National Library of Canada
Acquisitions and Bibliographic Services Branch
395 Wellington Street
Ottawa Ontario
K1A 0N4

Bibliothèque nationale du Canada
Direction des acquisitions et des services bibliographiques
395, rue Wellington
Ottawa (Ontario)
K1A 0N4

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

Motion Analysis from a Sequence of Range Images

Peiying Zhu

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy at

Concordia University

Montréal, Québec, Canada

April 1993

© Peiying Zhu, 1993

National Library of Canada
Bibliotheque nationale du Canada

Acquisitions and Bibliographic Services Branch
Direction des acquisitions et des services bibliographiques

395 Wellington Street
Ottawa, Ontario
K1A 0N4

395 rue Wellington
Ottawa (Ontario)
K1A 0N4

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-84684-4

Canada

ABSTRACT

Motion Analysis from a Sequence of Range Images

Peiying Zhu

The main goal of this work is to demonstrate the feasibility and potential of recovering motion from a sequence of range images as an alternate solution to the complex motion problem. The work presented in this thesis can be divided into two separate parts. The first part describes the long term process, and the second part discusses the short term process.

The major problem for the long term process is to reliably find matching features in two or more successive images. An approach is proposed to establish the best match of point features between successive frames using a Hopfield neural network. A model is developed to convert the correspondence problem to the problem of minimizing an energy function, which occurs at the stable state of a Hopfield neural network. After establishing the feature matching, a δ-bound matching concept is introduced to detect the reliable matching features, therefore increase the accuracy of the estimated motion parameters by removing the effect of mismatching features. In this way, the algorithm is tolerant to noise due to feature detection or occlusion.

For the short term process, the case of a single rigid moving object is first studied. A simple, yet powerful, algorithm is proposed to estimate motion of a single rigid object. The motion problem is modeled as solving a set of linear equations. A weighted least squares technique has been found to provide the best performance among several other versions of least squares techniques. Theoretical analysis on the necessary and sufficients

conditions for the unique interpretation of the motion parameters and on the sensitivity of the estimated motion parameters to noise provides further insight into the behavior of the algorithm.

For more complicated motion such as nonrigid motion, the complete process can be viewed in two separate levels: low and high. In this thesis, attention has been paid to the low level processing. A 3D velocity field has been chosen to be the output of the low level stage. We first develop an algorithm which uniquely estimates 3D velocities of points on smooth surfaces by its first and second order partial derivatives, except at parabolic points. The algorithm is very fast and easy to implement in hardware or software. However, it does not provide reliable estimates of velocities near edge points. Hence we propose another algorithm, which is based on the correlation of the local structure of principal curvatures. The advantage of this correlation approach is that it can estimate velocities of both corner points as well as points on smooth curved surfaces, and vernier velocities of line edge points. The disadvantage is that it is computationally intensive compared with the approach for smooth surfaces. Therefore, we suggest that two algorithms should be combined together to give the best performance.

Many experimental results on both synthetic and real images are presented in this thesis.

ACKNOWLEDGEMENTS

I wish to thank my supervisors, Dr. Tony Kasvand, and Dr. Adam Krzyzak for their guidance, continued encouragement, support and the time to evaluate the many branches of research that appeared. Without them, it would have been impossible for me to pursue this work.

I am grateful to Dr. C. Y. Suen as head of the Centre for Pattern Recognition and Machine Intelligence (CENPARMI) for providing the stimulating environment that allowed this work to progress smoothly.

Dr. J. Domey and Dr. M. Rioux at the Canadian National Research Council in Ottawa were very generous in providing range images, which were essential for the completion of this thesis. F. P. Ferrie, Pierre Tremblay and Gilbert Soucy at the McGill Research Center for Intelligent Machines were most kind in providing additional range images.

I also thank Paul, Bill and Stephen for providing a stable environment loaded with tools and for helping me to tame Unix.

Special thanks go to the South East University in the People's Republic of China, and the Canadian International Development Agency, for providing the opportunity to study in Canada.

I would like to express my sincere gratitude to my external examiner Dr. N. Abdelmalek and to examining committee: Dr. R. M. H. Cheng, Dr. T. D. Bui, Dr. C. Y. Suen, Dr. J. F. Hayes, Dr. E. I. Plotkin, Dr. T. Kasvand and Dr. A. Krzyzak for their useful comments, which improved the quality of the final version of the thesis.

Last but not least, I would like to thank my family for their immense love and their wholehearted support, and to all my friends for their encouragement.

TABLE OF CONTENTS

Table of Contents iv

List of Figures viii

List of Tables xv

1 Introduction **1**

1.1 2D Apparent Motion, 3D Motion, Methodologies and Difficulties 1

1.2 A Triangulating Range Scanner 4

1.3 Range Images vs. Intensity Images 6

1.4 Previous Work 7

1.5 Contributions 9

1.6 Overview of the Study 10

2 Motion Estimation Based on Correspondence **12**

2.1 Introduction 12

2.2 Neural Network 14

2.3 Point Corresponding Using Neural Network 15

2.4 Motion Estimation Using δ-Bound Matching Points 19

2.5 Simulation 22

2.6 Extension of the Algorithm for Plane and Surface Correspondence 28

2.7 Conclusion 28

3 Motion Estimation of a Single Rigid Object **30**

3.1 Introduction 31

3.2 Formulation of Rigid Body Motion 31

3.3 Solution of Rigid Body Motion 35

3.4 Ambiguity of Motion Perception 36

3.4.1 Pure Translation 37

3.4.2 Pure Rotation 38

3.4.3 General Rigid Motion 41

3.5 Error Analysis 42
3.5.1 Gradient Measurement Error 42
3.5.2 Conditioning 45
3.6 Implementation 50
3.6.1 Weighted Least Squares Method 50
3.6.2 Total Least Squares Method 53
3.6.3 Normalizing the Coefficients 55
3.7 Simulation and Results 55
3.7.1 Experiment 1: Performances of the Various Least Squares against Noise 55
3.7.2 Experiment 2: Estimating Different Motion 58
3.7.3 Experiment 3: Results on Real Data 69
3.8 Conclusion 72

4 Measuring 3D Velocity Field through Spatial-Temporal Gradient Analysis 74

4.1 Introduction 75
4.1.1 Low-Level Processing vs. High-Level Processing 75
4.1.2 3D Velocity Vector Field vs. 3D Displacement Vector Field 77
4.2 Related Work 78
4.3 Least Squares Methods 79
4.4 Robust Regression Methods 82
4.5 Clustering Approaches 84
4.6 Least Squares, Robust Regression vs. Clustering Approaches 88
4.7 Using the Second Order Derivative Information 88
4.7.1 Derivation of the Formula 89
4.7.2 Geometric Explanation 92
4.7.3 3D Displacement Field: The Discontinuous Case 93
4.7.4 Implementation 98
4.8 Experimental Results 103

4.8.1 Doll 109
4.8.2 A Piece of Paper 117
4.8.3 Grip 121
4.8.4 Quadratic Surface 121
4.8.5 Half Sphere and Half Ellipsoid 124
4.9 Conclusion 125

