol{x}_{LS})}{cov(\boldsymbol{x}_{LS})}}. \end{aligned} \quad (2.61)$$

This shows that according to the MSE criterion a less bias corrected x_α is better than a bias corrected $\boldsymbol{x}_{CLS}$. The larger the variance of the LS estimates, the less the correction should be.

Chapter 3

A shape constraint for linking frames

3.1 Overview

Structure from motion from single flow fields has been extensively studied and over the years many good techniques have been developed. However, the information in one motion field is not rich enough to allow for accurate estimation of 3D motion and structure. There are two issues. First, there is the ambiguity in the estimation of the motion parameters. For standard cameras with limited field of view, there is a confusion between translation and rotation [19, 20, 14, 21]. Thus any motion algorithm, because of noise, can estimate the motion only within a range of the true solution. The second issue is the stability of structure estimation. An erroneous estimate of the motion parameters clearly will lead to errors in the estimation of structure. Furthermore, even for the correct motion parameters, the estimation of structure, because of the small displacement between the cameras (small baseline), is very unreliable. Usually a global description of the scene, that is the relative depth estimates of different scene patches, can be obtained rather well. However, a local description of structure, that is shape estimates of scene patches, is very unstable.

To obtain good motion and structure estimates we need to combine the information of consecutive motion fields. One flow field, or in the abstraction two image

frames, are constrained only by the rigidity of 3D motion. The rigidity can be expressed with two constraints on the image measurements; the epipolar constraint, which says that individual flow vectors lie on a line, and the depth positivity constraint, which says that the reconstructed scene points have to be in front of the camera. Two or more motion fields are constrained in addition by the observed scene which remains constant. Existing approaches formulate this as a constraint enforcing the estimated structure, or depth of the scene, to be the same. In order to make use of this constraint, image points (or lines) over multiple frames need to be corresponded. However, automatic correspondence usually is not possible. Drifting occurs, that is errors in correspondence accumulate till eventually the correspondence cannot be established anymore. Another problem is, that since structure in the coordinate system of one frame is related to structure in the coordinate system of another frame through both the translation and rotation, the resulting constraints are not simple and don't allow for robust estimation of structure and motion. We have the trilinear constraint resulting in 27 and the quadrilinear constraint resulting in 64 parameters whose estimations are very sensitive. [10, 54, 55, 56].

In this chapter we propose to combine multiple motion fields, not through depth or inverse depth values, but through 3D shape. The 3D shape of a scene patch is described by the surface normal of the patch, that is by two parameters. Consider the scene as consisting of planar patches. The surface normal of a planar patch in one view is related to the surface normal of the corresponding patch in another view only through rotation. That is, let r_1 and r_2 be the surface normals of a patch in the first and second frame, and let Ω be the rotation relating the two

frames, then $r_2 = \Omega r_1$. This relationship can be formulated as a rank 3 constraint on a matrix containing the normal vectors of all the patches over all views. The advantage of this constraint is two-fold. First, we do not need to correspond image points over multiple frames, but we need to correspond image patches only, which is a much easier task. Second, the constraint can be combined easily with the estimation of motion and structure from individual flow fields. This way we can estimate structure and motion from multiple flow fields as a practical constrained minimization, where the constraint is the rank constraint on the surface normals.

The new constraint is implemented in an algorithm, which involves the following computations: We first segment planar patches in the scene and match the patches over the image sequence. The segmentation (Section 3.5) is based on color and motion information. Then using as input the image gradients, the 3D motion parameters and the shape of the extracted scene patches are estimated (Section 3.3). Starting from motion estimates from single flow fields we solve the constrained minimization iteratively in a two-step optimization; in one step the surface normal are obtained, in the next step the motion parameters. Section 3.6 presents experiments, and Section 3.7 discusses the role of the approach in a complete structure from motion framework.

The multiple view constraints defined on point correspondences are well understood [10, 54, 55, 56]. Nowadays most point correspondence methods employ the technique of bundle adjustment [57] to refine 3D structure and viewing parameters. Oliensis et al. [58, 59] proposed algorithms, which first eliminate the rotational components and then decompose the residual correspondences into structure and

translation. A number of studies considered the estimation from multiple flow fields assuming continuity in the motion [60, 61, 62, 63].

[64] and [65] developed algorithms using line and point correspondences for the reconstruction of scenes with planar objects. The first ones to present a subspace constraint on homographies of planes in multiple views were Shashua and Avidan [66]. In the sequel [67] presented a subspace constraint on the relative homographies of pairs of planes across the different views. Most closely related to our work is the study of Zelnik-Manor and Irani [2], which introduced a subspace constraint on image motion. The approach assumes that differential motion between a reference frame and any other frame at the same scene points can be obtained. Clearly, this assumption limits its application to small image sequences. The improvement from our methods stems from the use of 3D shape, which makes it computationally feasible to use longer sequences.

3.2 Preliminaries

In this section, we summarize the basic concepts in structure from motion, and explain how the relative motion of the camera and the scene manifests itself in the sequence of images which the observer obtains.

3.2.1 Optical flow and motion field

In a pinhole camera model, the camera is defined as a point, the center of projection, and a planar retina (image plane). The projection of a scene point can

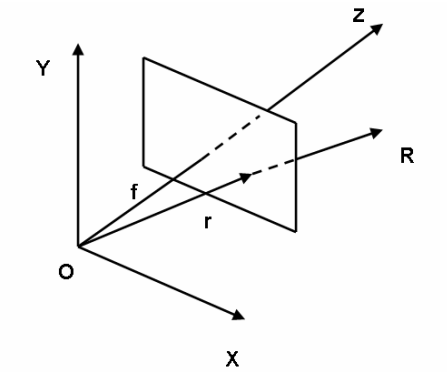


Figure 3.1: Image formation of a pinhole camera.

be found as the intersection of the image plane and the straight ray connecting the point with the projection center (See Fig. 3.1).

Let $R = (X, Y, Z)^t$ denote a 3D scene point and $r = (x, y, f)$ denote its corresponding point in the image plane $Z = f$, then we have

$$r = f\frac{R}{R \cdot \hat{z}} \tag{3.1}$$

where $\hat{z} = (0, 0, 1)^t$. Without loss of generality, we assume $f = 1$.

As the camera moves, scene features change position with respect to the camera and their image projections also move within the image. If the temporal interval between successive frames is small, the image sequence can be considered as a representation of a continuous image brightness function. So the image intensity at image position $r = (x, y)$ at time t can be written as $I(r, t)$. The relative motion between the scene and camera induces a 3D velocity field $\frac{dR}{dt}$ which denotes an instantaneous velocity relative to the camera. The projection of this 3D motion field onto the image plane is then called the *motion field* $\frac{dr}{dt}$. The *optical flow* field is an approximation of the motion field computed from the image sequence.

We make the standard brightness constancy assumption, assuming constant brightness of the projection of a scene point in the different images. Based on this assumption, differentiating $I(r,t)$ with respect to t yields:

$$\frac{dI}{dr}\cdot\frac{dr}{dt}+\frac{dI}{dt}=\frac{\partial I}{\partial x}\cdot\frac{dx}{dt}+\frac{\partial I}{\partial y}\cdot\frac{dy}{dt}+\frac{\partial I}{\partial t}=0, \tag{3.2}$$

This equation relates the image velocity $\frac{dr}{dt}$ with the spatio-temporal derivatives of the image intensity function. Thus, the brightness constancy assumption provides the component of image velocity parallel to the image brightness gradient direction. We call this component *normal flow*. It is easy to see that the velocity component perpendicular to the gradient direction cannot be directly recovered from the first order derivatives. This is the so-called *aperture problem*.

If the scene in view is static, the relative motion of the camera and of the scene is rigid, and is composed of a translation and a rotation. Suppose the observer is moving rigidly with instantaneous translation $t=(t_x,t_y,t_z)$ and instantaneous rotation $\omega=(\omega_x,\omega_y,\omega_z)$ around the nodal point of the camera. Then each scene point $R=(X,Y,Z)^t$ moves relative to the camera with velocity $\frac{dR}{dt}$:

$$\frac{dR}{dt}=-t-\omega\times R.$$

The velocity of its projected image point r is then obtained by differentiating (3.1) as follows:

$$\frac{dr}{dt}=\frac{\frac{dR}{dt}}{R\cdot\hat{z}}-\frac{(\frac{dR}{dt}\cdot\hat{z})R}{(R\cdot\hat{z})^2}=\frac{\hat{z}\times(\frac{dR}{dt}\times R)}{(R\cdot\hat{z})^2}.$$

or equivalently, it could be expressed as:

$$\frac{dr}{dt}=\begin{pmatrix}\frac{dx}{dt}\\ \\ \frac{dy}{dt}\end{pmatrix}=\frac{1}{Z}\begin{pmatrix}-t_x+t_zx\\ \\ -t_y+t_zy\end{pmatrix}+\begin{pmatrix}xy\omega_x-(x^2+1)\omega_y+y\omega_z\\ \\ -xy\omega_y+(y^2+1)\omega_x-x\omega_z\end{pmatrix}, \tag{3.3}$$

where $t = (t_x, t_y, t_z)$ and $\omega = (\omega_x, \omega_y, \omega_z)$ are the translation and rotation parameters respectively.

Thus (3.3) tells us how the depth Z and the camera motion $\{t, \omega\}$ are related to the motion field $\frac{dr}{dt}$. Recall that (3.2) provides us partial information on the relation of the motion field $\frac{dr}{dt}$ to the image gradients in the spatial and temporal domain. Thus the combination of (3.3) and (3.2) gives us an approach for computing the structure of the scene (depth) and camera movement based on image spatial-temporal gradients.

3.2.2 Motion and shape estimation from individual flow fields

Since we can obtain only normal flow fields from image gradients, we only have one linear constraint at each image point. Totally, we will have N (the number of valid image points) linear constraints. However, there are also N unknown depth variables Z and six motion parameters $\{t, \omega\}$. Thus structure from motion generally is not solvable without some assumption on the scene depth. Here we model the scene as a piece-wise planar surface. Such a model actually covers a wide range of indoor scenes.

Consider R on the world plane Γ which is determined by its plane parameter vector $n = (\alpha, \beta, \gamma)^t$ as follows:

$$n \cdot R = \alpha X + \beta Y + \gamma Z = 1.$$

The plane parameter vector $n = (\alpha, \beta, \gamma)$, describes the distance of the plane to

original point and its surface normal. Thus, the inverse depth at P amounts

$$\frac{1}{Z} = n \cdot \frac{R}{Z} = \alpha x + \beta y + \gamma.$$

Substituting the expressions for the image motion and the plane into the brightness constancy constraint (3.2), we obtain the constraint

$$[-I_x t_x - I_y t_y + (I_x x + I_y y) t_z][x\alpha + y\beta + \gamma]$$

$$+(I_x xy + I_y(y^2+1))\omega_x - (I_y xy + I_x(x^2+1))\omega_y + (I_x y - I_y x)\omega_z = -I_t, \quad (3.4)$$

which relates the image gradients to the motion and plane parameters. This equation is bilinear in the motion parameters and the plane parameters. That is, knowing t we can solve linearly for ω and n, and vice versa. Or, similarly knowing n we can solve linearly for t and ω. Because of the scaling ambiguity between translation and depth, we can only estimate the direction of vectors t and n.

Let $p_i = (x_i, y_i), i = 1, \cdots, N$ denote the image points on a single world plane. From (3.4) we obtain an equation system for this plane, which we write as:

$$f(t, \omega, n) = 0 =$$

$$[t_x A_1 + t_y A_2 + t_z A_3] \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} + B \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} - b, \quad (3.5)$$

where A_1, A_2, A_3, B are $N \times 3$ matrices and b is an $N \times 1$ vector, whose elements are described by the image point coordinates and the intensity gradients at points p_i. That is,

$$A_1 \;=\; -I_{x_i}(x_i, y_i, 1),$$

$$
\begin{aligned}
A_2 &= -I_{y_i}(x_i, y_i, 1), \\
A_3 &= (I_{x_i} x_i + I_{y_i} y_i)(x_i, y_i, 1), \\
b &= (I_{t_i}), \\
B &= (I_{x_i} xy + I_{y_i}(y_i^2 + 1) - (I_{y_i} x_i y_i + I_{x_i}(x_i^2 + 1)) + (I_{x_i} y_i - I_{y_i} x_i)).
\end{aligned}
$$

Our method starts with 3D motion estimates from individual flow fields. In principal, many of the algorithm from the literature could be employed. We used the algorithm in [68], which is based on the equations above.

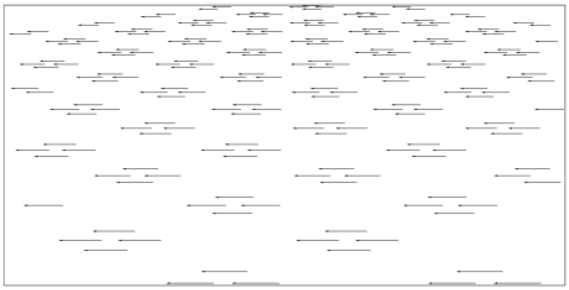
Consider a segmentation of the scene into P planar patches $V^1, V^2 \cdots V^P$. Combining equations (3.5) for all the patches we obtain an over-determined bilinear equation system of the form

$$
f^p(t, \omega, n^p) = 0 \quad \text{for} \quad p = 1, \cdots, P, \tag{3.6}
$$

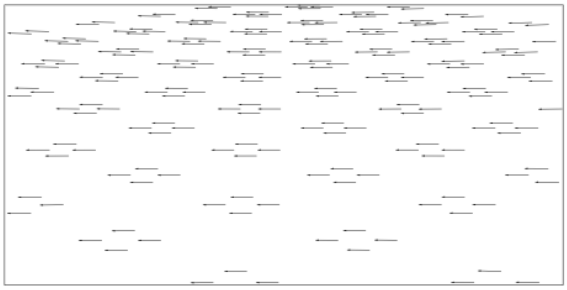
whose solution provides estimates for the direction of translation, the rotation and the structure parameters for the individual patches. The algorithm in [68] solves this minimization as a search in the space of translational directions. For each candidate translation one can solve closed-form for the rotation and the plane parameters. The translational direction minimizing eq. (3.6) in the least squares sense provides the solution. Note the algorithm does not require optical flow, but only the image gradients, which define the so-called normal flow.

3.2.3 Ambiguities in 3D motion estimation from a single flow field

Several studies have addressed the noise sensitivity in structure from motion. In particular it has been shown that for standard cameras with small field of view



(a) Pure translational motion field induced by t_x



(b) Pure rotational motion field induced by ω_y.

Figure 3.2: Within a small field of the view, translational and rotational motion fields of scenes with small depth variation are similar.

imaging a shallow scene, translation and rotation are easily confused (See Fig. 3.2 in [19]). This can be understood by examining the differential flow equation (3.3). Notice that for a shallow scene with $Z(x, y)$ varying very little, a zeroth order approximation of the flow amounts to $\frac{dx}{dt} \approx \frac{-t_x}{Z} - \omega_y$ and $\frac{dy}{dt} \approx \frac{t_y}{Z} + \omega_x$. Intuitively, we can see how t_x translation along the x axis can be confused with ω_y rotation around the y axis, and t_y with ω_x for a small field of view. Thus, in the presence of noise it is hard to distinguish these motions. The most likely estimation error is such that the projection of the translational error and the rotational error on the image are perpendicular to each other, and the estimated translation direction lies along a line passing through the true translation direction and the viewing direction

[19, 20, 14, 21]. [20] shows that if no prior information about the motion is available, the error of camera motion estimation satisfies the follow equations:

$$\gamma_\epsilon = 0,$$

$$\frac{\alpha_\epsilon}{\beta_\epsilon} = -\frac{x_{0_\epsilon}}{y_{0_\epsilon}},$$

$$\frac{x_0}{y_0} = \frac{x_{0_\epsilon}}{y_{0_\epsilon}},$$

where unprimed letters with subscript ϵ denote the estimation error and (x_0, y_0) is so-called *FOE* defined as

$$(x_0, y_0) = \frac{1}{t_z}(t_x, t_y).$$

Since the estimated motion field always is noisy, we can obtain only a range of possible solutions for the motion parameters, among which the correct one lies. If we consider the 2D subspace of translational directions of this range and visualize it on the image (or on a sphere), we usually obtain an elongated region, which we refer to as the *motion valley* of solutions. Each translation direction in the motion valley, along with its best corresponding rotation and structure will agree with the observed noisy flow field. Figure 3.3 from [69] shows error functions (residual of the minimization) plotted on the 2D spherical surface. The best solutions lie in the bright area of the surface. The error function makes it evident that attempting to pick a single solution in this valley is futile. Such valleys are ubiquitous, and if we pick an erroneous motion estimate, this results in the estimation of distorted structure [70].

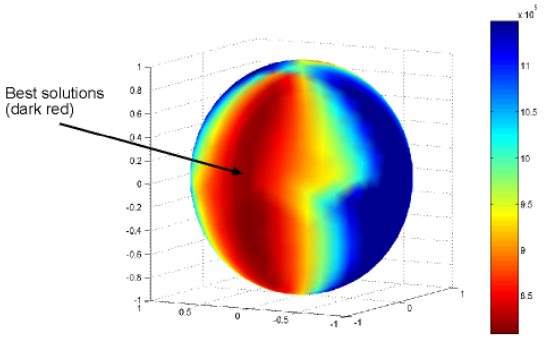


Figure 3.3: The motion valley is the area of smallest values on the error surface in the 2D space of translational directions. The error is found by computing for each translation the optimal rotation.

3.3 Rank constraint on 3D shape parameters

Let the image frames be denoted as $I_1, I_2, \cdots, I_F$ and the scene patches as V_f^p, where subscript f indicates the frame index and superscript p indicates the patch index. The translational and rotational velocities of the normal flow field at frame I_f are t_f and ω_f. One thing to be emphasized here is that the frames we use do not need to be consecutive. Actually, there is no need to combine all consecutive frames; one may combine frames far apart. The reason is that because of temporal smoothing and because of the small baseline, views close by do not provide very different information. However, note that the motion parameters t_f and ω_f are the differential velocities computed from the flow field at frame f between consecutive frames, and not the motions between the chosen frames. The normal flow field at every frame I_f provides an equation system $\{f(t_f, \omega_f, n_f^p)\}$ (3.6) for the estimation of motion and depth. In order to combine the systems of the different I_f we need to

formulate some constraint which models the fact that all the frames view the same static scene.

Many researchers have been working on some form of constraints to integrate multiple frames. [2] provides a nice rank constraint to enforce the fact that all the frames view the same static scene. However such a rank constraint is derived under the assumption that sequence is a short video, in other words, the motion between all the pair-wise frames could be well approximated by the differential motion. Such an assumption holds true for some applications such as short airborne video sequences. However, the benefit of this constraint aimed mostly toward improving the optical flow field. It couldn't overcome the motion ambiguity, because only frames with short baseline are integrated.

[67] attempted to integrate multiple frames with large baseline with another rank constraint. The authors derived a rank constraint on the so-called *relative homography* between planes, which essentially is a constraint on the relative parameters of all the planes in the scene to a large background plane. The effectiveness of this constraint heavily depends on the accuracy of the estimation of the background plane. In practice, an accurate estimation of the background plane is not always feasible. Furthermore, we have no reliable way to measure whether the estimation of the background plane is accurate or not. The residual from the flow field estimation is not exactly equivalent to the error in 3D information estimation.

Hence, our goal is to develop a constraint which is able to integrate multiple frames with large baseline, but also is practically feasible for general settings. The basic idea is to enforce the fact that the surface normals of scene patches to be the

same. The surface normals in the different coordinate systems of the frames are easily related. The relative orientation of the different patches is invariant across views. The absolute orientations of a patch in the different views are related by rotation only. This invariance can be expressed as a rank constraint on a matrix containing the surface normals.