5 Analyzing 3D Velocity Field by Matching Surface Features 129
5.1 Introduction 130
5.1.1 Motion Invariance 130
5.2 Computing Surface Features 132
5.3 Matching Strategy 140
5.4 Coping with Nonunique Peaks 143
5.4.1 Algorithm 1 144
5.4.2 Algorithm 2 148
5.5 Experimental Results 152
5.5.1 A Piece of Paper 154
5.5.2 Grip 154
5.5.3 Doll 157
5.6 Further Improvement 157
5.6.1 Using the Principal Directions 158
5.6.2 Decoupling Rotation and Translation 159
5.6.3 Increasing Speed 159
5.7 Conclusion 160

6 General Motion 161
6.1 Representation of the High Level Process 161
6.2 Overview of the Approach 163
6.3 Local Measurement of Point Motion Parameters 165
6.3.1 Uniform Locally Pure Translation or Dominant Translation 165
6.3.2 Uniform Locally Rigid Motion 165

6.3.3 Arbitrary Motion: Nonrigid and Nonuniform 167
6.4 Regularization of Point Motion Parameters 169
6.5 *Segmentation* by Motion Coherence 170
6.6 Estimation of Region Motion Parameters 170
6.7 Experiments 171
6.8 Conclusion 171

7 Conclusion **175**
7.1 The Long Term Process 175
7.2 The Short Term Process 176

Bibliography 180

A Linear Algebra **193**
A.1 Singular Value Decomposition 193
A.2 Inverse of a 3×3 Matrix 194

B Differential Geometry Review **195**
B.1 Space Curves 195
B.2 Surface 197

C Simulating 3D Rigid Motion **204**

D Approximating Quadric **207**

E Representing a Moving Object in a World Coordinate System **210**

LIST OF FIGURES

1.1 The principle of a range scanner. 5

2.1 A 4 × 4 Hopfield neural network. 16
2.2 The input-output voltage relationship of a neuron. 17
2.3 (a) A δ-bound matching. (b) Three δ bound matching points. 20
2.4 State error vs. iterative time. 24
2.5 The output V matrix for $A = B = D = 500$, $C = 200$, $M = N = 10$, $N_1 = 20$, $U_0 = 0.01$. 25

3.1 Coordinate system, where $(v_1, v_2, v_3), (\omega_1, \omega_2, \omega_3)$ are translational and rotational velocities in the x, y and z directions. 32
3.2 Local velocity constraint plane. 33
3.3 The aperture problem. 34
3.4 For a plane (a) and a cylindrical surface (b), their normal vectors are coplanar, therefore, pure translation cannot be determined. 38
3.5 Three cases when $\mathbf{r}_i'$ are coplanar. 40
3.6 When three planes P_1, P_2 and P_3 are orthogonal to each other, the problem tend to be well-conditioned, where plane P_i is constructed by two vectors $\mathbf{r}_i$ ans $\mathbf{n}_i$. 50
3.7 (a) Sum of fitting errors is more sensitive to random noise than variance of fitting errors. (b) Sum of fitting errors is more sensitive to the height of a jump edge than variance of fitting errors. 53
3.8 A synthetic range image. 58
3.9 The relative error of translation e_t is plotted as a function of noise NSR for LS, WLS, NLS,NWLS, TLS, WTLS, NTLS and NWTLS methods. . 59
3.10 The relative error of rotation axis e_n is plotted as a function of noise NSR for LS, WLS, NLS,NWLS, TLS, WTLS, NTLS and NWTLS methods. . . 60

3.11 The relative error of rotation angle ϵ_θ is plotted as a function of noise NSR for LS, WLS, NLS,NWLS, TLS, WTLS, NTLS and NWTLS methods. . . 61
3.12 The relative error of translation ϵ_t is plotted as a function of noise NSR for LS, WLS, NLS, and NWLS methods. 62
3.13 The relative error of rotation axis ϵ_n is plotted as a function of noise NSR for LS, WLS, NLS and NWLS methods. 63
3.14 The relative error of rotation angle ϵ_θ is plotted as a function of noise NSR for LS, WLS, NLS and NWLS methods. 64
3.15 The weight image of the synthetic range image. 65
3.16 The variance image of fitting errors of the synthetic image. 65
3.17 Grip5 range image taken by a range finder. 66
3.18 The weight image of Grip5. 66
3.19 The image of variance of fitting errors of Grip5. 67
3.20 The *break points* in Grip5 range image. 67
3.21 "Shut.yin" sequence: top: shutp10.yin, bottom: shutm10.yin. 70
3.22 Estimated motion parameters for "shut" sequence using WLS. 71

4.1 An example of perceived motion by our visual system. 76
4.2 If more than one moving object is present in S_n, least squares methods provide the average of the translational velocities of these objects. 81
4.3 When one or more points in S_n are affected by large noise such that their velocity constraint planes are far away from the correct intercepting point, least squares methods give a unreliable solutions. 81
4.4 Velocity constraint planes intercept at a common point if S_n undergoes pure translation. 85
4.5 Velocity constraint planes intercept in the vicinity of a common point if S_n undergoes dominant translation. 85

4.6 Possible distributions of clusters: (a) one blob if only a single surface is translating in S_n, (b) more than one blob if more than one moving surfaces in S_n, (c) one blob and some scattered points if part of the surface in S_n is subject to noise, (d) scattered points if the surface is very noisy, (e) scattered points along one axis if the surface in S_n is cylindrical. 87
4.7 Coordinate system. 89
4.8 Eight types of fundamental surfaces. 91
4.9 Principal coordinate system for planar and cylindrical surfaces. 92
4.10 Coordinate system for 3D displacement. 94
4.11 The neighborhood size for surface fitting. 99
4.12 An example of subsampling scheme for the LMS method, where the neighborhood is 5×5, it is subsampled at black dots such that the effective window is 3×3. 103
4.13 Left: the first balloon image, right: the second balloon image. 104
4.14 The projected 2D velocity field on u_1u_2-plane of the estimated velocity from the basic algorithm, where the neighborhood is 9×9. 106
4.15 The projected 2D velocity field on u_1u_3-plane of the estimated velocity from the basic algorithm, where the neighborhood is 9×9. 107
4.16 The projected 2D velocity field on u_2u_3-plane of the estimated velocity from the basic algorithm, where the neighborhood is 9×9. 107
4.17 The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where $thgau = 0.00001$, $thfit = 20.0$, $thcond = 20.0$, $thvel = 20.0$. 108
4.18 The projected 2D velocity field on u_1u_3-plane of the reliable velocity field, where $thgau = 0.00001$, $thfit = 20.0$, $thcond = 20.0$, $thvel = 20.0$. 108
4.19 The projected 2D velocity field on u_2u_3-plane of the reliable velocity field, where $thgau = 0.00001$, $thfit = 20.0$, $thcond = 20.0$, $thvel = 20.0$. 109
4.20 Upper left: binary image of Gaussian curvature. Upper right: binary image of condition number. Bottom left: binary image of fitting error. Bottom right: binary image of maximum allowable velocity. 110