Let n_f^p be the parameter vector of the inverse depth which describes the plane with index p in frame I_f. The normal vector r_f^p of this plane is obtained by normalizing the vector n_f^p, i.e

$$r_f^p = \frac{1}{||n_f^p||_2} n_f^p.$$

Let R_f denote the matrix which collects normal vectors of all planes in frame I_f as follows:

$$R_f = (r_f^1, r_f^2, \cdots, r_f^P).$$

Furthermore, we combine R_f of all F frames into a a matrix $3F \times P$ matrix M as follows:

$$M = \begin{pmatrix} R_1 \\ R_2 \\ \vdots \\ R_F \end{pmatrix}. \tag{3.7}$$

Then we get a rank constraint on M.

Theorem 3.1. *The rank of the matrix M defined in (3.7) is 3.*

Proof. The normal vectors of a plane in two frames I_{f_1}, I_{f_2} are related by the

rotation matrix Ω_{f_1,f_2} as

$$r^p_{f_1} = \Omega_{f_1,f_2} r^p_{f_2} \quad \text{for} \quad p = 1, \cdots, P,$$

Thus R_{f_1} amounts to

$$\begin{aligned} R_{f_1} &= (r^1_{f_1}, r^2_{f_1}, \cdots, r^P_{f_1}) \\ &= (\Omega_{f_1,f_2} r^1_{f_2}, \Omega_{f_1,f_2} r^2_{f_2}, \cdots, \Omega_{f_1,f_2} r^P_{f_2}) \\ &= \Omega_{f_1,f_2}(r^1_{f_2}, r^2_{f_2}, \cdots, r^P_{f_2}) = \Omega_{f_1,f_2} R_{f_2}. \end{aligned}$$

Then the $3F \times P$ matrix M is

$$M = \begin{pmatrix} R_1 \\ R_2 \\ \vdots \\ R_F \end{pmatrix} = \begin{pmatrix} I_3 R_1 \\ \Omega_{2,1} R_1 \\ \vdots \\ \Omega_{F,1} R_1 \end{pmatrix} = \begin{pmatrix} I \\ \Omega_{2,1} \\ \vdots \\ \Omega_{F,1} \end{pmatrix} R_1. \tag{3.8}$$

Notice that R_1 is a $3 \times P$ matrix; the rank of M is thus 3. □

A simple, straightforward way to utilize this constraint would be to perform ego-motion estimation independently on each frame I_f and subsequently perform a subspace projection on the estimated r^p_fs to regularize the estimates. Since the individually estimated r^p_fs are already very erroneous, such an approach will not lead to significant improvement. Instead, we incorporate the rank 3 constraint directly into the estimation process.

3.4 Motion and shape estimation from multiple flow fields

The exposition in this chapter is as follows. We first reformulate the estimation of motion and structure on the basis of individual flow fields only. Then we embed

the 3D shape estimation into this formulation (3.4.1) and (3.4.2).

For every frame I_f there is a bilinear system $\{f_f^p(t_f, \omega_f, n_f^p) = 0\}$ which we rewrite as:

$$A_f^p(t_f) n_f^p = d_f^p(\omega_f), \tag{3.9}$$

with

$$A_f^p(t_f) = t_{x_f} A_{1_f}^p + t_{y_f} A_{2_f}^p + t_{z_f} A_{3_f}^p$$

and

$$d_f^p(\omega_f) = -B_f^p \omega_f + b_f^p.$$

The estimation of motion and structure from the individual flow fields is formulated as a least squares optimization, that is

$$\min_{\omega_f, t_f, n_f^p} \sum_p ||A_f^p(t_f) n_f^p - d_f^p(\omega_f)||^2 = \min_{\omega_f, t_f} \min_{n_f^p} \sum_p ||A_f^p(t_f) n_f^p - d_f^p(\omega_f)||^2. \tag{3.10}$$

We can address this minimization in two iterative steps as follows: Given initial values for t_f and ω_f,

Step 1: Solve for structure: Substitute the values t_f, ω_f into the system and solve the over-determined linear system for all n_f^p using linear least squares estimation.

Step 2: Solve for motion: Substitute the values n_f^p into the system, and solve for t_f and ω_f using least squares estimation.

Go back to Step 1 until the estimation converges.

3.4.1 Adding the shape constraint to the estimation

The motion constraint above is defined on the vectors n_f^p, but our shape constraint is defined on the normalized surface normal vectors r_f^p. Thus, we first need to transform the non-homogeneous system $A_f^p n_f^p = d_f^p$ for the unknown vector n_f^p into a homogeneous system $W_f^p r_f^p = 0$ with $||r_f^p|| = 1$. This is shown next.

Let H_f^p be the matrix, such that the vector d_f^p spans the null space of H_f^p, i.e. $H_f^p d_f^p = 0$. H_f^p amounts to

$$H_f^p = I - \frac{d_f^p d_f^{p\,t}}{d_f^{p\,t} d_f^p}.$$

Then multiplying H_f^p on both sides of the equation $A_f^p n_f^p = d_f^p$ yields

$$(H_f^p A_f^p) n_f^p = H_f^p d_f^p = 0.$$

Let $W_f^p = H_f^p A_f^p$ and $r_f^p = \frac{n_f^p}{\|n_f^p\|}$ (the normalized version of n_f^p). We then obtain the following homogeneous constraint on the normal vectors r_f^p:

$$W_f^p r_f^p = 0 \quad \text{for} \quad p = 1, \cdots, P \qquad f = 1, \cdots, F.$$

Incorporating the shape constraint, the minimization becomes:

$$\min_{t_f, \omega_f} \min_{\|r_f^p\|=1} \sum_{p,f} ||W_f^p(t_f, \omega_f) r_f^p||^2, \quad \text{subject to} \quad rank(M(r_f^p)) = 3. \tag{3.11}$$

We adopt the two-step optimization for the estimation of motion and structure from multiple frames presented in [71]. Step 2, that is, the estimation of t_f and ω_f remains the same; given $n_f^p = \|n_f^p\| r_f^p$ we minimize (3.10) using least squares. However, Step 1, the estimation of n_f^p is rather different and more difficult. We solve it by first estimating r_f^p and then estimating $\|n_f^p\|$. See Fig. 3.4 for the outline of the algorithm, the details are described in the next subsection.

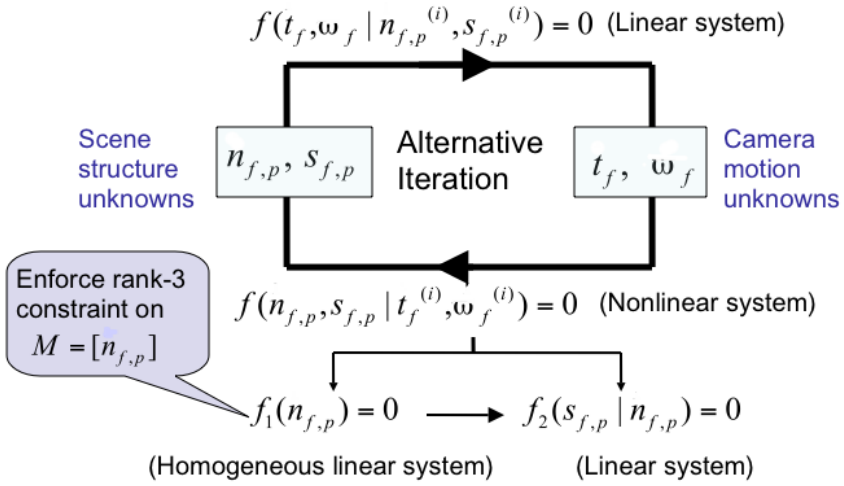


Figure 3.4: The sequence of optimizations.

3.4.2 Estimating the shape parameters

We are given estimates of t_f and ω_f and wish to estimate the r_f^p. We can pick frames which we want to combine, for example the first, tenth and twentieth frame of the sequence. In the general case, we will not concatenate the estimates ω_f to obtain a rotation between the selected frames, but re-estimate the rotation matrices.

Recall from Section 3.3 that $R_f = \Omega_{f,1} R_1$. Therefore

$$R_f^t R_f - R_1^t R_1 = R_1^t (\Omega_{f,1}^t \Omega_{f,1} - I) R_1 = 0.$$

In words, $R_f^t R_f$, which is the matrix encoding the relative orientations between the different patches, does not depend on the frame number. Using this we can rewrite the minimization (3.11) as:

$$\min \sum_{f,p} ||W_f^p r_f^p||^2, \quad \text{subject to} \quad ||r_f^p|| = 1, R_f^t R_f = R_1^t R_1. \tag{3.12}$$

This is a well defined least squares minimization with quadratic constraints. There are many standard algorithms dealing with such types of minimization. We used the Mukai-Polak version of the Augmented Lagrangian method (ALS) ([72]) which guarantees super-linear convergence.

After having obtained the r_f^p from (3.12), we need to estimate $\|n_f^p\|$, the length of the n_f^p, in order to solve subsequently for translation and rotation. This is done using the first equation in $A_f^p n_f^p = d_f^p$.

3.4.3 Consecutive frames

When combining a few consecutive frames (as opposed to significantly separated frames) $\Omega_{f,1}$ is well approximated by concatenating the differential motions ω_f. That is, $\Omega_{f,1} = (I + [\omega_f]_\times)\Omega_{f-1,1}$. For this case, the problem is much simpler. Since $\Omega_{f,1}$ is known, the minimization (3.11) becomes finding the least squares solution of an homogeneous system. That is,

$$\min_{||r_1^p||=1, r_f^p=\Omega_{f,1}r_1^p} \sum_{p,f} ||W_f^p r_f^p||^2 = \min_{||r_1^p||=1} \sum_p (\sum_f ||W_f^p \Omega_{f,1} r_1^p||^2), \tag{3.13}$$

which is solved using SVD decomposition. Although the combination of a few consecutive frames couldn't help much in resolving motion ambiguity, it could be utilized for accurate optical flow estimation for planar surfaces. We will adopt this technique for computing accurate optical flow used in the application of super-resolution imaging (Chapter 5).

3.5 Patch segmentation and matching

For our purpose the image segmentation does not need to give an "object-related" division of the scene. It only needs to locate some planar patches which are suitable for ego-motion estimation and which can be matched over the image sequence. Therefore the segmentation could be a little bit " too fine" in the sense that different patches could correspond to the same planar surface in space. However, the segmentation should not be be "too coarse" in the sense that individual image patches should not correspond to heavily curved surfaces or multiple planar surfaces.

3.5.1 Basic idea and algorithm outline

We perform segmentation and matching using a graph-based approach, consisting of the following components:

1. An edge enforced color segmentation in the individual frames, which provides an over-segmentation.

2. Matching of the color patches in the different frames of the sequence. The geometric transformation of the patches between the frames is described by the motion model $T(x, y)$ (involving 2D translation, rotation and scaling) as

$$T(x,y) = s \begin{pmatrix} \cos\theta & 0 \\ 0 & \sin\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}.$$

The matching is carried out by phase correlation.

3. Taking a bottom-up approach to segmentation, neighboring color patches are merged using a graph-based algorithm. The criterion for merging is based on a combined rank constraint on the flow fields. It is evaluated how well all the normal flow fields of a patch, which has been matched over the image sequence, model a plane.

Next we describe in more detail the individual steps of the algorithm and explain why it should result in a good patch segmentation and good matching. Segmentation based on color usually works best for weakly textured regions, while segmentation based on flow information works best for well textured regions. Thus these two cues complement each other. The procedure starts with an edge-enforced color segmentation which leads to an over-segmentation, and most likely many of the components are too small for ego-motion and shape estimation. However, color segmentation usually does not break smooth regions in space. There exist many color segmentation techniques which can achieve such a segmentation.

Next the patches need to be merged on the basis of flow information. This is done not on the frames individually, but by linking the patches between the frames first, and then merging them on the basis of the flow information in all frames. We model the movement of each color patch by the transform $T(x, y)$, and find the best matching color patch in every frame. This transform proved to be sufficient for the tracking of color patches. Then we perform merging on the linked patches based on some function, which measures how well the flow fields of the merged patch over all frames fit the planar motion model. Such measurement function V is defined as

follows. Let $V = \{V_f, f = 1, \cdots, F\}$ be the patches matched from frame I_1 to I_F. The measurement function $R(V)$ is defined as

$$R(V) = \sum_f \frac{1}{k_f} ||(S(V_f))||_2,$$

where k_f is a value denoting how good the matching of the patch in frame f is, In our implementation, k_f is the least square residual of the affine transform estimation between the matched patches. $S(V_f)$ is the value denoting how well the flow for patch V_f fit the planar motion model, which is also the least square residual from the fitting of a planar motion model to the flow.

It is important that flow information from multiple frames is used in the segmentation. A patch may be tracked well across multiple frames, but be non-planar. In this case the multiple rank constraint will most probably find the non-planarity. Alternatively, a patch may be tracked badly across the frames. In this case either there is an occlusion, or the movement of the patch is too complicated to be modeled by the transform T. In the latter case the patch most likely cannot be approximated well by a planar surface. In both cases, the combined rank constraint will not allow merging. This is exactly what we want, leaving out the patches close to occlusions or corresponding to non-planar regions. A shortage of such an approach is that it could leave only a small fraction of patches in the image. However, for the purpose of ego-motion estimation, this does not matter.

3.5.2 Detailed algorithm

We take a graph-based approach implementing bottom-up merging. Our algorithm is derived from [73], where it is used for color segmentation. The algorithm is closely related to Kruskal's minimum spanning tree for graphs [74]. The initial graph is constructed as follows: each color patch set V^k (the patches matched across different frames) denotes a vertex in the graph, and each neighboring pair of such vertices (U, V) denotes an edge e in the graph. The weight of an edge is defined as $R(U \cup V)$. Then given a graph $G = (V, E)$, the procedure is as follows:

1. Sort E into $\pi = (e_1, e_2, \cdots, e_m)$ according to no-decreasing edge weight.

2. Start with a segmentation S^0, where each vertex is a component.

3. Repeat step 4 until all edges in π are checked out.

4. Construct S^k from S^{k-1} as follows: Let V_i, V_j be two vertices connected by an edge e_k. If V_i and V_j in S^{k-1} are different components and the weight of the corresponding edge is not greater than the internal difference of the components, merge the two vertices, otherwise do nothing.

5. Final segmentation $S = S^m$.

For our method, we define the internal difference of two component U, V as

$$M_{int}(U, V) = \kappa \frac{R(U \cup V)}{size(U \cup V)},$$

where $size(W)$ is the size of the patch W and κ is the threshold which determines the average patch size. From the definition of the function M_{int} it is clear that when

Figure 3.5: Key frame for the sequence "Office".

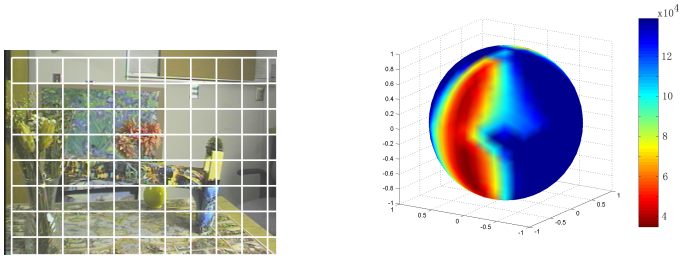
the patch size grows, the internal difference becomes smaller. Hence the resulting segmentation regions are neither too large (the gain in numerical stability is little and the risk for bad patch increases), nor too small (which causes numerical instability in later computations).

3.6 Experiments

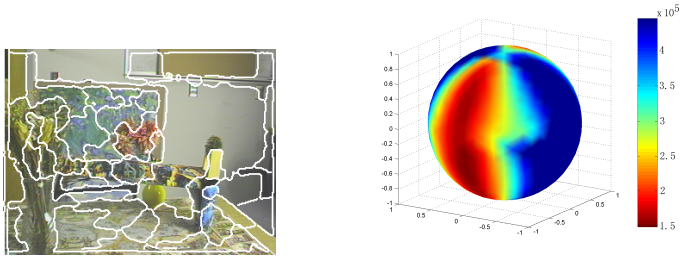
3.6.1 3D motion estimation

The color image sequence, "office" (see Fig. 3.5) was taken by a hand-held camera. The motion is a translation mostly along the x- and y-axes and a rotation. For this motion, because of the camera's small field of view, the ambiguity between translation and rotation makes the motion estimation very difficult. Fig. 3.6 compares the effects of different segmentations on the motion estimation from single flow fields. Although the residual value decreases (by a factor of 2.5) with better segmentation, the valleys are qualitatively very similar, demonstrating that the ambiguity in motion estimation cannot be avoided.

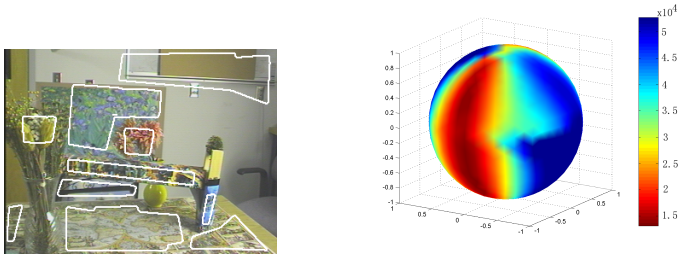
Next we integrated multiple flow fields using the algorithm described in the



(a)Brute-force segmentation

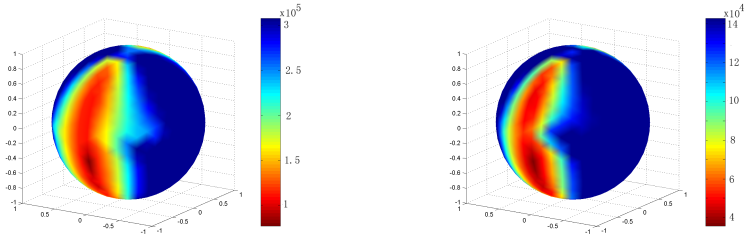


(b) Automatic segmentation



(c) Manual segmentation

Figure 3.6: Residual spheres from single frame ego-motion estimation in the sequence "office".



(a) Automatic segmentation (b) Manual segmentation

Figure 3.7: Residual for multi-frame motion estimation in the sequence "office".

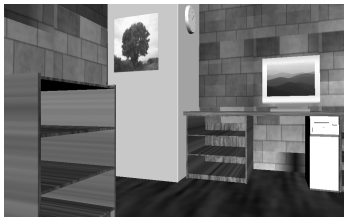


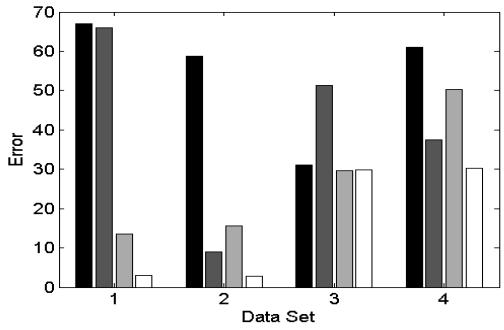
Figure 3.8: Key frame of the synthesized sequence used for comparison.

previous sections. We used three frames separated by a significant baseline. Fig. 3.7 demonstrates significant reduction in the ambiguity. It also shows that our segmentation performs similar to the manual one. The table below compares the absolute smallest values for the translational directions $(x_0, y_0) = (\frac{t_x}{t_z}, \frac{t_y}{t_z})$.

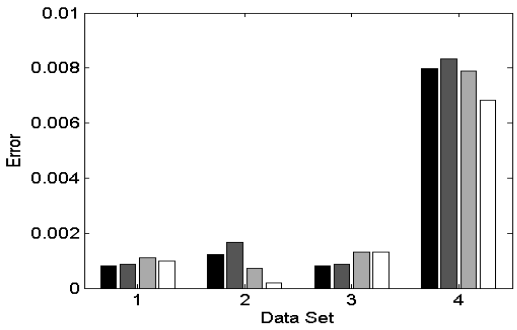
	Single frame	Multiple frame
Automatic Partition	(3.14, 0.02)	(2.15, -0.02)
Manual Partition	(2.30, 0.01)	(2.07, -0.01)

We compared our algorithm to the method of Zelnik-Manor and Irani [2] using the synthetic scene in 3.8. This method only estimates 8-parameters of the projective flow model for each patch. We added another step to obtain the 3D motion from these parameters. The results of the comparison for four different motions are presented in Figure 3.9. The error in translation is measured by the angular difference between the estimated translation direction and the true direction. The error in rotation is measured by the L_2 norm between estimated and true values. Referring to Figure 3.9,

It can be seen that our algorithm performs significantly better on data set 1 and 2 and a bit better on data set 3 and 4. The reason for the decreased performance on data set 3 is the large noise in the image gradients due to the large image dis-



(a) Translation



(b) Rotation

Figure 3.9: Comparison of errors in motion estimation for different types of camera motion between [2]'s and our algorithm. The bars from black to white denote in turn: The algorithm in [2] for single image motion fields, the algorithm in [2] for multiple frames, our ego-motion estimation for single flow fields, our approach for multiple frames. The motions in the four data sets are as follows. Data set 1: translation in the $x - z$ plane and small rotation. Data set 2: translation along the $y-$ axis and small rotation. Data set 3: dominating translation along the $z-$axis and small rotation. Data set 4: mostly rotation and small translation.

placement caused by zooming. This could be remedied by introducing a hierarchical framework. All algorithms perform poorly on data set 4. The reason is the large displacement. Since there is only small translation, and thus the flow carries very little information on structure, the combination of multiple frames cannot improve the estimation significantly.

3.7 The structure from motion feedback loop

We want to convey with this chapter that multiple image motion fields can be combined through a constraint on 3D shape. We have implemented this constraint in a technique, and demonstrated that it provides very good 3D motion estimation. However, there may be better ways of utilizing this constraint. We consider our estimation as one module in a structure from motion framework.

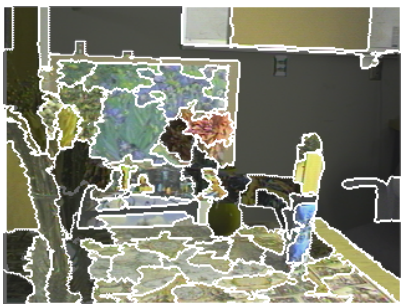
We cannot obtain good models using only local measurements (image motion or correspondence) in a bottom up approach. Local image measurements do not allow for good structure estimation and localization of the discontinuities. 3D motion is not effected very much by noise, because it is globally encoded in the image, but structure is spatially local. To obtain good structure we need processes that involve larger spatial areas. But to employ such processes we need models of the scene. In other words, there need to be feed-back loops.

Our algorithm provides us with 3D motion estimates over multiple frames. In the sequel we can obtain depth from image motion and we can fit shape models to the segmented patches. Using this information we can then go back to better

segment and estimate structure using images significantly separated by baseline.

Figure 3.10 shows first experimental results. Using the 3D motion estimate we rectified two significantly separated frames and computed the depth map using stereo. Then we inserted the boundaries obtained from stereo into the segmentation of our algorithm and merged areas for which the flow gave continuous depth values (Fig. 3.10a). This gives a segmentation based on motion, stereo and color. Fig. 3.11 shows the the depth map. On the basis of the 3D motion estimate, the parametric motion (corresponding to planes) was fit to the image intensity derivatives within the segmented areas. As can be seen, this computation, although based on flow and not disparity between far away views, leads to a very good reconstruction.

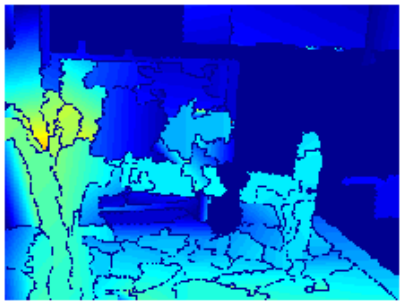
(a)Segmentation from color information



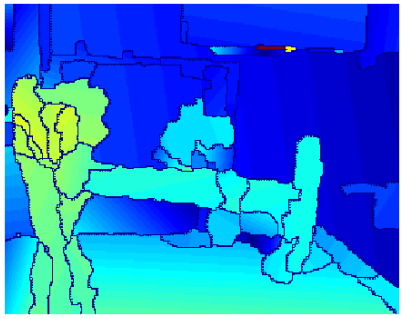
(b) Segmentation from affine flow information

(c) Segmentation from stereo and depth

Figure 3.10: Segmentation from motion and stereo and corresponding depth estimation from normal flow.



(a) Depth map from segmentation (b) in Fig. 3.10



(b) Depth map from segmentation (c) in Fig. 3.10

Figure 3.11: Depth estimations from various segmentations.

Chapter 4

A Projective invariant for texture

4.1 Overview

A key component in the description of images is texture. The ideal descriptor of texture for visual homing should be able to capture essential perceptual properties of the texture structure, and it should be invariant to environmental changes, such as changes in view-point, illumination and geometry of the underlying surface. In the Computer Vision literature, the search for invariance started in the nineties [75], when researcher dug up the mathematical literature on algebraic and geometric invariants. Great importance has been given to the study of quantities which are invariant to the viewpoint from which the image is taken. A number of projective invariants [76] have been found, which are defined on feature sets of points and lines and planar curves [75], and they have been used for object recognition and calibration. However, none of these descriptors provides a high-level characterization of textures.

Numerous texture descriptors have been proposed. Most of them are either statistics-based or filter-based ([77, 78, 79, 80]), which makes them sensitive to changes in viewpoint. Lazebnik, Schmid and Ponce ([3]) proposed a texture representation which is invariant to view-point changes in a weak form (it is *locally* invariant to *affine* transforms).

Here we introduce a novel texture descriptor called the MFS (multifractal spectrum vector), which is based on fractal geometry theory. The MFS is globally invariant under the *bi-Lipschitz transform*, a very general transform which includes perspective transforms (viewpoint changes) and smooth texture surface deformations. Furthermore, the MFS has extraordinary low dimension (26 using one feature and three times that much using three features as in our implementation here for a typical 640×480 image texture). We demonstrate with a number of experiments on synthesized and real image textures that, first the MFS in practice is very robust to 3D transforms and non-rigid smooth transforms. Second, its performance is similar to the top methods in traditional texture retrieval and classification on standard texture data sets, but by using much lower dimensional feature vectors.

Fractal geometry has been used before in the description of textures ([81, 82]) and texture segmentation ([83, 84, 85]). However, the invariance of the fractal dimension to bi-Lipschitz maps has not been utilized in the vision community. Furthermore, existing approaches either simply compute a single fractal dimension, a scalar number, which does not provide enough information for a good characterization of the texture; or they utilize the multi-fractal-spectrum for local feature description, in which there is no global invariance to bi-Lipschitz maps. Moreover, the fractal dimension was defined only on the image intensity, which makes it sensitive to changes in luminance. Intuitively, the MFS (multi-fractal-spectrum) is the extension of the fractal dimension. The MFS encodes the fractal dimension of every level set under a level set categorization of image points. Different categorizations of image points lead to different MFS vectors. The categorizations in our MFS

approach are based on the *density function* defined on (functions of) the image intensity, and this makes it robust to luminance variations.

This chapter is organized as follows. Section 2 gives a brief introduction to fractal geometry and multi-fractal spectrum theory. Section 3 presents our MFS texture descriptor and its invariance to spatial bi-Lipschitz transforms and local affine illumination changes. Section 4 provides experiments on texture retrieval and classification and a comparison to other methods. Section 5 presents the conclusions and future work.

4.2 Fractal theory and textures

4.2.1 Brief introduction to Fractal Geometry

We first give a brief introduction to fractal theory. In the later eighties it has been realized that irregular objects provide a much better representation of many natural phenomena than do the regular objects of classical geometry. Fractal geometry was developed, which provides a general framework for studying irregular sets as well as regular sets. The term "fractal" was coined in 1975 by Benoit Mandelbrot [86], meaning "broken" or "fractured'. Mathematically, a fractal is a geometric object whose Hausdorff dimension greater than its topological dimension. Fractal properties include scale independence, self-similarity, complexity, and infinite details. Fractal theory offers methods for describing the inherent irregularity of natural objects. In fractal analysis, the Euclidean concept of "length" is viewed as a process characterized by a constant called the *fractal dimension*.

Fractal dimension is the key quantity in the study of fractal objects. Fundamental to the fractal dimension is the concept of "measurement at scale δ". For each δ, we measure an object in a way that ignores irregularity of size less than δ, and we see how these measurements behave as δ goes to 0. In other words, given a fractal object, how many balls of radius δ would cover this object; and how does this number scale as δ becomes smaller.

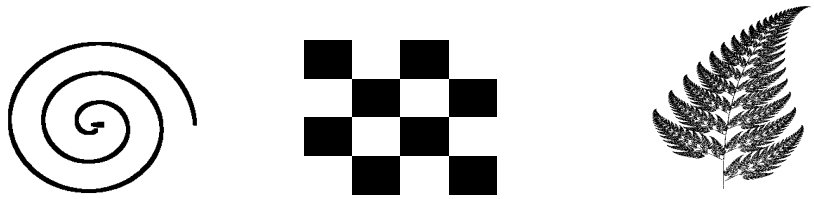
Scientists found that most of the natural phenomena satisfy the power law, which states that the the estimated quantity (for example the length of a coastline [86]), is proportional to $(\frac{1}{\delta})^D$ for some D. For most natural objects, D is almost the same for small scales δ. Thus we can compute its limit, which is called the *fractal dimension*. For the case of an irregular point set defined on R^2, the fractal dimension of the set E is defined as

$$\dim(E) = \lim_{\delta \to 0} \frac{\log N(\delta, E)}{-\log \delta}, \tag{4.1}$$

where $N(\delta, E)$ is the smallest number of sets of diameter less than δ that cover E.

Closely related to this approach is the so-called *box-counting dimension*, which considers, if the space were divided up into a grid of the boxes of size δ, how does the number of boxes scale, which contains that object. It is easy to see these two approaches are more or less the same.

Intuitively, the fractal dimension is a summary statistics measurement, which measures the overall "complexity". It gives a global description of how complex or how irregular a geometric object is. Like many summary statistics (e.g. mean), it is obtained by "averaging" variation in data. Some information is necessarily lost. For



(a) (b) (c)

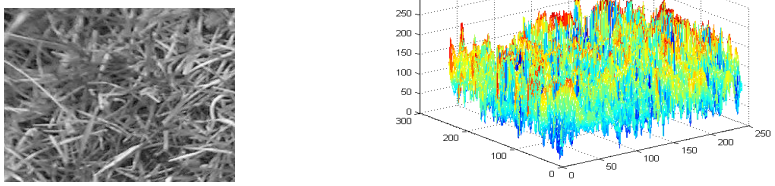
Figure 4.1: Fractal dimension D in 2d space. (a) Smooth spiral curve with $D = 1$. (b) The checkerboard with $D = 2$. (c) Fractal fern leaf with $D \approx 1.7$.

example, it tells us nothing about the actual size or overall shape of a object, nor can we reproduce the object from its fractal dimension alone. However, the fractal dimension of the object does tell us a great deal about the relative complexity of the object, and as such it is an very powerful and important descriptor.

The fractal dimension D of any object in 2D space is in the range of $1 \leq D \leq 2$. If the object's fractal dimension is integer, then it is a regular object. For example, any smooth curve has a fractal dimension of 1, and a completely filled rectangle or ellipse has a fractal dimension of 2, which are the same as their integer topological dimensions. Irregular sets have a fractional dimension between 1 and 2. Indeed, most man-made geometric objects have an integer fractal dimension D, while most objects in nature have a fractional fractal dimension. An illustration is given in Fig. 4.1.

4.2.2 Fractal dimension and textures

The question arises then whether natural textures are objects with fractional dimension information. Let's check first some typical texture. A typical grass tex-



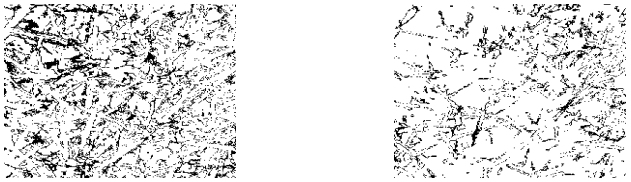
(a) (b)

Figure 4.2: (a) Original texture image. (b) 3D surface visualization of the image with $D = 2.79$.

ture image from [87] is shown in Fig. 4.2 (a). If we represent the image as a 3D-surface in Fig. 4.2 (b) by taking the image intensity as the height, it is easy to see that such a surface is a highly irregular surface. And its box-counting fractal dimension in 3D is actually 2.79.

So, the answer is positive: nature textures general are fractal objects and the fractal dimension does encode its regularity information. However the fractal dimension alone, as defined above, does not provide a rich description. As a scalar number, it is a global parameter which measures the "overall worst" behavior and does not account for a possible variability of the degree of regularity. Physicists and Applied Mathematicians have developed multifractal analysis as an extension to classical fractal analysis for more detailed and more powerful description.

To overcome the drawback of the scalar fractal dimension, the concept of the fractal dimension is extended to the concept of the MFS (Multifractal spectrum) as follows: First define a point categorization on the object function according to some criteria. Then the fractal dimensions are calculated for every point set from this



(a) (b)

Figure 4.3: (a) Black-white image obtained by thresholding intensity values between 100 and 120, its box-counting fractal dimension $D = 1.67$. (b) Black-white image obtained by thresholding intensity values between 80 and 100, its box-counting fractal dimension $D = 1.49$.

categorization. A such defined MFS gives a rich description of the inherent texture structure by providing a compact representation of the "spectral decomposition" of the function into parts of equal strength based on some strategy. Different strategies lead to different types of multifractal spectra.

To give a demonstration, we first take a simplistic categorization of image points based on the gray values of the pixels. The image pixels are categorized by their intensities. Each point set includes the image pixels whose intensity values fall into the same interval. Then for every point set, we obtain a black-and-white image by assigning 0 to pixels in this set and 1 to all other pixels. The collection of the fractal dimensions for all the black-and-white images is called an MFS. Two examples of such obtained images are shown in Fig. 4.3 (a) and Fig. 4.3 (b). Clearly these are fractal objects with fractional fractal dimensions.

As we mentioned earlier, different criteria for defining the point categorization lead to different fractal spectra. The simplistic approach demonstrated above has serious drawbacks in practice. A categorization based on image intensity is nu-

merically unstable in its computation, and it is sensitive to illumination changes. Therefore, we will adopt a more sophisticated categorization method which Mathematicians have developed in the past. Such a categorization is based on the idea of the "density function". The density function can be defined on different functions of the image intensity. Intuitively, the density function of a quantity defines the change of that quantity over scale. Our categorization leads to an MFS which is robust to illumination changes, as well as better discrimination. It is also important to point out that there exists an efficient and robust algorithm for computing it [88].

4.3 MFS for textures

The MFS is the vector of the fractal dimensions of some categorization of the image points. Using the idea of the local density function in the categorization has the advantage that we can compute the fractal spectra in a robust and efficient way.

4.3.1 General model of the MFS for texture

Let μ be a finite Borel regular measure on R^2. A regular measure is a function defined over a set of R^2 with the following important properties:

1. The empty set $\oslash$ has measure zero: $\mu(\oslash) = 0$.

2. μ is monotonic: $\mu(E_1) \leq \mu(E_2)$, if $E_1 \subseteq E_2$.

3. μ is a countable additive: if $E_1, E_2, \cdots$ is countable sequence of pair-wise

disjoint sets, then:

$$\mu(\cup_{i=1}^{\infty} E_i) = \sum_{i=1}^{\infty} \mu(E_i).$$

The details of μ for texture images will be discussed in the practical algorithm thereafter. Let's assume we have such a μ in hand. For $x \in R^2$, denote $B(x, r)$ the closed disc with center x and radius $r > 0$. It is easy to see that $\mu(B(x, r)) \to 0$ and $\log \mu(B(x, r)) \to \infty$ when $r \to 0$. Then the *local density function* of x is defined as

$$D(x) = \lim_{r \to 0} \frac{\log \mu(B(x, r))}{\log r}. \tag{4.2}$$

For $\alpha \in R$, define

$$E_\alpha = \{x \in R^2 : D(x) = \lim_{r \to 0} \frac{\log \mu(B(x, r))}{\log r} = \alpha\}. \tag{4.3}$$

That is, E_α is the set of all image points x with the local density α. Usually this set is irregular and has a fractional fractal dimension $dim(E_\alpha)$. Thus we obtain a point categorization $\{E_\alpha : \alpha \in R\}$ of the image with an MFS denoted as

$$f(\alpha) = \{dim(E_\alpha) : \alpha \in R\}. \tag{4.4}$$

The MFS $f(\alpha)$ defined by (4.4) is the natural extension of the concept of the fractal dimension, and it has clear physical meaning. However, the computation of the MFS directly based on such a definition is not stable in practice. The process of estimating local density for each point would suffer seriously from the limited resolution of the image in practice. Fortunately, there exists another equivalent expression for $f(\alpha)$ based on generalized moment measurements. (See [88] for more details.) The new expression of the MFS is both efficient and robust in practical computations.

4.3.2 Practical algorithm for computing the MFS

First, we partition R^2 into a r-mesh squares set $\{B(x,r)\}$. $\{B(x,r)\}$ is the collection of all disks with length r which are pair-wise non-overlapping. Then we define the q-th moment of a measure μ on the r-mesh squares $\{B(x,r)\}$ as

$$M_r(q) = \sum \mu(B(x,r))^q, \tag{4.5}$$

where the sum is over the r-mesh squares $\{B(x,r)\}$ for which $\mu(B(x,r)) > 0$. For a series of $r = 1, 2, \cdots, n$, we have corresponding measurements $M_r(q)$. Then the power law behavior of $M_r(q)$ is identified by the number $\beta(q)$ which is defined as:

$$\beta(q) = \text{Slope of } \log(M_r(q)) \text{ vs } \log(r). \tag{4.6}$$

In other words, $\beta(q)$ is the slope of the line which best fits the data set $\{\log(M_r(q)), \log(r)\}$. It is shown in [88] that the MFS and $\beta(q)$ are related to each other by a Legendre transform as

$$f(\alpha) = \inf_{-\infty \le q \le \infty} (\beta(q) + \alpha q). \tag{4.7}$$

Using equations (4.7), we can estimate the MFS. The reliability of $f(\alpha(q))$ is measured by the residual of fitting $\beta(q)$ in (4.6).

Intentionally we haven't given the definition of the measurement function $\mu(B(x,r))$ yet. The first approach is to work directly on the intensity domain. Define $\mu(B(x,r))$ as

$$\mu(B(x,r)) = \int_{B(x,r)} (G_r * I) dx, \tag{4.8}$$

where $''*''$ is the 2D convolution operator and G is a Gaussian smoothing kernel with variance r. In other words μ is the average intensity value inside the disc

$B(x,r)$. This results in the definition of the density of the intensity function, which describes how the intensity at a point changes over scale (as we change the size of the neighborhood). Finally, the corresponding MFS encodes the fractal dimension for multiple values of the density of the intensity. The algorithm is summarized below.

Algorithm 1 Computation of the MFS.

1) Compute $M_r(q)$ from (4.5) and (4.8).

2) Estimate $\beta(q)$ from (4.6) by fitting $M_r(q)$ linearly.

3) Compute $f(\alpha)$ from (4.7).