4.21 Histograms of the Gaussian curvature, condition number and fitting error (see text for details). Upper left: Gaussian curvature, threshold is 0.1. Upper right: condition number. Bottom left: fitting error of the first image. Bottom right: fitting error of the second image. 111
4.22 The projected 2D velocity field on u_1u_2-plane of the smoothed velocity field after 5 iterations, where $k = 16$. 112
4.23 The projected 2D velocity field on u_1u_3-plane of the smoothed velocity field after 5 iterations, where $k = 16$. 112
4.24 The projected 2D velocity field on u_2u_3-plane of the smoothed velocity field after 5 iterations, where $k = 16$. 113
4.25 The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, where the neighborhood is 9×9. 113
4.26 Top: the first doll image. Bottom: the second doll image. 114
4.27 The 2D velocity field on u_1u_2-plane of the estimated velocity field, where the neighborhood is 9×9. 115
4.28 The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where $thgau = 0.0001$, $thfit = 100.0$, $thcond = 50.0$, $thvel = 3.0$. 115
4.29 Upper left: binary image of Gaussian curvature. Upper right: binary image of condition number. Bottom left: binary image of fitting error. Bottom right: binary image of maximum allowable velocity. 116
4.30 The projected 2D velocity field on u_1u_2-plane of the smoothed velocity field after 100 iterations, where $k = 16$. 116
4.31 The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, where the neighborhood is 9×9. 117
4.32 A piece of paper. 118
4.33 A piece of paper in wire frame plot. 118
4.34 The projected 2D velocity field on u_1u_2-plane of the estimated velocity field where the neighborhood is 9×9. 119
4.35 The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where $thgau = 0.001$, $thfit = 50.0$, $thcond = 20.0$, $thvel = 5.0$. 119

4.36 Upper left: binary image of Gaussian curvature. Upper right: binary image of condition number. Bottom left: binary image of fitting error. Bottom right: binary image of maximum allowable velocity. 120
4.37 The projected 2D velocity field on u_1u_2-plane of the smoothed velocity field after 10 iterations, where $k = 16$. 120
4.38 The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, where the neighborhood is 5×5. 121
4.39 The projected 2D velocity field on u_1u_2-plane of the estimated velocity field, where the neighborhood is 9×9. 122
4.40 The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where $thgau = 0.0001$, $thfit = 150.0$, $thcond = 50.0$, $threl = 5.0$. 122
4.41 Upper left: binary image of Gaussian curvature. Upper right: binary image of condition number. Bottom left: binary image of fitting error. Bottom right: binary image of maximum allowable velocity. 123
4.42 The projected 2D velocity field on u_1u_2- plane of the estimated velocity field using the LMS method, the neighborhood is 7×7. 123
4.43 The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where the neighborhood is 5×5. 124
4.44 The projected 2D velocity field on u_2u_3-plane of the estimated velocity field using the LMS method, where the neighborhood is 5×5. 125
4.45 The projected 2D velocity field on u_1u_2-plane of the estimated velocity field, where the neighborhood is 5×5. 126
4.46 The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, where the neighborhood is 5×5. 126
4.47 The projected 2D velocity field on u_1u_2-plane of the estimated velocity field, where the neighborhood is 5×5. 127
4.48 The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, the neighborhood is 5×5. 127

5.1 (a) The surface normal N, the orientation angle θ and the tilt angle $\hat{\theta}$. (b) The surface element in the differential geometry. The orthogonal unit vectors $\mathbf{U_1}, \mathbf{U_2}$ and N, define the orientation of the surface element, where N is the normal vector, $\mathbf{U_1}$ is the direction of maximum surface curvature, and $\mathbf{U_2}$ is the direction of minimum surface curvature. 135
5.2 The disk C_1 and its projection C_2 in the xy space. 136
5.3 The normal vectors around the periphery of C_1. 137
5.4 Interpolation on the xy-plane. 139
5.5 Determining the sign of the curvatures. 141
5.6 A synthetic polyhedron image. 144
5.7 Correlation surfaces for the polyhedron image. 145
5.8 (a) The correlation surface at corner pixel (15, 15). (b) The correlation surface at line edge pixel (32, 17). 146
5.9 Thinning binary image of correlation surfaces for the polyhedron. 147
5.10 Classification of correlation surfaces for the polyhedron using the first algorithm. 148
5.11 Classification of correlation surfaces for the polyhedron using the second algorithm. 150
5.12 Peak line. 151
5.13 Full velocity field by the first algorithm. 152
5.14 Normal velocity field by the first algorithm. 153
5.15 Full velocity field by the second algorithm. 153
5.16 Normal velocity field by the second algorithm. 154
5.17 Maximum curvature image of paper sequence. 155
5.18 Minimum curvature image of paper sequence. 155
5.19 Classification of peaks of correlation surfaces of paper sequence. 156
5.20 Full velocity field of paper sequence. 156
5.21 Estimated full velocity field of grip image sequence. 157
5.22 Estimated full velocity field of doll image sequence. 158

6.1 Block diagram of the approach. 164
6.2 Synthetic image: half sphere - half ellipse. 172
6.3 Estimated point parameters of synthetic image: half sphere - half ellipse. . . 172
6.4 Smoothed point parameters of synthetic image: half sphere - half ellipse. . . 173
6.5 Weight image for smoothing. 173
6.6 Segmented synthetic image using motion coherence: half sphere - half ellipse. 174

B.1 Characteristics of a space curve, where $\mathbf{X}'(s) = \frac{d\mathbf{X}(s)}{ds}$ and $\mathbf{X}''(s) = \frac{d\mathbf{X}'(s)}{ds}$. 196
B.2 Osculating planes determined by vectors **t** and **n**. 197
B.3 (a) Normal curvature. (b) Normal section. 200
B.4 Surface shapes and principal curvatures. 203

C.1 Simulating 3D motion. (a) Approximating a local surface by plane facets. (b) Resampling by locating the intercepting point of a light ray and the facet. 206

E.1 $xyzt$ and $XYZT$ represent the world coordinate system (WCS) and object-centered coordinate system (OCS), where the OCS be aligned with WCS at the initial time $t = t_0 = 0$. 211

LIST OF TABLES

2.1 Results of correspondence (noiseless) for $M = 10$, $N = 10$, $N_1 = 12$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [5\ 5\ 5]^T$, $\theta = 10^0$. 25

2.2 Results of correspondence (noiseless) for $M = 15$, $N = 15$, $N_1 = 17$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [10\ 10\ 10]^T$, $\theta = 20^0$. 26

2.3 Performance statistics for estimated parameters (noiseless) for $M = 10$, $N = 10$, $N_1 = 12$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [5\ 5\ 5]^T$, $\theta = 10^0$. 27

2.4 Performance statistics for estimated parameters (noiseless) for $M = 15$, $N = 15$, $N_1 = 17$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [10\ 10\ 10]^T$, $\theta = 20^0$. 27

2.5 Results of correspondence (with added Noise) for $M = 15$, $N = 15$, $N_1 = 17$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [10\ 10\ 10]^T$, $\theta = 20^0$. 27

2.6 Performance statistics for estimated parameters (with added noise) for $M = 15$, $N = 15$, $N_1 = 17$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [10\ 10\ 10]^T$, $\theta = 20^0$. 27

3.1 Estimated rigid motion parameters using WLS grip range image. 68

3.2 Estimated motion parameters by WLS and LS without *break points* for Grip5 range image, thvar = 3. 68

3.3 Estimated region motion parameters for shut sequence using WLS. 69

Chapter 1

Introduction

1.1 2D Apparent Motion, 3D Motion, Methodologies and Difficulties

Understanding of motion is one of the principal requirements for a machine or an animal to interact meaningfully with its environment. Although human beings appear to perform this task quite effortlessly, attempts to have a machine duplicate it turns out to be a non-trivial task.

When objects move in front of an observer (camera or sensor) or when an observer moves through a fixed environment, there are corresponding changes in the successive images, these changes are called *2D apparent motion* in images. To recover the relative 3D motion between objects and the observer from the 2D apparent motion has been one of the major goals pursued by researchers interested in time-varying image processing. Obviously, at least two main problems exist. The first involves extracting the 2D apparent motion from a sequence of images, and the second involves inferring 3D motion from the extracted apparent motion.

By the term an *image*, one means the function, $F(x, y)$, at the pixel location (x, y). The function values may represent depth, intensity, color, or any other signal, depending on the type of sensor used. The most widely studied images for motion understanding are intensity images. In this case, the function represents the brightness pattern (intensity)

of a scene and the apparent motion is also called *2D visual motion.* It is natural for researchers to choose this kind of images in studying motion analysis since it seems that the human vision system uses the same kind of input.

Some recent reviews of this area can be found in [81, 5, 107]. It is generally accepted that the study of motion analysis involves two stages: *the measurement of the apparent motion* and *the interpretation of the measured apparent motion* [1] [103, 47]. In analogy with human vision systems [18, 19], schemes to compute visual motion can be classified according to the spatial-temporal range over which methods are applicable: called *short term process* and *long term process.* Other classifications distinguish between the fundamentally different processes involved; on the one hand are *gradient-based schemes* (for examples see [99, 54, 92, 93, 80, 110] etc.) which use spatial-temporal image brightness gradients to derive image motion, and on the other hand are *correspondence or similarity matching methods* (for example see [118, 100, 102, 125, 33, 113] etc.). The former are intrinsically short range, while the latter can be short range or long range. For gradient-based schemes, computing visual motion is also known as computing *optical flow.* Usually, a dense depth image and motion parameters are recovered simultaneously from the optical flow. For the latter, computing visual motion is a problem of extracting proper features, such as corners, line segments, facets, etc., and establishing the correspondence of features in successive images in order to compute the displacements between features. Only sparse depth values and motion parameters can be recovered from a finite number of displacements.

There is no doubt that significant progress has been made over the last decade, however, a general solution still eludes researchers. Computing visual motion still remains a problem for which a bewildering battery of different techniques continue to be proposed. Moreover, even when the visual motion has been recovered, it may not be immediately obvious how it is related to the *projected motion*, the perspective projection of 3D motion of a scene onto the image plane — if at all. Most work done so far simply assumes that

[1] It is by no mean clear whether or not biological systems explicitly recover 2D visual motion as the initial step, and to what extent biological systems use this route alone to process motion — certainly, other techniques are available. Indeed, the route is not necessary in computational vision: Nagahdaripour, Horn and Heel [84, 83, 46] have demonstrated the recovery of surface structure and motion without measuring 2D visual motion. Aloimonos [6] has shown that determining the 3D Motion of a rigid surface patch can be done without correspondence.

the apparent motion is equivalent to the projected motion. However, they are generally different, unless some special conditions are satisfied. Unfortunately, these conditions are rarely true in the real world. The well-known example is Horn's rotating sphere [51] [2]. The more general situations have been shown by Verri and Poggio [109].

Assuming that there are no differences between projected motion and 2D visual motion, then it is well known that 2D visual motion depends on both scene structure and motion [75, 74]. There is a depth/speed scaling ambiguity inherent in monocular motion processing—it is quite impossible to decide whether something is large, far-off and moving quickly, or small, nearby and moving slowly. The relation itself is highly nonlinear, which results in problems such as nonunique interpretation of 3D motion from 2D visual motion, high sensitivity to noise in 2D visual motion, etc. Some ambiguities are inherent [2, 82]. Despite the concentration of effort on explicit recovery of discrete visual motion, Verri and Poggio have raised the questions about the usefulness of quantitative optical flow [108].

Perhaps the fundamental reason for the difficulties of the task is the many-to-one nature of perspective projections. An infinite number of 3D objects can correspond to any single view. And therefore, very often the signal processing involved is ill-posed in the sense that one set of raw data permits a large, possibly infinite number of solutions. Therefore constraints about the way the world works or more usually, about the way we expect the world to work must be imposed in order to reconstruct a scene from imagery, moving or otherwise. Such constraints used may range from planar surface, smooth surface, curved surface, rigid object, smooth optical flow, constant optical flow in a neighborhood, etc.

Now one may ask, is there any hope for the success of visual motion processing? How can one proceed? Perhaps no one knows the exact answer. Instead, people tend not to ask which of these methods is the most correct, but rather which of these methods is most useful for the task at hand? This attitude results in numerous techniques which sound very successful in some special conditions. Unfortunately, this choice seems to be the most practical one at the moment.

Now let us ask ourselves what does motion really mean? Quoted from Ullman [103],

[2] (a) A smooth sphere is rotating under constant illumination - the image does not change (zero apparent motion), yet the projected motion is nonzero. (b) A fixed sphere is illuminated by a moving source - the shading in the image changes, yet the projected motion is zero

> Motion is a continuous change of location and disposition, hence information concerning the 3D structure of objects, and the way it changes over, bears directly on the motion of the objects in question.

The process discussed before is called *structure from motion* by Ullman. He has also demonstrated that human vision systems employ depth cues to infer motion and he called this process *motion from structure*. Quoting again from Ullman,

> When both processes are applied to a scene, they usually agree in their results. They will, however fail to do so when the static perception of structure which governs the motion from structure process, is erroneous.

Ullman also pointed out that there are some cases where structure from motion fails to recover motion, while motion from structure succeeds. Therefore, if structure information can be obtained from sources other than intensity images, a robust and general solution may be found for the motion problem. Fortunately, this information (depth, range) can be obtained through stereopsis [41, 76], structured light [85] or an active sensor such as a range scanner [58, 59]. The above observations motivated us to seek computational algorithms for motion from structure. Specifically, the author is interested in motion analysis from a sequence of images obtained from a range scanner, which will be briefly described in the next section.

1.2 A Triangulating Range Scanner

The principle of a range scanner using triangulation is very simple [64]. Figure 1.1 is a simplified illustration of such a scanner described in Rioux [87]. Light from laser source L is reflected from mirror M_1 and hits a point P_1 in a scene. At P_1 the light is scattered, some of which is focused by the collector lens onto the fixed mirror M_3. From M_3 the light is reflected to the back side of mirror M_1 and the light comes to a focus at P_1'. In the same direction, the scattered light from another point P_2 in the scene comes to focus at P_2'. The distance between P_1 and P_2 corresponds to the distance between P_1' and P_2'. A linear array of photocells is placed in the P_1' to P_2' line. A point P' is detected by locating