In practice the measurement function $\mu(B(x,r))$ is not very robust to large illumination changes. But we can define the density function on other quantities. One can imagine many meaningful definitions for $\mu(B(x,r))$, such that the corresponding MFS is less effected by illumination changes. One choice is to define $\mu(B(x,r))$ on the energy of the gradients. Consider $\{f_k, k = 1,2,3,4\}$ to be four directional differential operators along the vertical, horizontal, diagonal and anti-diagonal directions. Then we define the measurement function $\mu(B(x,r))$ for the image texture I as:

$$\mu(B(x,r)) = \int_{B(x,r)} \sum_k (f_k * (G_r * I))^2 dx)^{\frac{1}{2}}. \tag{4.9}$$

Another meaningful choice for $\mu(B(x,r)$ is the sum of the Laplacians of the image inside $B(x,r)$, i.e

$$\begin{aligned} \mu(B(x,r)) &= \textstyle\int_{B(x,r)} |\nabla^2 (G_r * I)| dx \\ &= \textstyle\int_{B(x,r)} |\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) G_r * I| dx. \end{aligned} \tag{4.10}$$

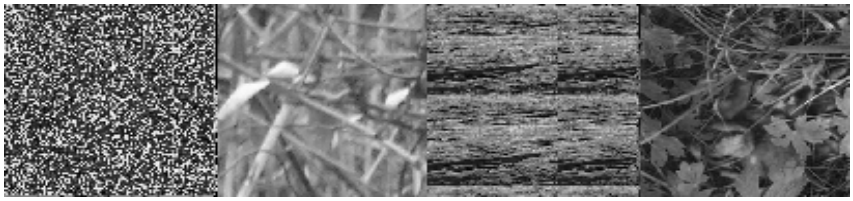


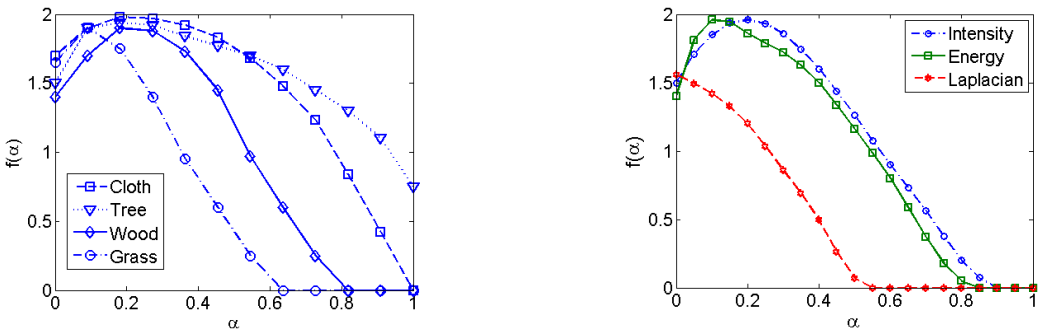
Figure 4.4: Four textures: cloth, tree, wood, grass from left to right.

(a) (b) (c)

Figure 4.5: (a) Intensity image of the grass texture. (b) The energy of image gradients as defined in Eqn. (4.9). (c) The energy of the Laplacian of the image.

Different definitions of $\mu(B(x,r))$ lead to different MFS which capture different aspects of the texture's structure. Multiple MFS could and should be combined to better describe the texture.

Fig. 4.6 (a) shows the MFS (based on (4.9)) for the four textures in Fig. 4.4. It demonstrates that the MFS of the different textures are significantly different. Fig. 4.6 (b) shows the three MFS based on the three measurement functions μ described above. From this and other textures we found that the density of the intensity and the density of the edge energy give similar MFS, whereas the density of the Laplacian leads to a significantly different MFS.



(a) (b)

Figure 4.6: (a) The MFS of the intensity for the four textures shown in Fig. 4.4. (b) The MFS for all three measurement functions shown in Fig. 4.5

.

4.4 Invariance of the MFS to various deformations

4.4.1 Spatial invariance

A very attractive property of the MFS is its is invariance under the so-called *bi-Lipschitz* transform. A function $f : R^2 \to R^2$ is called a *bi-Lipschitz* transform, if there exist two constants $c_1 > 0$, $c_2 > 0$ such that for any $x, y \in R^2$,

$$c_1\|x - y\| < \|f(x) - f(y)\| < c_2\|x - y\|, \tag{4.11}$$

where $\|x-y\|$ is the Euclidean metric between points x and y. Basically, any smooth transform is a bi-Lipschitz transform. It is a very general mapping which includes translation, rotation, perspective transformation, and texture warping on regular surfaces. Then we have the following theorem (See Appendix in this chapter for the proof).

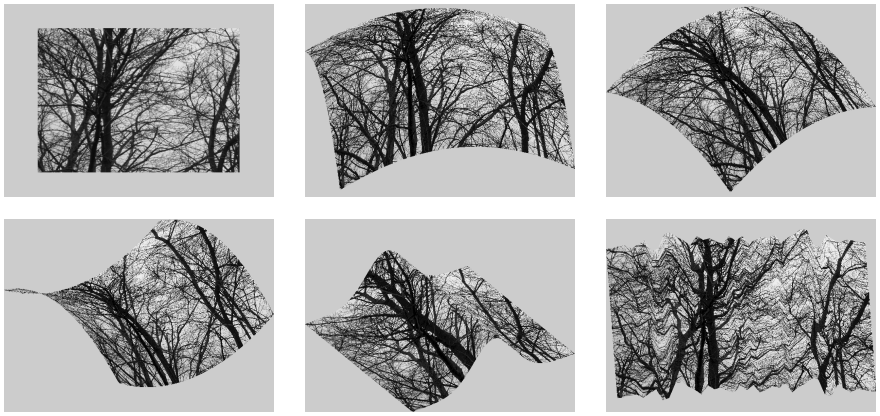


Figure 4.7: Perspective images of texture *tree* on different general smooth surfaces.

Theorem 4.1. *The MFS is invariant under any spatial bi-Lipschitz transform.*

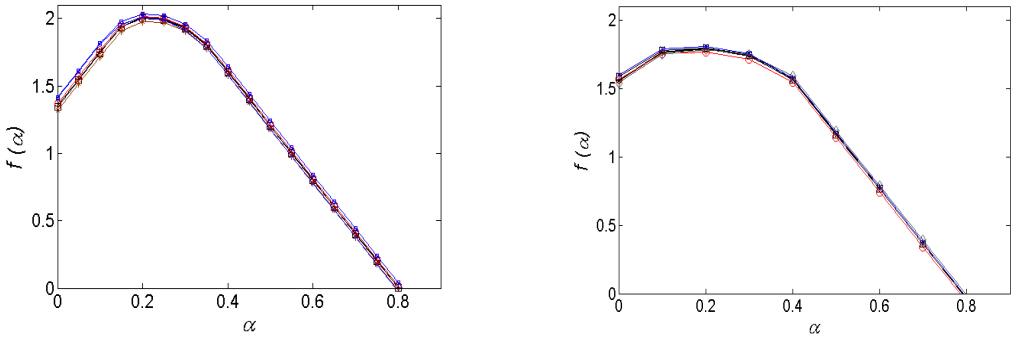
Thus, the MFS not only is invariant to perspective transformations, but also to non-rigid deformations.

Let us stress that the invariance of the MFS to bi-Lipschitz maps has been proven for images of infinite resolution. In practice we are dealing with finite resolution images. Nevertheless, we found that the MFS is very robust to perspective transforms and surface warping, even for low resolution images. Usually four to five levels of resolution (defined by radius r) are sufficient for a good estimation of the MFS.

Two examples are given: Fig. 4.8 shows the same plant texture on a plane seen from different view-points. It can be seen that the corresponding MFS in Fig. 4.9 (a) are nearly the same.

Fig. 4.7 shows six images of a tree texture on general smooth surfaces under perspective projection. Their MFSs are given in Fig. 4.9 (b). Although there is some self-occlusions in these images, the MFS are nearly the same.

Figure 4.8: Six perspective texture images of the *foliage* texture.



(a) (b)

Figure 4.9: (a) The MFS of the intensity for six perspective views of the *foliage* textures in Fig. 4.8. (b) The MFS of the intensity for the tree texture warped on six different surfaces as shown in Fig. 4.7.

4.4.2 Illumination invariance

Another important factor to be considered is the illumination. In theory it can be proven that the MFS (for all three definitions of measurement function μ) is illumination invariant.

In practice we don't have illumination variance. We can show, however, that for the density defined on first order derivatives and the Laplacian the MFS is locally invariant to affine changes in illumination. For the density defined on the intensity we have invariance to multiplicative changes. Consider, a brightness change due to a constant being added to each image pixel. This will not affect $\mu(B(x,r))$ (as defined in 4.9 and 4.10), because these measures are based on pixel differences. Next, consider a multiplicative change of each pixel value by a constant. This will just multiply $M(r,q)$ by a constant. Thus, this change will also not affect the estimation of $\beta(q)$, because the slope of the line fitting $log(M(r,q))$ is invariant to this multiplication. Therefore, the MFS is invariant to affine (multiplicative) changes in illumination. Even non-linear illumination changes do not much effect the MFS, because the fractal dimension is quite robust to small variations of its point set.

Fig. 4.10 shows nine real texture images taken by a family digital camera. The three images in each row depict the same scene from different view-points under different illumination conditions. The computed MFS is shown in Fig. 4.11. As can be seen the MFSs of different textures are significantly different, while the MFSs of the same texture under different view-points are almost identical. Fig. 4.13 shows

Figure 4.10: Three real 3D texture images under different lighting conditions and view points.

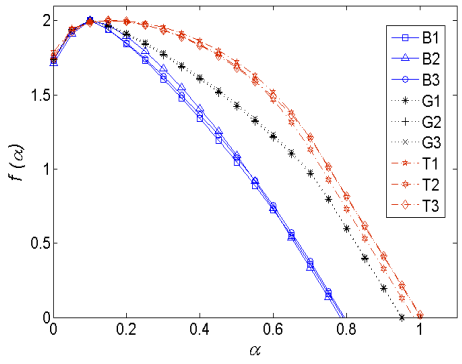


Figure 4.11: The MFS of the intensity for the nine texture images in Fig. 4.10. B1, B2, B3 are bulrushes, G1, G2, G3 are grasses and T1, T2, T3 are trees.

Figure 4.12: One frame of a human gesture video.

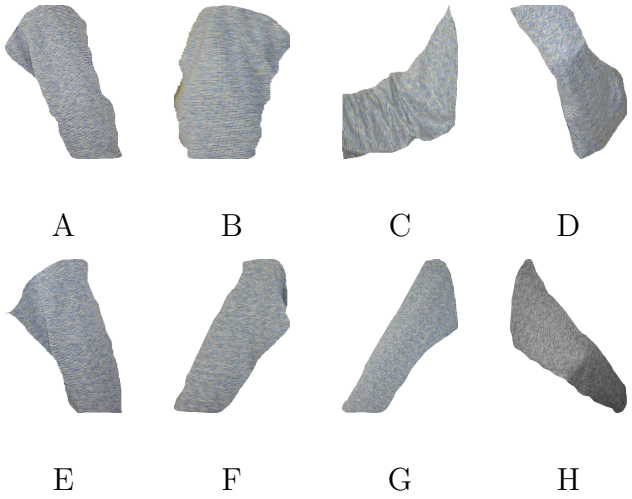


Figure 4.13: The arm parts of the human gesture video under different illumination conditions and views.

eight views of a cloth texture on the moving arm of a person. The illumination is not controlled. The corresponding MFSs are shown in Fig. 4.14, again demonstrating the robustness under geometric and illumination deformations.

4.4.3 Comparison to other texture descriptors

The most popular global texture descriptor is the histogram. It could be interpreted as a descriptor defined on some categorization of the image, such as for example a categorization of pixels based on intensity thresholding. The descriptor for each category is simply the number of pixels in the category. The drawback of

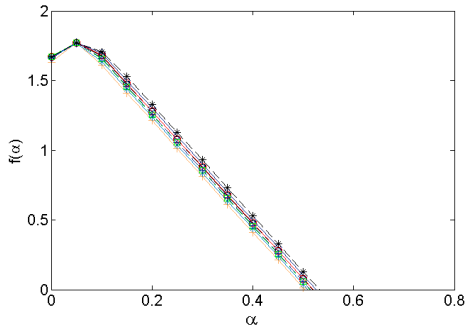


Figure 4.14: The MFS of the intensity corresponding to Fig. 4.13.

such a description is that the spatial information of how pixels are distributed is lost. In contrast, the fractal dimension encodes the spatial information (as irregularity). The intensity-based histogram is quite robust to small changes, but not invariant to perspective transformations, and it is very sensitive to changes in illumination. The modified texton-based histograms could reduce the effects of illumination but in trade-off to larger sensitivity to global geometric transforms. In contrast, the MFS is capable of incorporating various low-level local feature measurements, such as the Laplacian, the energy, or Gabors without sacrificing global spatial invariance.

Another type of texture descriptor utilizes clusters of locally spatial invariant descriptors of textons (elements of textures). In the following section we compare our approach against the sophisticated texture representation of Lazebnik et al [3], which we call the LA-method. The basic idea of the LA-method is to characterize the texture by clusters of elliptic regions. The elliptic regions are localized with the Harrison (H) and Laplacian (L) operators. The ellipses are transformed to circles. Thus this descriptor is locally invariant to affine transforms. Each region is represented by two descriptors: one encoding the histogram of the intensity distribution

in a neighborhood (S); the other encoding the histogram of edges in a neighborhood (R) (a variation of SIFT features).

The LA-method essentially represents an image texture by its pattern of texture elements. It doesn't have global spatial invariance, because the pattern itself is the 2D projection of a 3D pattern. Different view-points lead to different patterns. The dimension of the LA texture descriptor is very high (thousands for a typical image texture). Furthermore, the LA-method requires sophisticated preprocessing to compute the texture elements and k-mean clustering to construct a good texture descriptor. In comparison, the MFS is simple and straightforward to compute as it neither relies on feature detection nor on clustering techniques.

4.5 Experimental evaluation and summary

We evaluated the performance of the MSF descriptor on classic texture *retrieval* and *classification* and compared it to two top methods: the LA-method (described before) and VZ-Joint method ([4]). Varma and Zisserman's VZ-Joint method is a non-invariant algorithm which uses a dense set of 2D textons. The descriptor is a one-dimensional texton histogram encoding the joint distribution of all pixel values in the neighborhood. For the comparison we use the data sets provided in ([3]), which consists of 1000 uncalibrated, unregistered images of size 640×480: 25 different textures each with 40 samples (See Fig. 4.15, which is available at http://www-cvr.ai.uiuc.edu/ponce).

Within each class, significant viewpoint changes and scale differences are present,

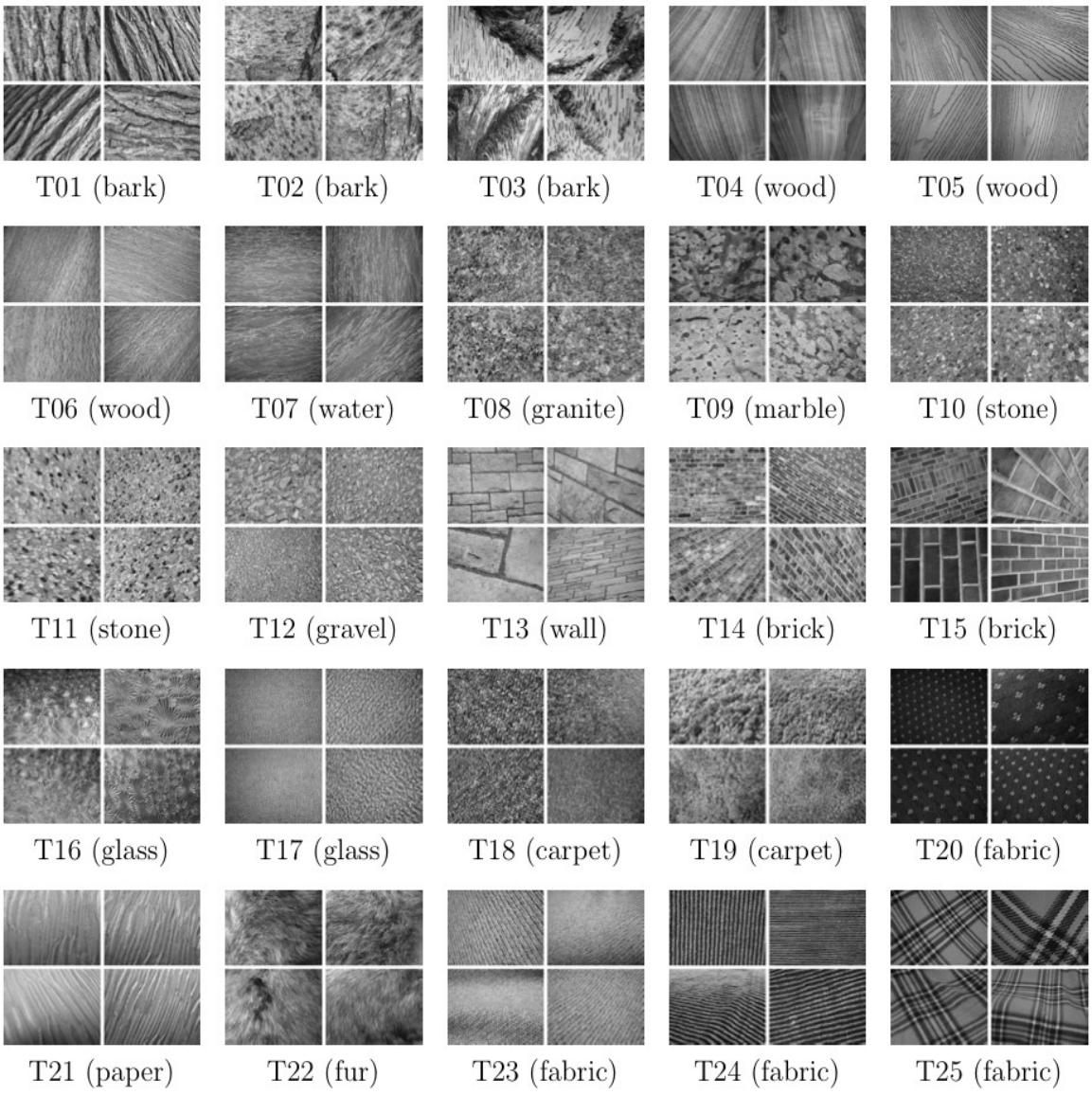


Figure 4.15: Four samples each of the 25 texture classes in Ponce's data sets.

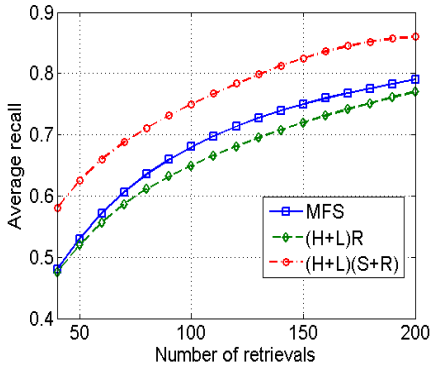
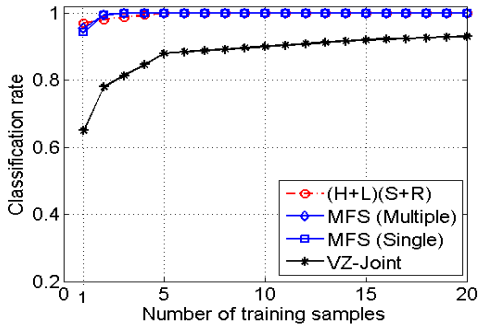


Figure 4.16: Retrieval curves for the Ponce database by our method and the methods in [3].

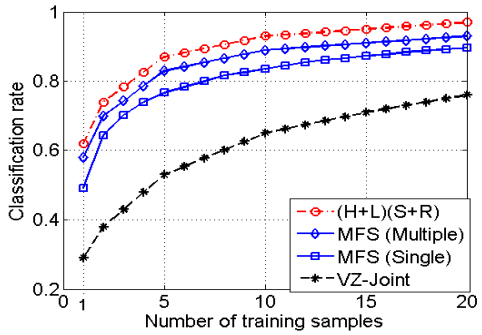
and the illumination conditions are uncontrolled. Additional sources of variability include non-planarity of the textured surface, significant non-rigid deformations between different samples of the same class, inhomogeneities of the texture pattern, and viewpoint-dependent appearance variations. (For details see [3].)

We used three MFS (as defined by the measurement functions 4.8, 4.9 and 4.10), and for each vector we computed 26 values. There is no clustering in our method. As distance function between the MFS we used the weighted L_1 norm. The weight coefficients are from the residuals in the computation of $\beta(q)$.