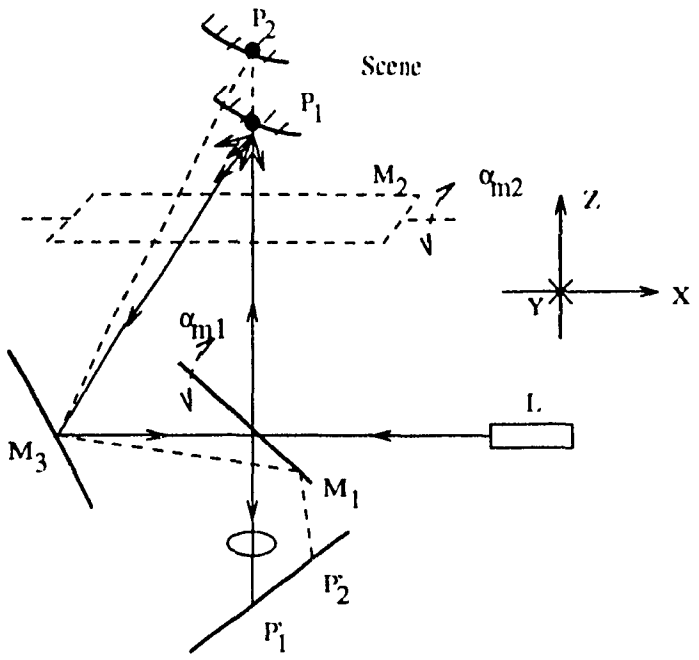


Figure 1.1: The principle of a range scanner.

the maximum signal on the linear array. The angle α_{m1} is measured and corresponds to x-dimension. Another mirror M_2 is used to deflect both the outgoing and incoming light. The angle α_{m2} corresponds to the y-dimension. After calibration and coordinate conversions, the range image $F(x, y)$ is obtained from the mirror angles α_{m1} and α_{m2}, and distance (P'_0, P') where P'_0 is some reference point and P' is the location of the maximum signal on the linear photocell array. Such scanners have been designed for various scene volumes, for example, $1m \times 1m \times 1m$. Some versions are commercially available.[3]

There are several drawbacks to the range scanner. The laser light source, used because of its ability to maintain focus over a long distance, can be harmful to the unprotected human eye. If a surface in a scene is totally absorbing (very black or "furry") or highly reflective, then no light is returned and the scanner registers infinite distance. If a scene contains very reflective surfaces, then the light may "bounce around" and may be returned at greater intensity to the array from some other direction than the correct direction in the scene, resulting in distance error. Finally, certain places "seen" by M_3 cannot be illuminated by M_1 or vice versa. However, the absence of light on the diode

[3]Servo-Robot, 1380 Graham Bell, Boucherville, Quebec J4B 6H5

array is an indicator of infinite distance or some difficulties in "seeing", and corrective actions can be taken later.

Despite some of the scanning difficulties, the images obtained are very good. The number of pixels in each dimension is from 256 up to 1024. The quantization accuracy on the z axis varies with the depth, since it depends on geometry. It also depends on the number of photocells used in the array, and the accuracy of locating the position of the peak signal along the photocell array. There are several other ways of detecting the range from the observer to the points in a scene, such as stereo, radar, etc. (for details see [58, 59, 13]), but the laser range scanner "beats" these methods in its simplicity of design and richness of information delivered since a laser scanner is capable of giving the distance as well as grey level and even color images of the scene.

1.3 Range Images vs. Intensity Images

Range images provide the position information, thus the apparent motion in range images is directly related to 3D motion. This greatly simplifies the problem of motion analysis. As will be shown later on, 3D motion parameters are linearly related to apparent motion in some cases, therefore the problem turns out to be well-posed. Besides, the problem of analyzing motion is decoupled from the problem of recovering structure.

Consider a simple example where a mobile robot has to estimate the collision time with either a moving object or the fixed environment in order to move safely around. It is extremely easy to extract this information from range images, while it still remains a difficult task if using intensity images. In spite of all these potentials, range scanners will not replace intensity cameras because of the following four reasons. First, it is much more expensive than a normal camera. Second, it uses a laser beam which may not be desired in some applications, such as in military applications. Third, it has a smaller field of view than a normal camera. Fourth, it usually takes longer time to obtain an image even though the prototype of a video-rate range scanner [12] has been developed by the National Research Council (NRC) in Canada, but the image quality is not as good as for the slower scanner. However, it is also useful in some cases where a normal camera will

fail absolutely, for example, it can "see" at night, and it also works better than a normal camera when objects in a scene have nearly uniform illumination and spectral reflectance characteristics. In other words, the study of motion from range image sequences is very important as an alternative tool.

1.4 Previous Work

Few researchers have studied motion from range image sequences. The main reason, the author believes, is that a range scanner represents a rather new technique and it used to be very slow, therefore, it was not suitable for capturing time-varying images. Nevertheless, there are still several pioneers in this area. One recent review about different approaches developed to estimate motion parameters from a sequence of two range images is presented by Sabata and Aggarwal [89]. The authors subdivided the main issues involved in the detection of motion from a sequence of range images into three steps:

1. the segmentation of the range images in each frame and extraction of the key features,
2. the establishment of the correspondence of the key features between frames, and
3. the computation of motion using these feature correspondences.

In the paper, it is assumed that the key features used are produced *a priori* by an image segmentation algorithm. Also, the correspondence between pairs of features is assumed to be established in the preprocessing stage. The discussion is focussed on the third issue under the assumption of rigid object motion. In this case, the problem becomes optimal parameter estimation. Motion parameters have to be restrained by physical constraints, such as the *rotation matrix* has to be orthonormal. In this way, motion estimation usually turns out to be a constrained nonlinear optimization problem. Various techniques to solve this optimization problem for different kinds of features are discussed in the paper. Segmentation of range images have been extensively studied by researchers in the processing of 3D images [14, 13, 63, 65, 61].

Kehtarnavaz and Mohan [67] gave a framework for estimation of rigid motion parameters from range images. They first segmented images into small patches whose centers of

masses were taken as feature points, then they established the correspondence between the patches using a graph matching technique, and rigid motion parameters were estimated using a least squares technique. The weakness of this method is that the center point of each segment may not exactly correspond to the same 3D point when the object is viewed from different angles because of sampling of the surface.

Yamamoto et al. [117] described a method capable of estimating motion parameters of a rigid object as well as a nonrigid object. They used a net of $n \times n$ nodes connected by links as a deformable model to represent surface motion of a nonrigid object. A link can stretch but not bend. The links which connect to a universal joint at each node can independently rotate relative to each other. Then they derived the linear relations between the small variations of the link parameters (the length and the direction in polar coordinates), i.e., motion parameters, and the displacement. Motion parameters are obtained by solving a system of linear equations, which are usually very large. Therefore, the solution will be sensitive to noise and the equations can easily become ill-conditioned unless the number of motion parameters is restricted according to the nature of actual object motion. It also required that surfaces be smooth.

Horn and Harris [53] extended the gradient-based scheme for optical flow to range image and directly estimated rigid motion parameters using a least squares technique. This method is similar to a part of chapter 3 in this thesis and to the method for rigid motion estimation in [117]. However the present algorithm was developed independently. In addition, the discontinuities of surfaces are considered, and the theoretical analysis is carried out about conditions for unique motion interpretation and error propagation, which are lacking in the other two papers.

Several other papers propose methods to estimate motion from intensity images with the assumption of known depth [4, 116]. Chaudhuri and Chatterjee [22] estimated local deformation parameters from the given 3D point correspondences. Other techniques which may be useful for motion analysis are the ones for pose determination [15, 20, 25].

In summary, most methods proposed so far are correspondence-based. For these kinds of methods, reliable feature extracting and matching remains a difficult problem, even though the features from range images may possess physical or geometric meaning.

Once the correspondence is established, the estimation of rigid motion parameters is made easier by using range images than by using intensity images. Not much work has been done toward the analysis of motion of multiple objects, and nonrigid objects.

1.5 Contributions

The main goal of this work is to demonstrate the feasibility and potential of estimating motion from a sequence of range images as an alternate solution to the complex motion problem. Our work is motivated by witnessing the hard struggle of recovering motion from intensity images and by the impressive new range sensor techniques. We believe that range images may be more suitable for the study of the motion problem, since they contain geometric information of a scene. To prove this, we have explored a collection of ideas and concepts that have been around motion understanding for the past few decades and modified them to be useful for range images. We have also developed some new ideas which are suitable for our problem. The principal contributions are as follows:

- We have shown that a Hopfield neural network can be used to establish feature correspondence. A model is developed to convert the correspondence problem to the problem of minimizing an energy function, which occurs at the stable state of a Hopfield neural network. After establishing the feature matching, a δ bound matching concept is introduced to detect the reliable matching features, therefore increase the accuracy of the estimated motion parameters by removing the effect of mismatching features. In this way, the algorithm is tolerant to noise due to feature detection or occlusion.

- A simple, yet powerful, algorithm is proposed to estimate motion of a single rigid object. A 3D velocity constraint equation is derived by extending the well-known brightness change constraint equation, and the motion problem is modeled as solving a set of linear equations. A weighted least squares technique has been found to provide the best performance among several other versions of least squares techniques. We have also proved the necessary and sufficient conditions for the unique interpretation of the motion parameters. The sensitivity of the estimated motion parameters

to noise is analyzed. These theoretical studies provide the further insight into the behavior of the algorithm. Experiments on real range image sequence provided the promising results.

- An new algorithm was developed to estimate a 3D velocity field, which is useful for interpreting motion of rigid and nonrigid objects. It has been shown that the 3D velocity of a point on a smooth surface can be uniquely determined by its first and second order partial derivatives except at a *parabolic point*. We also proved that the same algorithm can be derived through gradient based analysis or similarity matching, which are usually considered as two totally different principles in motion understanding. This proof allowed us to gain a better understanding of these two fundamental principles. Some practical techniques are proposed to cope with discontinuities of surfaces and *parabolic points*.

- We have also demonstrated that 3D velocity field can be estimated based on the correlation of the local structure of principle curvatures, which are invariant to rigid motion. The computation method proposed by Kasvand [62], which simulates the derivation of curvatures in differential geometry is used to estimate curvatures. Since large curvatures are computed more accurately than small curvatures because of noise in images, the standard correlation is modified to adapt to this computational problem. Two techniques are proposed to deal with the correlation surfaces with nonunique peaks. This algorithm can even estimate full velocities of corners, which can not be provided by the previous surface. However, this last algorithm is very computationally intensive. Therefore, we suggested that it should be combined with the previous algorithm to give the best performance.

1.6 Overview of the Study

The next chapter emphasizes the issue of correspondence. A Hopfield neural network is used to perform feature matching, and an algorithm, which is able to reliably estimate motion parameters from the matching results, is developed. The features can be either 3D points, which are provided by a range scanner, or can be corners, lines, facets etc. If all 3D

points in the images are used as features, then feature extraction can be bypassed. In this case, the number of feature points is huge. However, since neural networks are essentially based on parallel processing, the speed will not be reduced dramatically by increasing the number of features, which is one of the advantages of using neural network for matching.

Chapter 3 is devoted to the estimation of short-term rigid motion of a single object. An algorithm, which is based on an extension of the *brightness change constraint equation* [54], is used to directly estimate rigid motion parameters by weighted least squares technique. Theoretical analysis of error propagations and conditions for unique interpretation of motion parameters are also provided in the chapter.

Chapter 4 describes an algorithm for measuring a 3D velocity field. It is shown that 3D velocities can be uniquely determined by the first and second order partial derivatives except at points with zero *Gaussian curvatures*. For each measured velocity, a reliability measure is provided simultaneously. It will also be proven that the same algorithm can be obtained by similarity matching and by a gradient-based method under the assumption that a local smooth surface can be approximated by a second degree polynomial.

Chapter 5 describes an algorithm for computing the 3D velocity by feature matching. Theoretical justification for the invariance of principal curvature features to rigid motion is obtained by means of differential geometry. Then a matching strategy is discussed. Subsequently, two post-processing procedures are proposed to deal with points on line edges such that the vernier velocities can be estimated at those points.

Chapter 6 proposes a framework for general motion. It divides the task into: preprocessing, estimation of point motion parameters, regularization of point motion parameters, clustering based on coherent motion and estimation of region motion parameters. Theoretically, it can be used for any kinds of motion.

The final chapter, chapter 7, gives a brief discussion of unsolved problems and areas of future research.

Chapter 2

Motion Estimation Based on Correspondence

An algorithm for estimating motion parameters of a rigid body from range images is presented in this chapter. The best correspondence between two sets of three-dimensional points is established using Hopfield neural network. Here the "best" is in the sense that the number of found matching pairs which satisfy the physical constraints should be as large as possible. Once the correspondence is built, the *δ-bound* matching concept is introduced to select reliable matching pairs, which are then used to estimate the motion parameters. The proposed method can be tolerant to random noise and missing points. It is easily extended to plane and surface matching. Simulation results are given for noisy synthetic data.

2.1 Introduction

Correspondence is a process that identifies elements in different views as representing the same object at different times, therefore maintaining the perceptual identity of objects in motion or change [103]. This definition gives rise to several questions. What are these elements? How can they be located? How can they be related to the object to which they belong? How are these elements recognized? How can these features be tracked? Unfortunately, the features are in many cases application-dependent, therefore, no single

approach will be useful for any applications.

The task of establishing and maintaining correspondence is nontrivial. The ambiguity is aggravated by the effects of occlusion which cause features to appear or disappear and also give rise to "false" features. The development of robust techniques to solve the correspondence problem is an active area of research and is still in its infancy. The same problem is common to other areas of computer vision such as stereoscopy, pose determination and model-based object recognition.

Aggarwal et al. [3] have classified correspondence processes into two categories: those based on iconic models and those based on structural models. The former approach uses templates extracted from the first frame which are then detected in the second and subsequent frames. The second approach consists of extracting tokens with a number of attributes from the first image, and using domain constraints and structural models to match these tokens with those extracted from the second and subsequent images. The former approach will be discussed in chapter 5. Here we will emphasize the latter one.

Sethi and Jain [94] described a method to establish correspondence between feature points extracted from a long sequence of monocular images. Their algorithms are based on preserving the smoothness of velocity changes. The iterative optimization algorithms search for an optimal set of trajectories for feature points in a sequence of images based on constraints on the direction and magnitude of change in motion. A hypothesize and test approach is also proposed to handle occlusion. This method hypothesizes occlusion if the number of feature points detected in a frame is less than that detected in two or more proceeding or succeeding frames. Interpolating the missing point position using the preceding two frames and testing this with the subsequent two frames verifies the existence of occlusion. Experiments with manually extracted features illustrate that the approach is able to deal with limited occlusion. The problem of automated extraction of features, however, has not been addressed by the authors.

Another class of matching strategies considers matching problem as finding isomorphic subgraphs from two graphs or finding isomorphic subtrees from two trees using different tree-pruning algorithms, such as, Wong [115], Baird [10], Umeyama and Kasvand [105]. Umeyama and Kasvand [106] proposed an eigendecomposition approach to solve a

weighted graph matching problem.