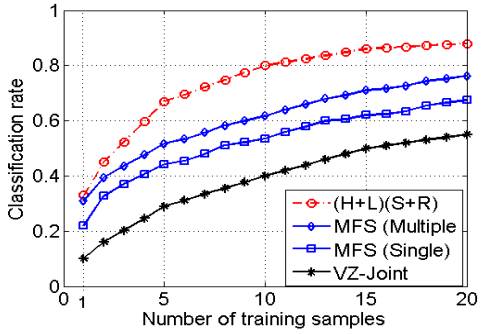
Retrieval and classification are implemented as described in [3]. Briefly, for retrieval, given a query image we selected other images from our database in an increasing order of the distance, i.e., from the most similar to the least similar. Each image in the database is used as a query image once, and the performance is summarized as a plot of average recall vs. the number of retrievals. For classification we used the nearest-neighbor technique. The training set is selected as a fixed size



(a)



(b)



(c)

Figure 4.17: Classification rate vs. number of training samples. Three methods are compared: the MFS method, the (H+L)(S+R) method in [3] and the VZ-Joint method in [4]. (a) Classification rate for the best class. (b) Mean classification rate for all 25 classes. (c) Classification rate for the worst class.

random subset of the class, and all remaining images are used as test set. The reported classification rate is the average over 200 random subsets. The results for the methods [3] and [4] are from the paper [3].

Fig. 4.16 presents a comparison of our method to the techniques in [3] for retrieval. Our method performs better than the "(H+L)R" channel of the LA-method, but not better than the "(H+L)(S+R)" channel. The reason is that the MFS is not as robust as [3]'s methods to illumination changes. [3]'s robustness is due to the use of only edge and corner information. In addition there is parameter-dependent clustering. It is worth stressing that the dimension of the MFS is significantly smaller than the ones of the methods in [3] (seventies vs thousands).

Fig. 4.17 evaluates classification and shows the influences of the number of samples on the classification rate. The mean classification rate is plotted for our method using 3 MFS vectors as well as for a single MFS defined on the density of the intensity (26 numbers only). The figure demonstrates that our method has a classification rate which is slightly worse than the LA-method, but generally much better than the VZ-Joint method. It is worth mentioning that the MFS performs worse than the LA-method only in four of the 25 classes; it is those with significant illumination changes. For all other classes it actually has a very impressive classification rate (see Figure 4.17a and c).

We would like to address that a major advantage of our method is its low dimension. Also, our algorithm will behave much better for larger images. The images used in the experiment are not large enough to obtain very good estimates of the fractal dimension. As a result the estimated MFS is not perfectly invariant

anymore. The performance of our method increases a lot with an increased image size. In comparison, the feature-based method ([3]) will not benefit much from larger image size, as it strongly depends on clustering.

In summary, we showed both in theory and experiment that the MFS offers a new and promising efficient texture description. Not only it is comparable to other top methods in classifying textures, but also it has a mathematically justified global bi-Lipschitz invariance. To our knowledge, no other approach proposed in the past has such global properties. In practice, the MFS is very robust to spatial transformations and somewhat less robust to changes in illumination, but comparable to feature-based methods. Experiments on standard data sets demonstrated its effectiveness and efficiency in texture retrieval and classification tasks, achieving a performance comparable to the top feature based methods but with far lower dimension (seventies vs thousands). Furthermore, the MFS is very efficient and simple to compute without requiring feature detection and clustering. In future work, we will investigate how to better combine MFS from different measurement functions in order to achieve greater robustness to illumination changes.

4.6 Appendix: Proof of Theorem 4.1

Let f denote the bi-Lipschitz transform, then the new image after applying such a transform on the original image $I(x)$ could be written as $I'(x) = I(f(x))$. In order to prove the invariance of the MFS to bi-Lipschitz transforms, we first prove that

for each α,

$$E'_\alpha = f(E_\alpha),$$

where E'_α denotes the corresponding level set of $I'(x)$. Following, we prove that $f(E_\alpha)$ has the same fractal dimension as E_α.

For a constant α, let x be any point in the set of E_α, then from (4.3) we have

$$\lim_{r\to 0} \frac{\log \mu(B(x,r))}{\log r} = \alpha.$$

Thus, for the point $f(x)$ in the new image $I'(x) = I(f(x))$, we have

$$\begin{aligned} \frac{\log \mu((B(f(x),r)))}{\log r} &\leq \frac{\log c_2^2 \mu(B(x,r))}{\log r} \\ &= \frac{2\log c_2 + \log \mu(B(x,r))}{\log r}. \end{aligned}$$

Similarly, we have

$$\frac{\log \mu(B(f(x),r)))}{\log r} \geq \frac{2\log c_1 + \log \mu(B(x,r))}{\log r}.$$

Taking the limit on both sides of the two inequalities above, we obtain

$$\lim_{r\to 0} \frac{\log \mu(B(f(x),r))}{\log r} = \lim_{r\to 0} \frac{\log \mu(B(x,r))}{\log r} = \alpha.$$

Thus, we conclude that $E'_\alpha = f(E_\alpha)$.

To complete the proof, we need to show that the fractal dimension of $f(E_\alpha)$ and the fractal dimension of E_α are the same. Suppose E_α is covered by $N(\delta, E_\alpha)$ sets, which is the smallest number of sets with diameter less than δ. Then the $N(\delta, E_\alpha)$ images of these sets under f form a cover of $f(E_\alpha)$ by sets of diameter less than $c_2\delta$. By the definition the fractal dimension, we have

$$\dim(f(E_\alpha)) = \lim_{c_2\delta\to 0} \frac{\log N(c_2\delta, f(E_\alpha))}{-\log c_2\delta} \leq \lim_{\delta\to 0} \frac{\log N(\delta, E_\alpha)}{-\log c_2\delta}$$

$$
\begin{aligned}
&= \lim_{\delta \to 0} \frac{\log N(\delta, E_\alpha)}{-\log c_2 - \log \delta} = \lim_{\delta \to 0} \frac{\log N(\delta, E_\alpha)}{-\log \delta} \\
&= \dim(E_\alpha).
\end{aligned}
$$

On the other hand, suppose $f(E_\alpha)$ is covered by $N(\delta, f(E_\alpha))$ sets, which is the smallest number of sets of diameter at most δ. The same argument yields

$$
\dim(E_\alpha) \leq \lim_{\delta \to 0} \frac{\log N(\delta, f(E_\alpha))}{-\log c_1^{-1} - \log \delta} = \dim(f(E_\alpha)).
$$

Thus, $\dim(E_\alpha) = \dim(f(E_\alpha))$. The MFS of the new image $I(f(x))$ equals the MFS of the original image $I(x)$. $\square$

Chapter 5

Application: Wavelet-based Super-resolution imaging

5.1 Overview

The problem of *super-resolution* imaging, which is defined as restoring a high-resolution (HR) image from a sequence of low-resolution (LR) images (See Fig. 5.1), has been studied by many researchers in recent years. Most super-resolution algorithms formulate the problem as a two-stage process: one is aligning the images of the sequence; the other is reconstructing a high-resolution image from the aligned image frames. Both processes are critical for the success of the super-resolution reconstruction. There is extensive literatures on solving image alignment problem. Many researchers takes a flow-based approach for image alignment ([89, 90, 91, 92]). Here we adapt the algorithm described in Chapter 3 to the case of large image displacement to provide a homography-based alignment of multiple frames. This is described in Section 5.6. The remained of the chapter focuses on how to reconstruct a HR image from aligned LR images.

Image restoration remains an important research topic since digital computers made the process of large amounts of data possible. Image restoration may be broadly categorized into two classes based on the number of observed frames: single-input and multi-input. While the field of single-frame image restoration appears to have matured, the processing of digital video is still in its infancy. Since video

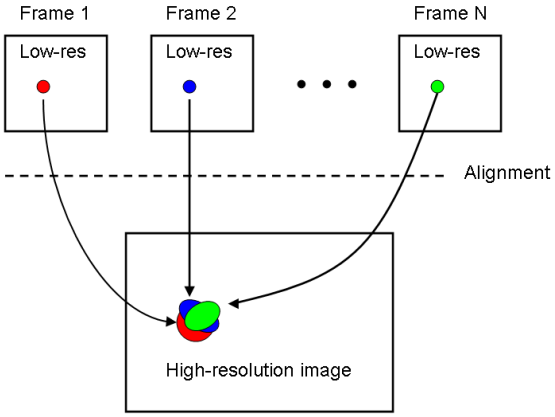


Figure 5.1: Illustration of super-resolution imaging.

typically consists of a sequence of similar, though not identical frames, it becomes possible to use all the information contained in an image sequence to increase spatial resolution of the still image. However, such a process is difficult. It turns out that super-resolution restoration is an ill-posed inverse problem:

1. Usually, a LR image sequence is acquired by an imaging system recording a scene. The imaging process creates the observed image sequence. The corresponding inverse problem is to determine estimates of the scene given the observed image sequence and a model of the image formation process. Thus *restoration* is the inverse problem and *simulation* is the forward problem, given the image formation process.

2. Super-resolution image restoration is an ill-posed problem according to the Hadamard conditions.

   (a) Non-existence of the solution. The presence of noise in the image forma-

tion process may result in an observed image sequence which is inconsistent with any scene.

(b) Non-uniqueness of the solution. The operator, which characterizes the image formation process, is a many-to-one mapping. Thus, there exists a nontrivial space of solutions consistent with any given image sequence.

(c) Discontinuous dependence of the solution on the given data. Depending on the characteristics of the image formation process, the inverse problem may be highly sensitive to perturbations of the date. For example, consider an imaging system with a spectral response which decreases asymptotically toward zero with increasing frequency. An arbitrarily small noise component at sufficiently high frequencies will lead to an arbitrarily large spurious signal in the computed restoration.

Despite the difficulties caused by the ill-posedness of super-resolution image restoration, researchers have made great progress toward stable algorithms. Iterative back-projection methods ([89, 93]) have been shown to be effective for high-resolution image reconstruction. It is known, however, that the de-blurring process, which is part of this approach, makes it very sensitive to the noise. Thus, the requirement of very accurate image alignment estimates limits its practical use. Various regularization methods have been proposed to deal with the noise issue. However, these methods either are very sensitive to the assumed noise model (Tikhonov regularization) or are computationally expensive (Total-Variation regularization). See [94] for more details.

Our contributions to the reconstruction process are two-fold. First, we model the image formation procedure from the point of view of filter bank theory . Then based on this new formulation, we provide an analysis of the limits of the high-resolution reconstruction. The conclusion is that in general full recovery is not possible without enforcing some constraints on the recovered images. At best we could reconstruct the image convolved with a specific low-pass filter (namely $\frac{1}{4}(1,1)\otimes(1,1)$ for the case of the Box-type PSF).

Second, based on our new formulation, we present a robust wavelet-based algorithm to reconstruct the image. The iteration scheme in our algorithm is inherently more robust to noise than that of classic back-projection methods ([89, 93]), since the projection matrix of our new back-projection scheme has a better condition number. We will show that, both in theory and experiments, it has better performance in suppressing the error propagation than other back-projection iteration schemes.

Furthermore, our algorithm allows us to include a wavelet-based de-noising scheme in each iteration of the reconstruction which effectively removes the noise without creating smoothing artifacts. The advantage of our de-noising scheme over regularization methods is that it is nearly optimal with respect to the risk bound. That is, it has the theoretical minimal error in removing noises of unknown models. Its effectiveness in removing mixed noises and relatively large amounts of noise is demonstrated in experiments. It is worth mentioning that our de-noising scheme adds very little computational burden compared to other complicated regularization methods. Briefly, our method could be described as a generalized iterative back-

projection method with a fast and optimal regularization criteria in each iteration step.

Wavelet theory has previously been used for image de-noising and de-blurring from static images ([95, 96]). However, it has not been studied much with respect to the super-resolution problem. In recent work wavelet theory has been applied to this problem [97], but only for the purpose of speeding up the computation. Our contribution lies in an analysis that reveals the relationship between the inherent structure of super-resolution reconstruction and the theory of wavelet filter banks. This relationship is fully exploited by using various techniques from wavelet theory in the iterations of the reconstruction.

5.2 Formulation of high-to-low image formation

We first formulate the high-to-low image formation process in the same way as [98] did. To simplify the exposition, in the following we only discuss 1D signals with resolution enhancement by a factor 2. Later, without much difficulty, the analysis will be extended to the 2D case with arbitrary resolution enhancement.

Adopting Farsiu's notation ([94]), the image formation process in the pixel domain can be modeled as

$$y = \sigma[H * X(F(t))] + N, \tag{5.1}$$

where t is the spatial variable; $X(t)$ is the continuous signal and y is the discrete signal; H is the blurring operator (either optical blurring or motion blurring or both); F is the geometric transform; N is the noise in the low-resolution image; σ is

the decimation operator; and "$*$" is the convolution operator. Not considering the noise now, the high-resolution (HR) signal x and the low-resolution (LR) signal y can be defined as:

$$x = \sigma[X], \quad y = [\sigma[H * X(F(t))]] \downarrow_2 . \tag{5.2}$$

where $\downarrow_2$ is the downsampling operator with rate 2.

Next we derive the relationship between the LR signal y and the HR signal x. Define the velocity of the signal by $\epsilon(t) = F(t) - t$, which is also called the *optical flow* in computer vision. For the simplicity of notation, here we assume a sub-pixel flow model with $0 < \epsilon(t) < 2$ on the denser grid of the HR image x (Larger flow could always be reduced to the case of sub-pixel flow by re-assigning the pixel value). Thus, in the LR image the flow values are all sub-pixel shifts (Recall a 1-unit shift on the coarse grid of y equals a 2-unit shift on the fine grid of x).

Let $\{j\}$ be a fine grid for the spatial coordinates x , then for point j of y on the coarse grid (it's coordinats is $2j$ in the fine grid) with $0 \leq \epsilon(2j) < 1$, the first-order Taylor approximation of Equation (5.2) at point $2j$ can be written as

$$\begin{aligned} y(j) &= [H * X(F(t))]_{t=2j} \\ &= [H * X(\epsilon(t) + t)]_{t=2j} \\ &= [H * X(t)]_{t=2k} + \epsilon(2j)[H * X'(t)]_{t=2j} \\ &= [H * X(t)]_{t=2k} + \epsilon(2j)[H' * X(t)]_{t=2j}. \end{aligned}$$

For all other points j' of y with $1 \leq \epsilon(2j') < 2$, a similar argument yields

$$y(j') = [H * X(t)]_{t=2j'+1} + \epsilon(2j'+1)[H' * X(t)]_{t=2j'+1}.$$

Thus, a LR sequence y could be expressed in the pixel domain as a subsequence of the following two sequences:

$$\begin{aligned} &[a * x] \downarrow_2 + \epsilon \cdot * [b * x] \downarrow_2 \\ &a * x(\cdot + 1)] \downarrow_2 + (\epsilon - 1) * [b * x(\cdot + 1)] \downarrow_2, \end{aligned} \tag{5.3}$$

where a, b are discrete versions of the convolution kernels H and H' respectively, and $\cdot *$ denotes the component-wise multiplication operator. Having available the optical flow values ϵ_k for multiple low-resolution images y_k, we can extract the four components:

$$\begin{aligned} &[a * x] \downarrow_2, \quad [a * x(\cdot + 1)] \downarrow_2 \\ &[b * x] \downarrow_2, \quad [b * x(\cdot + 1)] \downarrow_2 . \end{aligned} \tag{5.4}$$

As will be shown in the next subsection, the two filters a and b (which are determined by the blurring kernel H and its derivative H') characterize the super-resolution reconstruction.

Let us next look at some examples of filters a and b for different blurring kernels.

Example 2.1. Consider the box-type blurring kernel $H = \frac{1}{2n}\chi_{[-n,n]}$. Let $E(t) = \epsilon \leq 1$. Then we have

$$y(j) = \int_{-\infty}^{\infty} \chi(2j - t) X(F(t)) dt = \frac{1}{2n} \int_{2j-n}^{2j+n} X(F(t)) dt = \frac{1}{2n} \int_{2j-n}^{2j+n} X(t + \epsilon) dt.$$

Approximating the integration by quadrature rules, we obtain

$$y(j) \;=\; \frac{1}{2n}(\frac{1}{2}(1 - \epsilon)x(2j - n) + \sum_{i=-n+1}^{n-1} x(2j - i) + \frac{1}{2}(1 + \epsilon)x(2j + n)).$$

Or equivalently,

$$y = [a * x + \epsilon(b * x)] \downarrow_2, \tag{5.5}$$

where a and b are the following low-pass and high-pass filters respectively:

$$a = \frac{1}{4n}(1, 2, \cdots, 2, 1), \quad b = \frac{1}{4n}(-1, 0, \cdots, 0, 1).$$

Example 2.2. Consider a Gaussian-type blurring kernel H. Using the Cubic Cardinal B-spline $B(t)$ as approximation to the Gaussian function we have

$$y(j) = \int_{-\infty}^{\infty} B(2j - t)X(F(t))dt.$$

Again, by the quadrature rule, we have the approximation

$$y = \sum_i x(2j - i)(a(i) - \epsilon b(i)),$$

where $a = \frac{1}{96}(1, 8, 23, 32, 23, 8, 1), \quad b = \frac{1}{48}(3, 12, 15, 0, -15, -12, -3)$.

5.3 Analysis of the HR reconstruction

Given multiple LR signals y_k with different motions ϵ_k, theoretically we can obtain two complete sequences $a * x$ and $b * x$ from (5.4). An interesting question arises. Without any assumption on a given finite signal x, can we reconstruct the signal sequence x exactly from these two sequences $a * x$ and $b * x$?

To answer this question, let us write the sequence in another form, namely, as its Z-transform. The Z-transform of a signal sequence $x = \{x(i)\}$ is defined as

$$x(z) = \sum_i x(i)z^{-i}.$$

It is easy to see that such a transform is a one-to-one mapping between sequence space and polynomial space. Let $a(z)$ and $b(z)$ denote the Z-transforms of the filters a and b, then the Z-transforms of $a*x$ and $b*x$ are $a(z)x(z)$ and $b(z)x(z)$ respectively.

Now the question can be addressed by checking whether the polynomial equation

$$(a(z)x(z))u(z) + (b(z)x(z))v(z) = x(z), \tag{5.6}$$

is solvable for the two unknowns $u(z)$ and $v(z)$. Eliminating $x(z)$ from both sides of (5.6) yields

$$a(z)u(z) + b(z)v(z) = 1. \tag{5.7}$$

From the theory of Diophantine equation we know the follows

Lemma 4.1. *Given two polynomials $a(z)$ and $b(z)$, (5.7) is solvable if and only if the greatest common divisor of $a(z)$ and $b(z)$ is a scalar, that is, $a(z)$ and $b(z)$ are co-prime.*

It is observed that $a(z)$ and $b(z)$ in our two examples (Example 1 and 2) both have a common divisor

$$c(z) = (1 + z).$$

This can be seen from the fact that $a(-1) = b(-1) = 0$, and therefore $z = -1$ is the root of both $a(z)$and $b(z)$. Thus, for these blurring kernels we cannot reconstruct $x(z)$ from $a(z)x(z)$ and $b(z)x(z)$ exactly. This observation is not an incident. It actually holds true for general blurring kernels, as we will show next.

We follow Baker's modeling of the blurring kernel H ([99]). The blurring kernel (Point spread function) can be decomposed into two components:

$$H = \Omega * C,$$

where $\Omega(X)$ models the blurring caused by the optics and $C(X)$ models the spatial

integration performed by the CCD sensor. Typically Ω is modeled by a Gaussian-type function and C is modeled by a Box-type function. Notice that

$$H' = \Omega' * C.$$

Thus we can express the corresponding discrete filters as:

$$a = \ell * c; \quad b = \tau * c,$$

where c is the discrete version of the spatial integration kernel C, and ℓ and τ are the discrete versions of H and H'. Since $a(z)$ and $b(z)$ have a common divisor $c(z)$, we cannot reconstruct $x(z)$ for general $x(z)$, unless C is a Dirac function, which generally is not true. Based on Lemma 1, we then have the following claim.

Claim 4.1. *Given multiple LR finite signals y_k, we can not perfectly reconstruct the HR finite signal x without any assumptions on x. At most we can reconstruct $c*x$ for some low-pass filter c. The corresponding Z-transform $c(z)$ of c is the greatest common divisor of $a(z)$ and $b(z)$, which includes the spatial integration filter.*

Notice that c is a low-pass FIR (finite impulse response) filter. To recover x from $c * x$, we have to apply a high-pass filter on $c * x$ and impose some boundary condition on the signal x. Such a de-blurring process generally is sensitive to the noise and creates artifacts in the recovered image. A good strategy then is to modify our reconstruction goal during the intermediate iterative reconstruction process: Instead of trying to reconstruct x, we reconstruct $c * x$ in the iterative process, and we leave the recovery of x from $c * x$ to the last step after finishing the iterative reconstruction.

Thus, the modified HR signal to be reconstructed is $\widetilde{x} = c * x$. Now the LR sequence $\{y_k\}$ is a subset of the following two sequences:

$$[\ell * \widetilde{x}] \downarrow_2 + \epsilon_k \cdot [\tau * \widetilde{x}] \downarrow_2 \tag{5.8}$$

and

$$[\ell * x(\cdot + 1)] \downarrow_2 + (\epsilon_k - 1) \cdot [\tau * \widetilde{x}(\cdot + 1)] \downarrow_2,$$

where the Z-transform of ℓ and τ are $a(z)$ and $b(z)$ divided by their greatest common divisor $c(z)$ respectively. It is worth mentioning that we model the blurring procedure from HR to LR by a first-order Taylor approximation. But our reasoning could easily be extended to a modeling by higher-order Taylor approximations, leading to the same conclusion.

5.4 Reconstruction based on PR filter banks

5.4.1 Introduction to PR filter banks

Before presenting our algorithm, we first give a brief introduction to 2-channel PR (*perfect reconstruction*) filter banks, also called wavelet filter banks (see [100] for more details). A two-channel filter bank consists of two parts: an analysis filter bank and a synthesis filter bank. In our case, the signal $\widetilde{x}$ is first convolved with a low-pass filter ℓ and a high-pass filter h and then subsampled by 2. In other words, we analyze the signal by an analysis filter bank. Then a reconstructed signal $\widehat{x}$ is obtained by upsampling the signal by zero interpolation and then filtering it with a dual low-pass filter g and a dual high-pass filter q. In other words, we reconstruct

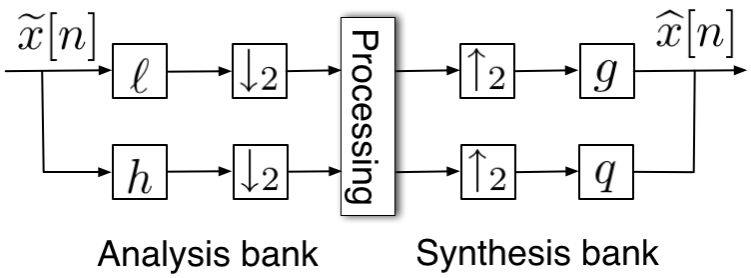


Figure 5.2: The two-channel filter bank.

the signal by synthesizing the output from the analysis bank with a synthesis filter bank. See Fig. 5.2 for an illustration.

Such a filter bank is called a PR filter bank if $\widehat{x} = \widetilde{x}$ for any input $\widetilde{x}$. The question then is, what makes $\{\ell, h, q, g\}$ a PR filter bank. It is easy to see that the process illustrated in Fig. 5.2 could be expressed using Z-transforms as follows:

$$\widehat{x}(z) = \frac{1}{2}(\widetilde{x}(z) + \widetilde{x}(-z)) \begin{pmatrix} \ell(z) & h(z) \\ \ell(-z) & h(-z) \end{pmatrix} \begin{pmatrix} g(z) \\ q(z) \end{pmatrix}.$$

Thus the sufficient and necessary condition for $\widehat{x}(z) = \widetilde{x}(z)$ is

$$\begin{pmatrix} 2z^m \\ 0 \end{pmatrix} = \begin{pmatrix} \ell(z) & h(z) \\ \ell(-z) & h(-z) \end{pmatrix} \begin{pmatrix} g(z) \\ q(z) \end{pmatrix},$$

or equivalently,

$$\begin{pmatrix} g(z) \\ q(z) \end{pmatrix} = \begin{pmatrix} \ell(z) & h(z) \\ \ell(-z) & h(-z) \end{pmatrix}^{-1} \begin{pmatrix} 2z^m \\ 0 \end{pmatrix}.$$

Then in order to obtain FIR g and q, we require that

$$\det(H(z)) = \det(\begin{pmatrix} \ell(z) & h(z) \\ \ell(-z) & h(-z) \end{pmatrix}) = cz^n.$$

In summary, the synthesis filters $\{\ell, h\}$ of a perfect reconstruction filter bank have to satisfy the following condition:

$$\ell(z)h(-z) - \ell(-z)h(z) = z^m \quad \text{for some integer } m, \tag{5.9}$$

and the corresponding synthesis filters amount to

$$g(z) = h(-z); \quad q(z) = -(\ell(-z)).$$

Thus, given any low-pass filter $\ell(z)$, we can find the corresponding high-pass filter $h(z)$ such that we have a PR filter by solving the linear system (5.9).

Example 2.3. For well-known "Harr" wavelet filter bank, the synthesis and analysis filters amount to:

$$\begin{aligned} \ell &= \frac{1}{2}(1,1), \quad h = \frac{1}{2}(1,-1); \\ g &= \frac{1}{2}(1,1), \quad q = \frac{1}{2}(-1,1). \end{aligned}$$

5.4.2 Iterative reconstruction scheme

We have available a number of signals y_k and the corresponding estimates of the optic flow values ϵ_k. We also have estimates of the convolution kernels ℓ and τ. Obviously, it is not wise to directly estimate $\ell * x$ and $\tau * x$ from Equation (5.3). Special attention is necessary here for numerical stability. Fortunately the scheme of PR filter bank provides us with an iterative scheme.

Let ℓ which corresponds to the blurring kernel be the low pass filter of a PR filter bank, then the corresponding high-pass filter h could be computed by solving

(5.9). Note h may be different from τ. Recall that for each LR signal y_k, we have

$$y_k = [\ell * \widetilde{x}] \downarrow_2 + \epsilon_k \cdot * [\tau * \widetilde{x}] \downarrow_2 .$$

Notice that the incompleteness of the original sequence could be overcome by a simple interpolation. Thus $[\ell * \widetilde{x}] \downarrow_2$ amounts to

$$[\ell * \widetilde{x}] \downarrow_2 = y_k - \epsilon_k \cdot * [\tau * \widetilde{x}] \downarrow_2 . \tag{5.10}$$

Notice that the process of a signal $\widetilde{x}$ passing through a PR filter bank as shown in Fig. 5.2 can be expressed as:

$$\widetilde{x} = g * [(\ell * \widetilde{x}) \downarrow_2] \uparrow_2 + q * [(h * \widetilde{x}) \downarrow_2] \uparrow_2 . \tag{5.11}$$

Combining (5.10) and (5.11), we obtain the iterative reconstruction of $\widetilde{x}$ from K LR signals y_k as follows: At step $n+1$

$$\widetilde{x}^{n+1} = q * [(h * \widetilde{x}^n) \downarrow_2] \uparrow_2 + g * \Big(\frac{1}{K} \sum_{k=1}^{K} [y_k - \epsilon_k \cdot * (\tau * \widetilde{x}^n) \downarrow_2] \uparrow_2 \Big). \tag{5.12}$$

Theorem 1. *The iteration of Equation (5.12) converges to the true value x under the condition that*

$$\|g * \tau\| \leq \frac{1}{2}. \tag{5.13}$$

Proof of Theorem 1. As in [89] for simplicity we omit the down-sampling and upsampling process, as well as the fusion process. That is, we write

$$y = \ell * \tilde{x} + \epsilon \cdot * (\tau * \tilde{x})$$

Then the iteration is

$$\tilde{x}^{(n+1)} = g * (y - \epsilon \cdot *(\tau * \tilde{x}^{(n)})) + q * (\tau * \tilde{x}^{(n)}). \tag{5.14}$$

Subtracting $\tilde{x}$ on both sides of (5.14) yields

$$\begin{aligned} \tilde{x}^{(n+1)} - \tilde{x} &= g * (y - \epsilon \cdot *(\tau * \tilde{x}^{(n)})) + q * (\tau * \tilde{x}^{(n)}) \\ &= g * (\ell * \tilde{x} + \epsilon \cdot *(\tau * (\tilde{x} - \tilde{x}^{(n)}))) + q * (\tau * \tilde{x}^{(n)}) - \tilde{x}. \end{aligned}$$

Recall that we have

$$\tilde{x} = g * (\ell * \tilde{x}) + q * (h * \tilde{x}).$$

Then

$$\begin{aligned} \tilde{x}^{(n+1)} - \tilde{x} &= -g * (\epsilon \cdot *((\tau * (\tilde{x}^{(n)} - \tilde{x})) + q * h * (\tilde{x}^{(n)} - \tilde{x}) \\ &= (g * (-\epsilon \cdot *\tau *) + q * h *)(\tilde{x}^{(n)} - \tilde{x}). \end{aligned}$$

Let A denote the operator which represents $g * (-\epsilon \cdot *\tau * + q * h *)$. Then the above equation can be rewritten as

$$\tilde{x}^{(n+1)} - \tilde{x} = A(\tilde{x}^{(n)} - x).$$

Since we have (See [100] for more details):

$$\|q * h\| = \|g * \ell\| = \frac{1}{2},$$

from the fact, $\|\epsilon\|_\infty < 1$, we obtains

$$\|A\| \leq \|g * \tau\| + \|q * h\| < \|g * \tau\| + \frac{1}{2}.$$

Thus $\|g * \tau\| \leq \frac{1}{2}$ is sufficient for the convergence of the iteration. □

5.4.3 Relation to other back-projection methods

Applying (5.11), we can rewrite (5.12) in the form

$$\begin{aligned}\widetilde{x}^{n+1} &= (\widetilde{x}^n - g * [\ell * (\widetilde{x}^n) \downarrow_2] \uparrow_2 + g * \Big(\frac{1}{K}\sum_{k=1}^{K}[y_k - \epsilon_k \cdot *(\tau * \widetilde{x}^n) \downarrow_2] \uparrow_2 \Big) \\ &= \widetilde{x}^n + g * \Big(\frac{1}{K}\sum_{k=1}^{K}[y_k - (\ell * \widetilde{x}^n + \epsilon_k \cdot *(\tau * \widetilde{x}^n)) \downarrow_2] \uparrow_2 \Big).\end{aligned}$$

It can be seen that the iteration scheme presented here falls in the class of back-projection methods. But it has advantages over the usual back-projection iterations. Consider the well-known method by Irani and Peleg [89]. Its iteration can be described as:

$$x^{n+1} = x^n + \frac{1}{K}\sum_{k=1}^{K} T_k^{-1}\Big(((y_k - [\ell * T_k(x^n)] \downarrow_2) \uparrow_2) * p\Big), \tag{5.15}$$

where T_k is the geometric transform between y_k and $\widetilde{x}$, and the high-pass filter p is the de-blurring kernel. Notice that the two methods differ in the de-blurring kernel: one uses g with $g(z) = h(-z)$ defined in (5.9); the other is p in (5.15), the approximate inverse filter of ℓ.

The requirement on p in (5.15) is

$$||\delta - \ell * p|| < 1, \tag{5.16}$$

where δ is the ideal unit impulse response filter. In other words, p should be a good approximation for the inverse of ℓ. In comparison, g in our iteration only needs to be a companion filter for the smooth filter ℓ with sufficient decay, such that condition (5.13) holds. This difference makes g more desirable than p. Let's investigate this in more detail in the following paragraph.

The noise will be propagated exponentially as $O(\|p\|^n)$ in (5.15) and as $O(\|g\|^n)$ in (5.12). Generally the flexibility of $g(z)$ makes it possible to design a g that has much smaller norm than p. This leads to much better resistance to noise propagation. Here is an example: Consider $\ell = \frac{1}{4}(1, 2, 1)$. Then

$$g = (-1/8, -1/4, 3/4, -1/4, -1/8)$$

is a dual PR filter for ℓ with

$$\ell(z)g(-z) = -1 + 9z^{-2} + 16z^{-3} + 9z^{-4} - z^{-6}.$$

It is easy to check that $\|g\|_2$ is around 0.85. The minimum for the norm of all filters with the same length as g is around 1.1. The corresponding p is

$$p = (\frac{1}{2}, -\frac{2}{3}, \frac{4}{3}, -\frac{2}{3}, \frac{1}{2}).$$

In order to make the norm of p close to the norm of g, a lengthy p with much slower decay is necessary. Such a filter is not desirable since it causes artifacts, like the "ring" effect. This clearly indicates that our iteration scheme is more robust to noise and causes less artifacts in the reconstructed image.

5.5 Robust algorithm on 2D images with de-noising

Next we generalize the algorithm to 2D images. Then we introduce a de-noising process during the iterative reconstruction to suppress the noise in the optical flow estimation. Furthermore, the algorithm is adjusted to handle outliers.

5.5.1 Extension to 2D images with a built-in de-noising process

All the previous analysis can be generalized using the tensor product. By an argument similar as for the 1D case, we approximate the LR image I^{LR} with the HR image I^{HR} as follows:

$$I^{LR} = [(a \otimes a) * I^{HR} + u \cdot *((a \otimes b) * I^{HR}) + v \cdot *((b \otimes a) * I^{HR})] \downarrow_2,$$

where "$\otimes$" is the Kronecker tensor product and (u, v) is the 2D optical flow vector. Then the 2D analysis bank is

$$\begin{array}{ll} \text{Low-pass filter:} & L = \ell \otimes \ell, \\ \text{High-pass filters:} & H_1 = \ell \otimes h, H_2 = h \otimes \ell, H_3 = h \otimes h \end{array},$$

and the 2D synthesis filter bank is

$$\begin{array}{ll} \text{Low-pass filter:} & G = g \otimes g, \\ \text{High-pass filters:} & G_1 = g \otimes q, G_2 = q \otimes g, G_3 = q \otimes q. \end{array}$$

It is easy to verify that the 2D filter bank defined above is a perfect reconstruction filter bank with the analysis filter bank $\{L, H_i\}$ and the reconstruction filter bank $\{G, Q_i\}$. Then generalizing (5.12), the iterative equation for the reconstruction of the HR image $\widetilde{I}$ from LR images I_k^{LR} amounts to

$$\begin{aligned} \widetilde{I}^{n+1} &= \textstyle\sum_{i=1}^{3} Q_i * [(H_i * \widetilde{I}^{(n)}) \downarrow_2] \uparrow_2 \\ &+ G * \tfrac{1}{K}\Big(\textstyle\sum_k^K [\widetilde{I}_k^{LR} - u \cdot *((\ell \otimes \tau) * \widetilde{I}^n) \downarrow_2 - v \cdot *((\tau \otimes \ell) * \widetilde{I}^{(n)}) \downarrow_2] \uparrow_2 \Big) \end{aligned}$$

Recall that here $\widetilde{I}$ is the blurred version of the true I with $\widetilde{I} = (c \otimes c) * I$.

There always is noise in the estimated flow u, v. However, the deconvolution operator could make the HR image reconstruction very sensitive to such noise. It is

known that the noise variance of the solution will have hyperbolic growth when the blurring low-pass filter has zeros in the high frequencies. Thus, de-noising is necessary in order to suppress the error propagation during the iterative reconstruction.

To suppress the noise, we introduce a wavelet de-noising scheme which subtracts some high-frequency components from $\widetilde{I}^n$. Briefly, we first do a wavelet decomposition of the high-pass response, then apply a shrinkage of wavelet coefficients to the decomposition, and then reassemble the signal.

Our iteration scheme with built-in de-noising operator amounts to

$$\begin{aligned}\widetilde{I}^{n+1} &= \textstyle\sum_{i=1}^{3} Q_i * [\Psi(H_i * \widetilde{I}^{(n)}) \downarrow_2] \uparrow_2 \\ &+ G * \tfrac{1}{K}\Big(\textstyle\sum_k^K [\widetilde{I}_k^{LR} - u \cdot *((\ell \otimes \tau) * \widetilde{I}^n) \downarrow_2 - v \cdot *((\tau \otimes \ell) * I^{(n)}) \downarrow_2] \uparrow_2 \Big).\end{aligned}$$

The de-noising operator Ψ defined in the equation above is

$$\Psi(H_i * \widetilde{I}^n) = G * [(L * (H_i * \widetilde{I}^n)) \downarrow_2] \uparrow_2 + \sum_{i=1}^{3} [Q_i * \Big(\Gamma[H_i * (H_i * \widetilde{I}^n)]\Big) \downarrow_2] \uparrow_2,$$

where Γ is the shrinkage operator.

5.5.2 Shrinkage operator and robust regression

The basic idea of wavelet de-noising is to reduce the noise by shrinking the wavelet coefficients where typically most noise exist. Here we take a hybrid shrinkage approach. The hybrid shrinkage operator Γ is defined as:

$$\Gamma(\nu) = \begin{cases} 0 & |\nu| \le \mu_1; \\ sign(\nu)\mu_2 \frac{|\nu|-\mu_1}{\mu_2-\mu_1} & \mu_1 < \nu \le \mu_2; \\ \nu & \text{Otherwise.} \end{cases} \tag{5.17}$$

Such a shrinkage offers both benefits from hard shrinkage (uniformly small risk) and soft shrinkage (overall small risk). The reasoning as follows: The optical flow we adopt here is a parametric model. Therefore the noise introduced by the optical flow is not related to the local intensity variation. Thus, a hard shrinkage with appropriate threshold μ_1 could effectively remove such noise. Moreover, noise in illumination which is related to the intensity of the pixels should be removed adaptively. Therefore, a soft-shrinkage is needed in some range. Finally, in order to keep the sharp edges, we keep the large intensity variations. Occasional outliers will be handled by the median operator when fusing multiple frame reconstruction.

In summary, our algorithm is as follows: Given an initial HR image $I^{(0)}$, for each low-resolution I_k, we have

$$\begin{aligned} I^{(n+1)} &= G * median_k\{[I^{LR} - u_k \cdot *(\ell \otimes \gamma * I^{(n)}) \downarrow_2 \\ &- v_k \cdot *(\gamma \otimes \ell * I^{(n)}) \downarrow_2] \uparrow_2\} \\ &+ \left(\sum_{i=1}^{3} Q_i * [\Phi(H_i * I^{(n)}) \downarrow_2] \uparrow_2 \right). \end{aligned} \tag{5.18}$$

The algorithm above could be easily adapted to different blur filters. We only need to adjust the dual filters G, Q_i to make a new perfect reconstruction filter bank. Also, here we only considered a doubling of the image resolution. But any other resolution increase could be achieved by changing the 2-channel perfect reconstruction filter bank to an M-channel perfect reconstruction filter bank.

5.5.3 Relation to regularization methods

One popular de-noising technique used for robust reconstruction is regularization ([101]). Recall that back-projection methods basically find $\widetilde{x}$ by minimizing

$\sum_k^K \|y_k - \widetilde{y}_k(\widetilde{x})\|_2^2$, where $\widetilde{y}_k(\widetilde{x})$ is the LR signals derived from our estimated $\widetilde{x}$. Such a least squares estimation problem usually is ill-conditioned. One way to increase the stability is to enforce a regularization term and solve:

$$\min_{\widetilde{x}} \sum_k^K \|y_k - \widetilde{y}_k(\widetilde{x})\|_2^2 + \alpha \|\Phi(\widetilde{x})\|,$$

where Φ is some regularization function and α is some pre-defined smoothing factor. If the regularization is a least squares problem, we call it a Tikhonov-type regularization. The advantage is its simplicity and efficiency, the disadvantage is its relatively poor performance. A nonlinear diffusion regularization, like Total Variation regularization usually performs better, but is computationally expensive.