The relaxation algorithm initially developed by Renade and Rosenfield [86] and several modified versions have also been applied to the matching problem. In Fang and Huang's [31] paper, the relaxation algorithm is modified by incorporating different scales to allow for large scale changes in the images (due to large translation in depth). Kim [68] have applied the relaxation technique for matching features in stereo imagery as well as for matching 3-D features in depth maps. Barnard and Thompson [11] have proposed an iterative relaxation labeling technique for matching features in stereo imagery based on smoothness in change of depth.

In this chapter, an algorithm is presented for finding the best correspondence between two sets of three-dimensional points using Hopfield [49, 50] neural network. Once the correspondence is built, the δ-bound matching concept is introduced to select reliable matching pairs, which are then used to estimate motion parameters. The proposed method is tolerant to random noise and missing points due to occlusion, and is easily extended to plane and surface matching. Section 2.2 gives a brief description of Hopfield neural network. Section 2.3 discusses point correspondence using neural network. Section 2.4 describes δ-bound matching and the algorithm for estimating motion parameters. Simulation results are given in Section 2.5. Section 2.6 gives the extension of the algorithm to plane and surface matching.

2.2 Neural Network

The neural network under consideration is based on the Hopfield model. In this network, each neuron is implemented by an analog amplifier and an accompanying resistor, capacitor circuitry. As an example, Figure 2.1 shows a 4×4 Hopfield neural network, where neurons are arranged in a matrix form and each neuron is identified by a set of double indices x and i indicating its row and column respectively. The input-output voltage relationship of a neuron on row x and column i is given as $V_{xi} = g(U_{xi})$ shown in Figure 2.2, which is a sigmoid function. There is a feedback path among pairs of neurons, designated as $T_{xi,yj}$[1],

[1]Conductance $T_{xi,yj}$ connects the input of neuron xi to the output of neuron yj

and referred to as a connection matrix. Further, there is an external bias I_{xi} supplied to each neurons. The differential equation describing the dynamics of a neuron is given by

$$\frac{dU_{xi}}{dt} = -\frac{U_{xi}}{\tau} + \sum_{y=1}^{N}\sum_{j=1}^{N} T_{xi,yj}V_{xi} + I_{xi} \tag{2.1}$$

where $\tau = Rc$ is time constant.

Hopfield has shown that equation (2.1) for a network with symmetric connection ($T_{xi,yj} = T_{yj,xi}$) has a convergent solution to stable state, in which the outputs of all neurons remain constant. Also, when the width of the amplifier gain curve is narrow the high gain limit—the stable states of a network comprised of N^2 neurons are the local minima of the equation

$$E = -\frac{1}{2}\sum_{x=1}^{N}\sum_{i=1}^{N}\sum_{y=1}^{N}\sum_{j=1}^{N} T_{xi,yj}V_{xi}V_{yj} - \sum_{x=1}^{N}\sum_{i=1}^{N} V_{xi}I_{xi} \tag{2.2}$$

The state space over which the circuit operates is the interior of the N-dimensional hypercube defined by $V_{xi} = 0$ or 1. However, in the high-gain limit, the minima only occur at corners of this space. Hence, the stable states of the network correspond to those locations in the discrete space consisting of the 2^{N^2} corners of this hypercube which minimizes equation (2.2).

2.3 Point Corresponding Using Neural Network

To solve the point corresponding problem, an energy function is constructed in the form of equation (2.2), for which local minima correspond to best point matching. It is assumed that the three-dimensional coordinates of a set of feature points on an object at two time instances have been obtained, designated as $\mathbf{a} = \{\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_N\}$ and $\mathbf{b} = \{\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_M\}$. For calculating all distances between points in the same set, two distance matrices $\mathbf{D_a}$ and $\mathbf{D_b}$ are obtained, where the element $D_a(x,i)$ denotes the distance between points $\mathbf{a}_x$ and $\mathbf{a}_i$, and the element $D_b(y,j)$ denotes the distance between points $\mathbf{b}_y$ and $\mathbf{b}_j$. For rigid motion, distance $D_a(x,i)$ should equal to distance $D_b(y,j)$ if points $\mathbf{a}_x$ and $\mathbf{a}_i$ match points $\mathbf{b}_y$ and $\mathbf{b}_j$ respectively. Point correspondence is found under this constraint. For each point in set $\mathbf{a}$, there are M possible corresponding points in set $\mathbf{b}$, or no corresponding point in set $\mathbf{b}$, but it is not allowed for one point in one set to correspond to more than one