Wavelet de-noising is closely related to nonlinear diffusion regularization. [102] discussed the relationship of wavelet de-noising and Total Variation regularization for two simple cases. More specifically, consider a wavelet de-noising scheme based on Haar wavelet ($\ell = \frac{1}{2}(1,1), h = \frac{1}{2}(1,-1)$). Then for the case of the shrinkage operator in wavelet de-noising being a *soft thresholding* operator defined as (5.19), it was shown that such a wavelet de-noising process is equivalent to Total Variation based nonlinear diffusion ($\Phi(\widetilde{x}) = \|\widetilde{x}\|_1$) for a two-pixel signal.

$$\Gamma(\mu) = \begin{cases} \mu - \tau sgn(\mu) & \text{if} \quad (|\mu| \geq \tau); \\ 0 & \text{Otherwise} \end{cases} \tag{5.19}$$

Although [102] only showed the equivalence between Total Variation regularization and a simplistic wavelet de-noising scheme for a signal with 2 pixels, these results still demonstrate that the wavelet de-noising process in our reconstruction is comparable to some nonlinear diffusion regularization schemes in its ability to

suppress the error propagation. However, it doesn't have the computational burden of most nonlinear diffusion regularizations, since it only needs a linear wavelet decomposition over one level. In comparison nonlinear regularizations need to solve a nonlinear optimization.

5.6 Flow estimation for super-resolution

5.6.1 Basic notations

We consider here the planar motion model. In other words, we assume that the underlying 3D structures of the interesting image regions are planar surfaces. Let $I_0, I_1, I_2, \cdots, I_K$ be the image frames in the sequence. Fix frame I_0 as the reference image. We need to estimate the homographies between the reference frame I_0 and the frames I_k. There is the following constraint on the planar homography P_k from reference frame I_0 to frame I_k:

$$P_k = R_k + \vec{v}_k \vec{n}^t, \tag{5.20}$$

where R_k is a rotation matrix, $\vec{v}_k$ is the translation and $\vec{n}$ is the normal of the plane. For two frames close by, the optical flow $\vec{u}$ at a point $\vec{r}$ is constrained by the brightness consistency constraint

$$(\frac{dI}{d\vec{r}})^t \vec{u}(\vec{r}) = -\frac{dI}{dt}. \tag{5.21}$$

Let $\vec{p}_k = vec[P_k] = (p_{1_k}, p_{2_k}, \cdots, p_{9_k})^t$ be the vectorized version of the homography P_k. Assume I_k very close to I_0, then under the small motion assumption, the

brightness consistency constraint becomes

$$(\frac{dI}{d\vec{r}})^t(\vec{p}_k(\vec{r}) - \vec{r}) = \frac{dI}{dt}, \tag{5.22}$$

with

$$\vec{p}_k(\vec{r}) = \begin{pmatrix} \frac{p_1x+p_2y+p_3}{p_7x+p_8y+p_9} \\ \\ \frac{p_4x+p_5y+p_6}{p_7x+p_8y+p_9} \end{pmatrix}.$$

$\vec{p}_k(\vec{r})$ is a 2D linear rational polynomial. Thus multiplying the denominator of $\vec{p}_k(\vec{r})$ on both sides of (5.22) yields a linear homogeneous equation system on $\vec{p}_k$:

$$A_k\vec{p}_k = 0.$$

5.6.2 Multi-frame homography estimation

We need to simultaneously estimate all homographies between the reference image I_0 and the frames I_k. The P_ks are not independent. They share the same plane normal. That is,

$$P_k = R_k + \vec{v}_k \cdot \vec{n}^t, \quad \text{for} \quad k = 1, \cdots, K.$$

In order to improve the estimation of the P_ks, this constraint on the surface normal has to be incorporated into a batch algorithm to Furthermore we need to deal with frames I_k with large displacement to the reference frame I_0.

Here we take an iterative approach to estimate the homographies P_k. Suppose that at the jth step we are given the approximate solution $P_k^j = R_k^j + v_k^j \cdot (n^j)^t$ for the true homography P_k. Then Equation (5.22) could be applied to P_k as follows:

$$(\frac{dI}{d\vec{r}})^t(\vec{p}_k(\vec{r}) - \vec{p}_k^j(\vec{r})) = I_k(\vec{r}) - I_0(p_k^j(\vec{r})).$$

In other words, we apply the differential brightness consistency constraint between the frame $I_0(p_k^j(\vec{r}))$ (the image obtained by warping I_0 from the homography p_k^j) and $I_k(\vec{r})$. The differential motion is due to the the difference between the actual homography and its estimation in the current stage.

We write these linear equations on $\vec{p}_k$ as

$$A_k(P_k^j)\vec{p}_k = 0.$$

Then the minimization across all homographies P_k could be written as

$$\min_{R_k,\vec{n},v_k} \sum_k \|A_k(P_k^j)vec[R_k + \vec{v}_k\vec{n}^t]\|^2 \tag{5.23}$$

subject to the constraints that the R_ks are rotation matrices and $\|\vec{n}\| = 1$. This is a constrained minimization, bilinear in $R_k, \vec{v}_k$ and $\vec{n}$. In the remaining of this section we show how to robustly solve the minimization (5.23) using an alternative two-steps optimization.

Given P_k^j at step j, we compute $P_k^{j+1} = R^{j+1} + \vec{v}_k^{j+1}(\vec{n}^{j+1})^t$ at step $j+1$.

Given $P_k^j = R_k^j + \vec{v}_k^j(\vec{n}^j)^t$, we first update $\vec{n}^{j+1}$. This minimization of (5.23) is just a regular least squares minimization over the sphere of $\vec{n}^{j+1}$ and can be written as:

$$\min_{\vec{n}^{j+1}} \sum_k \|A_k(P_k^j)vec[R_k^j + \vec{v}_k^j(\vec{n}^{j+1})^t]\|^2 \tag{5.24}$$

subject to $\|\vec{n}^{j+1}\| = 1$. The minimization of (5.24) could easily be solved by SVD decomposition. The algorithm is as follows: Given A and $\vec{b}$, the following procedure computes a vector n such that $\|A\vec{n} - \vec{b}\|_2$ is minimum, subject to the constraint $\|\vec{n}\| = 1$. Compute the SVD $A = U\Sigma V^t$ and save

$$V = \{v_1, v_2, \cdots, v_n\}, b = U^t b, \Sigma = diag(\sigma_i).$$

Find λ_* such that

$$\sum_i (\frac{\sigma_i b_i}{\sigma_i^2 + \lambda_*})^2 = 1.$$

Then

$$\vec{n} = \sum_i (\frac{\sigma_i b_i}{\sigma^2 + \lambda_*}) v_i.$$

Then given $P_k^j, \vec{n}^{j+1}$, we estimate $\{R_k^{j+1}, v_k^{j+1}\}$ for $k = 1, \cdots, K$. We need to solve the following minimization: For each k

$$\min_{R_k^{j+1}, v_k^{j+1}} \|A_k(P_k^j) vec(R_k^{j+1} + \vec{v}_k^{j+1}(\vec{n}^{i+1})^t)\|^2, \quad \text{subject to} \quad (R_k^{j+1})^t R_k^{j+1} = I_3. \tag{5.25}$$

This is not a trivial task. Here we present a fast linear method to obtain an approximate solution. Let R_k^{j+1} be approximated by $(I + [\omega]_\times) R_k^j$, where $[\omega]_\times$ is a skew-symmetric matrix from rotation vector ω. Using the observation that $P_k^{j+1} = R_k^{j+1} + \vec{v}_k^{j+1}(\vec{n}^{j+1})^t$ we perform the following decomposition

$$P_k^{j+1} = (I + [\omega_k^j]_\times + \Delta v_k^j \cdot (R_k^j n^{j+1})^t) P_k^j,$$

with

$$v_k^{j+1} = (I + [\omega]_x) v_k^j + (1 + (R_k^j n^{j+1})^t v_k^j) \Delta v_k^j. \tag{5.26}$$

Then the the minimization (5.25) is simplified to a standard least squares minimization on ω_k^i and Δv_k^i:

$$\min_{\omega_k^j, \Delta v_k^j} \|A_k(P_k^j) vec[(I + [\omega]_\times + \Delta v_k^j \cdot (R_k^j \vec{n}^{j+1})^t) P_k^j]\|^2. \tag{5.27}$$

Thus R_k^{j+1} and v_k^{j+1} could be derived from Δv_k^j and ω_k^i after solving the least squares minimization of (5.27).

We adopt the procedure in [103] to compute initial values for the homographies P_k^0s from multiple relative motions P_{k_1,k_2} between frames I_{k_1} and I_{k_2} related by small displacement. Briefly, the P_k^0s are determined by an overdetermined linear system:

$$P_{k_1,k_2}P_{k_1}^0 - P_{k_2}^0 = 0.$$

See [103] for more details. The iteration algorithm is as follows: Given $P_k^j = R_k^j + \vec{v}_k^j(\vec{n}^j)^t$ at Step j,

1. The minimization (5.24) is solved by SVD to obtain $\vec{n}^{j+1}$.

2. The minimization (5.27) is solved by least squares to obtain ω_k^j, Δ_k^j.

3. $\vec{v}_k^{j+1}$ is obtained through (5.26) and the rotation matrix is obtained as:

$$R_k^{j+1} = I + [\omega_k^j]_\times + (1 - (1 - \|\omega\|^2)^{\frac{1}{2}})([\omega_k^j]_\times)^2.$$

The iteration is terminated after the p_k^{j+1}s are close enough to the p_k^js.

At a quick glance, it seems that decomposing P_k is an overkill since we don't need motion and structure. But actually it doesn't make much difference. It is known that the decomposition of $P = R + \vec{t} \cdot \vec{n}^t$ is unique up to two solutions. By enforcing the consistency between the P_k, the decomposition becomes unique, and the decomposition is not difficult to obtain.

5.7 Experiments and conclusion

We compare our algorithm's high-resolution reconstruction to standard methods using both simulated and real data.

5.7.1 Simulated data

We simulated 4 low-resolution images (16×16) from a high resolution image by shifting, blurring and downsampling. The blurring filter is

$$\ell = \frac{1}{16} \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}.$$

Three kinds of noise were simulated:

1. Error in motion estimation. It is modeled by local Gaussian white noise with parameter σ. The local covariance matrix is due to the magnitudes of the image gradients.

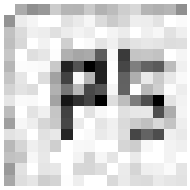
2. Noise in pixel formation. We added a Gaussian white noise with parameter γ to the pixel values.

3. Error in PSF modeling. We also checked how error in the PSF modeling influences the performance. The approximated PSF $\hat{\ell}$ used in the reconstruction was

$$\hat{\ell} = \frac{1}{16} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 8 & 1 \\ 1 & 1 & 1 \end{pmatrix}. \tag{5.28}$$

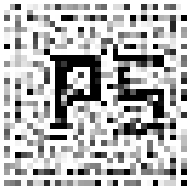
We compared our wavelet-based method to the popular "POCS" back-projection method ([104]) enforced by Tikhonov regularization (See Fig. 5.3). It may be possible that another scheme, namely Total Variation regularization would give a bit better results. However, this would require solving a nonlinear minimization over

(a) Original image



(b) Noisy LR image

(c) Wavelet method



(d) Tikhonov regularization

Figure 5.3: The HR images (c) and (d) are reconstructed from four LR images by five iterations. (c) is reconstructed by our method. (d) is reconstructed by the back-projection method with Tikhonov regularization. The motion noise is local Gaussian noise with $\sigma = 0.2$. The image formation noise is Gaussian noise with $\gamma = 0.01$. The approximation $\hat{\ell}$ in (5.28) is used in the reconstruction instead of the true PSF ℓ.

each iterative step during the reconstruction, which is computationally expensive. In our implementation, the regularization term is the 2-norm of the Laplacian smoothness constraint with parameter $\alpha = 1$.

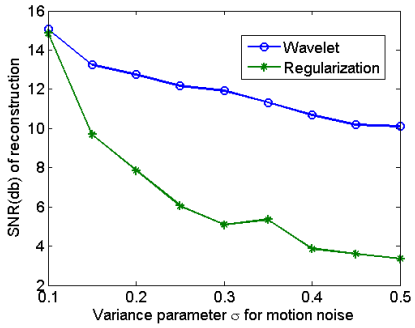
Fig. 5.4 demonstrates how well the wavelet-based method performs for various noise settings. Performance is measured by the SNR (Signal-to-Noise ratio) of the reconstructed image to the true image, which is defined as:

$$SNR = 20 \log_{10} \frac{\|x\|_2}{\|x - \hat{x}\|_2},$$

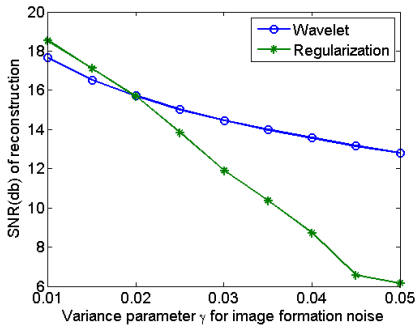
where $\hat{x}$ is the estimate for the true image x. Fig. 5.4 clearly indicates the advantage of our wavelet-based method in suppressing the noise. Especially when noise is large, the boost in performance is significant.

5.7.2 Real data

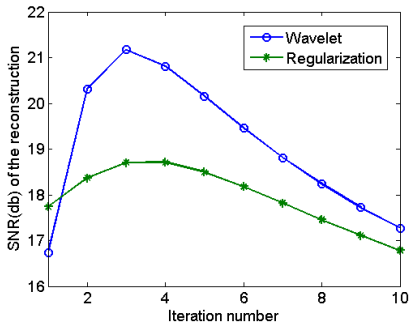
We used an indoor sequence depicting 13 image frames of a paper box (Fig. 5.5). An interesting planar region was chosen manually. Fig. 5.6 and Fig. 5.7 show a comparison of the results by four different methods for different regions. Here the reconstructed HR images double the resolution of the LR images. The HR image in Fig. 5.6-5.7(a) were obtained by cubic interpolation from a single LR image. In Fig. 5.6-5.7(b) we used the POCS method, where the flow field is estimated by an affine motion model. Fig. 5.6-5.7(c) show the results from our reconstruction scheme. The difference can be visually evaluated. Clearly, there is large improvement from (b) to (c) in Fig. 5.6 and Fig. 5.7. The letters in Fig. 5.6(c) and Fig. 5.7(c) are the clearest, and there are minimal artifacts around the edges.



(a) Comparison for motion noise



(b) Comparison for image formation noise

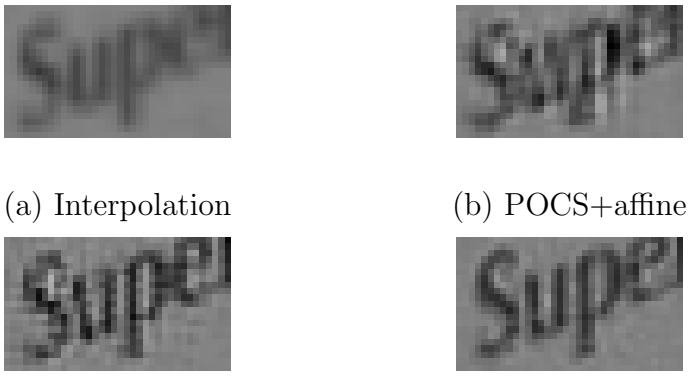


(c) Comparison for PSF error

Figure 5.4: Comparison between the two methods for various amounts of motion noise and image formation noise. The reconstructed image is obtained by 5 iterations. The x-axis denotes the variance of the noise, the y-axis denotes the SNR of the reconstruction.

(a) Reference frame (b) LR planar image region

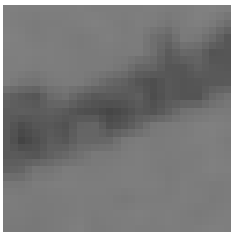
Figure 5.5: Reference image frame of first indoor video and its selected region.



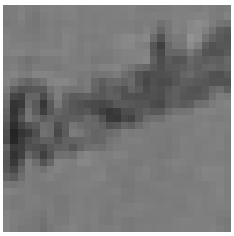
(a) Interpolation (b) POCS+affine

(c) POCS+Homography (d) Wavelet+Homography

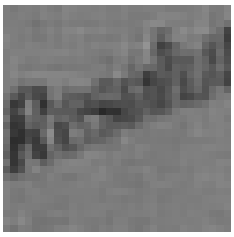
Figure 5.6: Comparison of one reconstructed HR region for various methods.



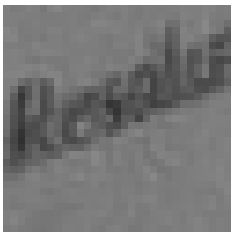
(a) Interpolation



(b) POCS + Affine



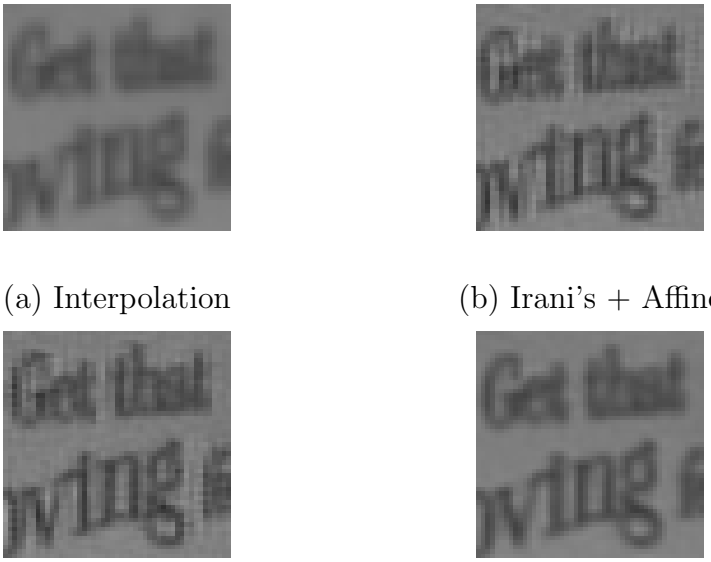
(c) POCS + Homography



(d) Wavelet + Homography

Figure 5.7: Comparison of another reconstructed HR region from Fig. 5.5 for various methods.

Figure 5.8: Reference frame from second indoor video.



(a) Interpolation (b) Irani's + Affine

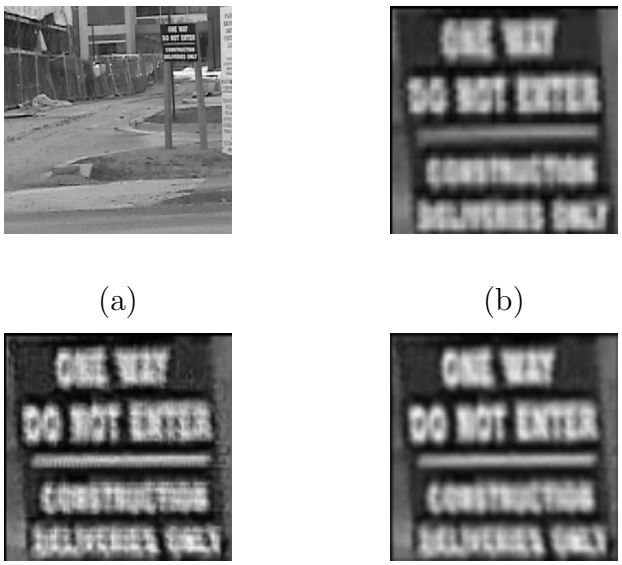
(c) Irani's + Homography (d) Wavelet + Homography

Figure 5.9: Comparison of one reconstructed HR region from Fig. 5.8 for various methods.

A second indoor sequence depicting a box wrapped in newspaper (Fig. 5.8) is tested. This time, we compared our method against Irani's method [89]. See Fig. 5.9 for a visual comparison. The same conclusion holds as for the previous experiment; both our homography-based motion estimation and our reconstruction method lead to improved results.

We also used an outdoor sequence of 11 frames with some warning sign in the scene. We compared the reconstructions from different methods for a manually selected region. Since the regions are too small to provide enough information for getting estimates from the homography flow model, here instead we use an affine flow model. See Fig. 5.10 for a visual comparison.

In summary, we have presented a theoretical analysis and a new algorithm for super-resolution problem based on wavelet theory. It has been demonstrated both



(a) (b)

(c) (d)

Figure 5.10: (a) The key frame in the video. (b) The reconstruction from the interpolation. (c) The reconstruction from Irani's method using affine flow. (d) The reconstructed image from the wavelet method with de-noising using affine flow.

in theory and experiments that the proposed method in this chapter is very robust to noise without sacrificing efficiency. The reconstruction scheme allows for super-resolution reconstruction from general video sequences, even when the estimated optical flow is very noisy.

Chapter 6

Conclusion and future work

This thesis proposes a framework for solving the general structure from motion problem using feed-back loops. The major reason for the scene recovery being difficult is that structure recovery is closely related to scene segmentation. On one hand, structure information is needed to segment images because large spatial regions are needed for good segmentation; on the other hand, the structure can not be obtained accurately without scene segmentation. This chicken-and-egg situation can not be resolved with traditional modular approaches. We propose to overcome this chicken-and-egg dilemma by introducing a feed-back loop approach. Segmentation and scene recovery interact with each other in our approach. Structure estimation and scene segmentation are improved gradually over stages. Our first implementation demonstrates the potential of such an approach for solving the general structure from motion problem.

The shape constraint we presented for linking multiple frames makes possible a robust method for estimating accurate camera motion. Accurate camera motion plays an important role in our feed-back framework. It leads to better scene recovery and better scene segmentation, but also makes it possible to fuse multiple well-developed techniques (stereo matching, occlusion detection) into the structure from motion process.

However, the approach we proposed in this thesis, is just the first step towards a complete architecture for solving the general structure from motion problem. There are several modules, which need to be developed in future work. The first is a module that uses image motion or matching over three or more images to detect occlusions and ordinal depth information, and uses this information to detect and locate independently moving objects in a dynamic environment. The second is a module that refines the scene structure using a more sophisticated surface representation such as a PDE-based implicit surface representation, using as input a fairly good piece-wise planar surface model and an accurate camera motion estimate. Such refinement appears promising, since with some preliminary models of the scene, we can employ larger spatial areas and develop global spatial constraints. Thus better matching can be achieved and more complicated structure models can be implemented.

The MFS (multifractal spectrum) proposed in Chapter 4 serves well the purpose of matching planar patches over large displacements. The advantage of our texture descriptor over existing texture signatures lies in its invariance to environmental changes. The invariance is justified not only from mathematical derivation, but also from our experiments on textures in a real settings. Our new MFS texture descriptor combines global statistics and local image features and results in a very low feature dimension.

Furthermore, the MFS is useful for other application sectors. We applied our new texture descriptor for texture retrieval and classification. The experiments show that the performance of our texture descriptor in texture classification is close to

the top methods, despite having a much smaller feature dimension. All these clearly indicate that our texture descriptor does capture some of the essential structure of natural textures.

In future research, it will be very interesting to investigate how to incorporate the MFS idea into the segmentation process. The goal would be to find a computational process for obtaining a cluster of MFS vectors with every element corresponding to a segmented texture region. Such a representation plus 3D information from ego-motion could possibly lead to a very good framework which provides sufficient information for visual homing and is robust in an unconstrained environment.

A robust solution to the general structure from motion problem benefits visual navigation, but it also constitutes a necessary module for many interesting applications. We applied our algorithm of combining flow fields over multiple fields to provide better alignment of images in a super-resolution imaging system. Moreover, we developed a new wavelet-based reconstruction scheme for reconstructing a high-resolution image from multiple aligned low-resolution images. Our reconstruction scheme is based on an analysis of image formation from the view of filter bank theory. So some unnecessary de-blurring operator is avoided, which improves the robustness of the algorithm. Furthermore, our wavelet-based algorithm allows a wavelet de-noising operator to be built in without much computational expense. It is known that wavelet de-noising is essentially a form of anisotropic diffusion. Therefore, our reconstruction algorithm not only is more robust to noise (reduces the blurring process) but also avoids the smoothing effect introduced by regular

interpolation methods. Our wavelet-based approach could be extended to other applications in image de-blurring, such as motion de-blurring.

BIBLIOGRAPHY

[1] G. Mitchison and S. McKee, "Mechanisms underlying the anisotropy of stereoscopic tilt perception", *Vision Research*, vol. 30, pp. 1781–1791, 1990.

[2] L. Zelnik-Manor and M. Irani, "Multi-Frame estimation of planar motion", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1105–1114, 2000.

[3] S. Lazebnik, C. Schmid and J. Ponce, "A sparse texture representation using affine-Invariant regions", *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2005.

[4] M. Varma and A. Zisserman, "Classifying images of materials: Achieving viewpoint and illumination independence", in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, volume 3, pp. 255–271, 2003.

[5] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.

[6] O. D. Faugeras, *Three-Dimensional Computer Vision*, MIT Press, Cambridge, 1992.

[7] O. D. Faugeras and Q-T. Luong, *The Geometry of Multiple Images*, MIT Press, Cambridge, 2001.

[8] Y. Ma, S. Soatto, J. Kosecka and S. S. Sastry, *An Invitation to 3-D Vision*, Springer, Berlin, 2004.

[9] R. Y. Tsai and T. S. Huang, "Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 6, pp. 13–27, 1984.

[10] M. E. Spetsakis and Y. Aloimonos, "Structure from motion using line correspondences", *Int'l J. Computer Vision*, vol. 4, pp. 171–183, 1990.

[11] B. K. P. Horn, *Robot Vision*, MacGraw Hill, New York, 1986.

[12] S. J. Maybank, *Theory of Reconstruction from Image Motion*, Springer, Berlin, 1993.

[13] K. Kanatani, "3-D interpretation of optical flow by renormalization", *Int'l J. Computer Vision*, vol. 11, pp. 267–282, 1993.

[14] D. J. Heeger and A. D. Jepson, "A subspace method for recovering rigid motion I: Algorithm and implementation", *Int'l J. Computer Vision*, vol. 7, pp. 95–117, 1992.

[15] P. Allen, B. Yoshimi and A. Timcenko, "Real-time visual servoing", in *International Conference on Robotics and Automation*, pp. 851–856, 1991.

[16] C. E. Smith, S. A. Brandt and N. P. Papanikolopoulos, "Eye-in-hand robotic tasks in uncalibrated environments", *IEEE Trans. on Robotics and Automation*, vol. 13, n. 6, pp. 903–914, 1997.

[17] W. B. Thompson and T.-C. Pong, "Detecting moving objects", *Int'l J. Computer Vision*, vol. 4, pp. 39–57, 1990.

[18] R. C. Nelson, "Qualitative detection of motion by a moving observer", *Int'l J. Computer Vision*, vol. 7, pp. 33–36, 1991.

[19] K. Daniilidis and H.-H. Nagel, "Analytical results on error sensitivity of motion estimation from two views", *Image and Vision Computing*, vol. 8, pp. 297–303, 1990.

[20] C. Fermuller and Y. Aloimonos, "Observability of 3D Motion", *Int'l J. Computer Vision*, vol. 37, pp. 43–63, 2000.

[21] S. J. Maybank, "Algorithm for analysing optical flow based on the least-squares method", *Image and Vision Computing*, vol. 4, pp. 38–42, 1986.

[22] M. E. Spetsakis and Y. Aloimonos, "A multi-frame approach to visual motion perception", *Int'l J. Computer Vision*, vol. 6, pp. 245–255, 1991.

[23] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization method", *Int'l J. Computer Vision*, vol. 8, pp. 137–154, 1992.

[24] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms", *Int'l J. Computer Vision*, vol. 47, pp. 7–42, 2002.

[25] Y. Aloimonos, "Shape from texture", *Biological Cybernetics*, vol. 58, pp. 345–360, 1998.

[26] R. Bajcsy and L. Lieberman, "Texture gradient as a depth cue", *Computer Graphics and Image Processing*, vol. 5, pp. 52–67, 1976.

[27] J. Garding, "Shape from texture and contour by weak isotropy", *J. of Artificial Intelligence*, vol. 64, pp. 243–297, 1993.

[28] D. C. Knill, "Surface orientation from texture: Ideal observers, generic observers and the information content of texture cues", *Vision Research*, vol. 38, pp. 1655–1682, 1998.

[29] K. Ikeuchi and B. Horn, "Numerical shape from shading and occluding boundaries", *Artificial Intelligence*, vol. 17, pp. 141–184, 1981.

[30] S. K. Nayar, K. Ikeuchi and T. Kanade, "Determining shape and reflectance of hybrid surfaces by photometric sampling", *IEEE Journal of Robotics and Automation*, vol. 6, n. 4, pp. 418–431, 1990.

[31] J. Koenderink, "What does the occluding contour tell us about solid shape", *Perception*, vol. 13, pp. 321–330, 1984.

[32] K. Kutulakos and C. Dyer, "Global surface reconstruction by purposive viewpoint control", *Artificial Intelligence*, vol. 1-2, pp. 147–177, 1995.

[33] J. Koenderink and A. V. Doorn, "Surface perception in picture", *Perception and Psychophysics*, vol. 52, pp. 387–426, 1992.

[34] G. J. Andersen, M. L. Braunstein and A. Saidpour, "The perception of depth and slant from texture in three-dimensional scenes", *Perception*, vol. 27, pp. 2635–2656, 1998.

[35] J. Perrone, "Slant underestimation: A general model", *Perception*, vol. 11, pp. 641–654, 1982.

[36] R. Goutcher and P. Mamassian, "A ground plane preference for stereoscopic slant", in *European Conference on Vision Perception*, 2002.

[37] J. Todd and V. J. Perotti, "The visual perception of surface orientation from optical motion", *Perception and Psychophysics*, vol. 61, pp. 1577–1589, 1999.

[38] C. Fermuller and C. Malm, "Uncertainty in visual processes predicts geometrical optical illusions", *Vision Research*, vol. 44, pp. 727–749, 2004.

[39] C. Fermuller, R. Pless and Y. Aloimonos, "The Ouchi illusion as an artifact of biased flow estimation", *Vision Research*, vol. 40, pp. 77–96, 2000.

[40] H. Ji and C. Fermüller, "Bias in shape estimation", in *Proc. European Conf. Computer Vision*, volume 3, pp. 405–416, 2004.

[41] H. Ji, C. Fermüller and Y. Aloimonos, "Noise causes slant under-estimation in stereo and motion", Vision Research, In Press.

[42] W. Fuller, *Measurement Error Models*, Wiley, New York, 1987.

[43] S. V. Huffel and J. Vandewalle, *The Total Least Squares Problem: Computational Aspects and Analysis*, SIAM, 1991.

[44] G. W. Stewart, "Stochastic pertubation theory", *SIAM Review*, vol. 32, pp. 576–610, 1990.

[45] K. Kanatani, *Statistical Optimization for Geometric Computation: Theory and Practice*, Elsevier, Amsterdam, 1996.

[46] A. Chowdhury and R. Chellappa, "Statistical error propagation in 3d modeling from monocular video", in *IEEE Workshop on Statistical Analysis in Computer Vision*, 2003.

[47] C. Tomasi and J. Zhang, "Is structure from motion worth pursuing", in *Proc. of the Seventh International Symposium on Robotics Research*, pp. 391–400. Springer Verlag, 1995.

[48] R. Cagnello and B. J. Rogers, "Anisotropies in the perception of stereoscopic surfaces: the role of orientation disparity", *Vision Research*, vol. 33, n. 16, pp. 2189–2201, 1993.

[49] C. Ryan and B. Gillam, "Cue conflict and stereoscopic surface slant about horizontal and vertical axes", *Perception*, vol. 23, pp. 645–658, 1994.

[50] L. Ng and V. Solo, "Errors-in-variables modeling in optical flow estimation", *IEEE Transactions on Image Processing*, vol. 10, pp. 1528–1540, 2001.

[51] N. Lydia and S. Victor, "Errors-in-variables modeling in optical flow estimation", *IEEE Trans. on Image Processing*, vol. 10, pp. 1528–1540, 2001.

[52] H. Nagel, "Optical flow estimation and the interaction between measurement errors at adjacent pixel positions", *International Journal of Computer Vision*, vol. 15, pp. 271–288, 1995.

[53] C. Yo and H. R. Wilson, "Moving 2D patterns capture the perceived direction of both lower and higher spatial frequencies", *Vision Research*, vol. 32, 1992.

[54] A. Shashua and M. Werman, "On the trilinear tensor of three perspective views and its underlying geometry", in *Proc. Int'l Conf. Computer Vision*, 1995.

[55] S. Carlson and D. Weinshall, "Dual computation of projective shape and camera positions from multiple images", *Int'l J. Computer Vision*, vol. 27, n. 3, pp. 227–241, 1998.

[56] O.D. Faugeras and T. Papadopoulo, "A nonlinear method for estimating the projective geometry of 3 views", in *Proc. Int'l Conf. Computer Vision*, pp. 477–484, 1998.

[57] B. Triggs, P. McLauchlan, R. Hartley and A. Fitzgibbon, "Bundle adjustment—a modern synthesis", in B. Triggs, A. Zisserman and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*. Springer Verlag, 2000.

[58] J. Oliensis, "A multi-frame structure-from-motion algorithm under perspective projection", *Int'l J. Computer Vision*, vol. 34, n. 2, pp. 163–192, 1999.

[59] R. Vidal and J. Oliensis, "Structure from planar motions with small baselines", in *Proc. European Conf. Computer Vision*, volume 2, pp. 383–398, 2002.

[60] M. J. Black and P. Anandan, "Robust dynamic motion estimation over time", in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 296–302, 1991.

[61] M. J. Black, "Combining intensity and motion for incremental segmentation and tracking over long image sequences", in *Proc. European Conf. Computer Vision*, pp. 485–493, 1991.

[62] D. Forsyth, S. Ioffe and J. Haddon, "Bayesian structure from motion", in *Proc. European Conf. Computer Vision*, pp. 660–665, 1999.

[63] G. Qian and R. Chellappa, "Structure from motion using sequential monte carlo methods", *Int'l J. Computer Vision*, vol. 59, pp. 5–31, 2004.

[64] C. Baillard and A. Zisserman, "Automatic reconstruction of piecewise planar models from multiple views", in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 559–565, 1999.

[65] A. Dick, P. Torr and R. Cipolla, "Automatic 3D modelling of architecture", in *Proc. British Machine Vision Conf.*, pp. 372–381, 2000.

[66] A. Shashua and S. Avidan, "The rank 4 constraint in multiple ($\geq$ 3) view geometry", in *Proc. European Conf. Computer Vision*, 1996.

[67] L. Zelnik-Manor and M. Irani, "Multi-view subspace constraints on homographies", in *Proc. Int'l Conf. Computer Vision*, pp. 710–715, 1999.

[68] T. Brodsky, C. Fermüller and Y. Aloimonos, "Motion segmentation: a synergistic approach", in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, volume 2, pp. 2226–2230, 1999.

[69] H. Ji and C. Fermüller, "A 3D shape constraint on video", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, n. 6, pp. 1018–1023, 2006.

[70] L. Cheong, C. Fermuller and Y. Aloimonos, "Effects of errors in the viewing geometry on shape estimation", *Computer Vision and Image Understanding*, vol. 71, n. 356-372, 1998.

[71] H. Ji and C. Fermüller, "Integration of motion fields through shape", in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, volume 2, pp. 663–669, 2005.

[72] E. Polak, *Optimization: algorithm and consistent approximation*, Spring, Berlin, 1996.

[73] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation", *Int'l J. Computer Vision*, vol. 59, pp. 161–181, 2004.

[74] T. H. Cormen and C. E. Leiserson, *Introduction to Algorithms*, McGraw-Hill, 1990.

[75] I. Weiss, "Geometric invariants and object recognition", *Int' Journal on Computer Vision*, vol. 10, n. 3, pp. 207–231, 1993.

[76] J. L. Mundy and A. Zisserman, *Geometric Invariance in Computer Vision*, MIT Press, 1992.

[77] R. Azencott, J. Ping and L. Younes, "Texture classification using windowed Fourier filters", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, n. 2, pp. 148–153, 1997.

[78] C. Kervrann and F. Heitz, "A Markov random field model-based approach to unsupervised texture segmentation using local and global spatial statistics", *IEEE Trans. Image Process*, vol. 4, n. 6, pp. 856–862, 1995.

[79] A. Laine and J. Fan, "Texture classification by wavelet packet signatures", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, n. 11, pp. 1186–1191, 1993.

[80] A. Teuner, O. Pichler and B. J. Hosticka, "Unsupervised texture segmentation of images using tuned matched Gabor filters", *IEEE Trans Image Process*, vol. 4, n. 6, pp. 863–870, 1995.

[81] F. Espinal, B. D. Jawerth and T. Kubota, "Wavelet-based Fractal Signature Analysis for Automatic Target Recognition", *Optical Engineering*, vol. 37, n. 1, pp. 166–174, 1998.

[82] R. Hartley S. Peleg, J. Naor and D. Avnir, "Multiple resolution texture analysis and classification", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 6, pp. 518–523, 1984.

[83] L. Kam and J. Blanc-Talon, "Are multifractal multipermuted multinomial measures good enough for unsupervised image segmentation", in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, volume 1, pp. 58–63, 2000.

[84] L. M. Kaplan, "Extended fractal analysis for texture classification and segmentation", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, n. 11, pp. 1572–1585, 1999.

[85] A. Conci and L. H. Monteiro, "Multifractal characterization of texture-based segmentation", in *ICIP*, pp. 792–795, 2000.

[86] B. B. Mandelbrot, *The Fractal Geometry of Nature*, CA: Freeman, San Francisco, 1982.

[87] Y. Xu, H. Ji and C. Fermüller, "A projective invariant for textures", in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2006.

[88] K. J. Falconer, *Techniques in Fractal Geometry*, John Wiley, 1997.

[89] M. Irani and F. Peleg, "Motion analysis for image enhancement: Resolution, occlusion and transparency", *Journal of Visual Comm. and Image Repre.*, vol. 4, pp. 324–335, 1993.

[90] M. Eland and A. Feuer, "Restoration of a signal super-resolution image from several blurred, noisy and undersampled measured images", *IEEE Trans. on Image Processing*, pp. 1646–1658, 1997.

[91] B. Bascle, A. Blake and A. Zisserman, "Motion deblurring and super-resolution from an image sequence", in *Proc. European Conf. Computer Vision*, volume 2, pp. 573–582, 1996.

[92] W. Zhao and H. S. Sawhney, "Is super-resolution with optical flow feasible", in *Proc. European Conf. Computer Vision*, pp. 599–613, 2002.

[93] A. Tekalp, M. Ozkan and M. Sezan, "High-resolution image reconstruction from low-resolution image sequences and space-varying image restoration", in *ICASSP*, pp. 169–172, 1992.

[94] S. Farsiu, D. Robinson, M. Elad and P. Milanfar, "Robust shift and add approach to super-resolution", in *SPIE*, 2003.

[95] A. Chambolle, R. Devore, N. Lee and B. Lucier, "Nonlinear wavelet image processing: Variational problems, compression and noise removal through wavelets", *IEEE Trans. Image Processing*, vol. 7, 1998.

[96] R. Chan, T. Chan, L. Shen and Z. Shen, "Wavelet deblurring algorithms for spatially varying blur from high-resolution image reconstruction", *Linear algebra and its applications*, vol. 366, pp. 139–155, 2003.

[97] N. Nguyen and N. P. Milanfar, "A wavelet-based interpolation-restoration method for superresolution", *Circuits, Systems and Signal Processing*, vol. 19, pp. 321–338, 2000.

[98] H. Ji and C. Fermüller, "Wavelet-based super-resolution reconstruction: Theory and Algorithm", in *Proc. European Conf. Computer Vision*, 2006.

[99] S. Baker and T. Kanade, "Limits on super-resolution and how to break them", in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 372–379, 2000.

[100] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, 1999.

[101] J. Weickert, *Anisotropic Diffusion in Image Processing*, Teubner, Stuttgart, 1998.

[102] P. Mrazek, J. Weickert and G. Steidl, "Correspondences between wavelet shrinkage and nonlinear diffusion", in *Proc. Scale-Space Methods in Computer Vision*, pp. 101–116, 2003.

[103] V. M. Govindu, "Combining two-view constraints for motion estimation", in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2001.

[104] C. Youla, "Generalized image restoration by the method of alternating orthogonal projections", *IEEE Trans. Circuits System*, 1978.