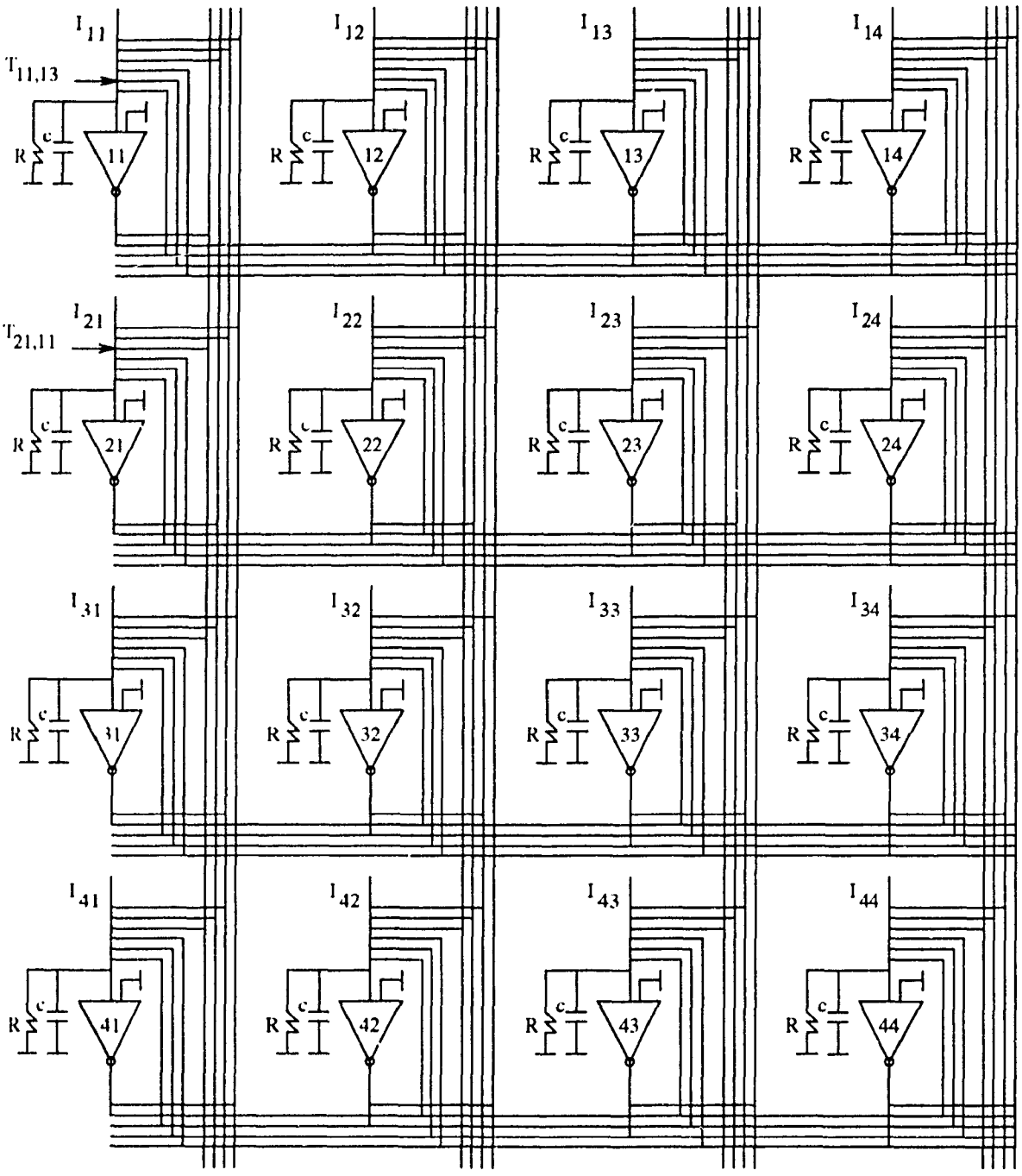


Figure 2.1: A 4×4 Hopfield neural network.

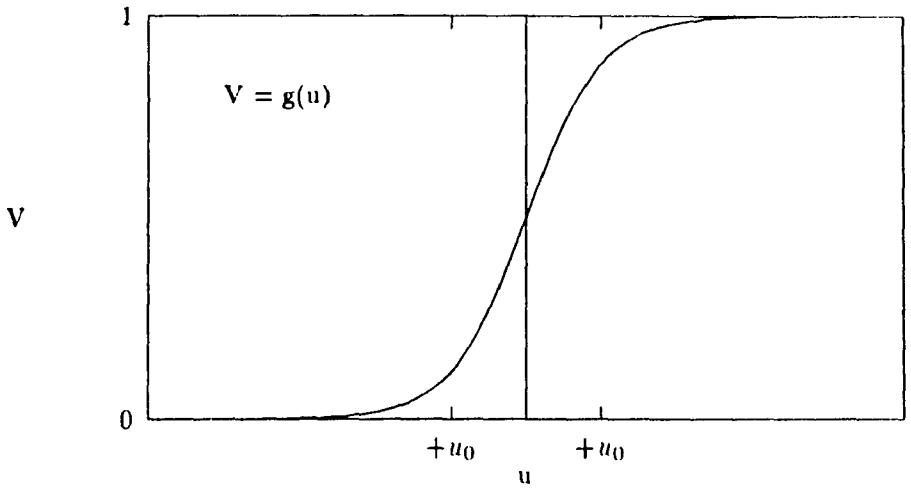


Figure 2.2: The input-output voltage relationship of a neuron.

point in another set. Therefore, in the case $N \leq M$, there are $\frac{M!}{(M-N)!}$ possible correspondence relations if every point in set **a** has a corresponding point in set **b**. Constructing an $M \times N$ permutation matrix V whose rows represent points in set **b**, and columns represent points in set **a**

$$V_{xi} = \begin{cases} 1 & \text{if point } \mathbf{b}_x \text{ matches point } \mathbf{a}_i \\ 0 & \text{otherwise} \end{cases}$$

The above discussion shows that there are no more than one element whose value equals one at each row and column for this matrix. The following matrix is an example.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

It represents correspondence between points $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_4, \mathbf{b}_5$ and $\mathbf{a}_1, \mathbf{a}_4, \mathbf{a}_2, \mathbf{a}_5, \mathbf{a}_3$ respectively. It is desirable to find one of such matrix from numerous possible matrices, which represents the desired matching.

Constructing an energy function in the following form makes the lowest energy correspond to the best matching. The "best" is in the sense that the number of found matching pairs which satisfy the physical constraints should be as large as possible.

$$E = \frac{A}{2}\sum_{x=1}^{M}\sum_{i=1}^{N}\sum_{j\neq i,j=1}^{N} V_{xi}V_{xj} + \frac{B}{2}\sum_{i=1}^{N}\sum_{x=1}^{M}\sum_{y\neq x,y=1}^{M} V_{xi}V_{yi} + \frac{C}{2}(\sum_{x=1}^{M}\sum_{i=1}^{N} V_{xi} - N_1)^2$$
$$+\frac{D}{2}\sum_{x=1}^{M}\sum_{y\neq x,y=1}^{M}\sum_{i=1}^{N}\sum_{j\neq i,j=1}^{N} V_{xi}V_{yj}|D_b(x,y) - D_a(i,j)| \qquad (2.3)$$

where the first term equals zero if and only if there is no more than one "1" at each row of V, which embodies the constraint that one point in **b** is not allowed to be matched to more than one point in **a**, the second term equals zero if and only if there is no more than one "1" at each column of V, which represents the constraint that one point in **a** is not allowed to be matched to more than one point in **b**, the third term equals zero if and only if there are N_1 "1's" in the matrix V and the fourth term refers to the rigidity constraints. Obviously, V represents the desired matching when E reaches a minimum. E can be rewritten as follows

$$E = \frac{A}{2}\sum_{x,y=1}^{M}\sum_{i,j=1}^{N} V_{xi}V_{yj}\delta_{xy}(1-\delta_{ij}) + \frac{B}{2}\sum_{x,y=1}^{M}\sum_{i,j=1}^{N} V_{xi}V_{yj}\delta_{ij}(1-\delta_{xy})$$
$$+\frac{C}{2}\sum_{x,y=1}^{M}\sum_{i,j=1}^{N} V_{xi}V_{yi} - CN_1\sum_{x=1}^{M}\sum_{i=1}^{N} V_{xi} + \frac{C}{2}N_1^2$$
$$+\frac{D}{2}\sum_{x,y=1}^{M}\sum_{i,j=1}^{N} V_{xi}V_{yj}(1-\delta_{xy})(1-\delta_{ij})|D_b(x,y) - D_a(i,j)| \qquad (2.4)$$

where

$$\delta_{xy} = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases}$$

Since the term $\frac{C}{2}N_1^2$ is constant, it does not affect locating the minimum of the energy function, therefore, it can be omitted.

Let V_{ij} be the output of neuron (i,j) and comparing equation (2.4) with equation (2.2), we get

$$T_{xi,yj} = -A\delta_{xy}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{xy})$$
$$-C - D(1-\delta_{xy})(1-\delta_{ij})|D_b(x,y) - D_a(i,j)|$$
$$I_{xi} = +CN_1 \qquad (2.5)$$

where the first term refers to inhibitory connections within each row, the second term refers to inhibitory connections within each column, the third refers to global inhibition and the fourth is the data term. Substituting $T_{xi,yj}$ and I_{xi} into equation (2.1), the analog network for the matching problem corresponds to the equations of neuron state

$$\begin{aligned}
\frac{dU_{xi}}{dt} &= -\frac{U_{xi}}{\tau} - A\sum_{j\neq i,j=1}^{N} V_{xj} - B\sum_{y\neq x,y=1}^{M} V_{yi} - C(\sum_{y=1}^{M}\sum_{j=1}^{N} V_{yj} - N_1) \\
&\quad -D\sum_{y\neq x,y=1}^{M}\sum_{j\neq i,j=1}^{N} V_{yj}|D_b(x,y) - D_a(i,j)| \\
V_{xi} &= g(U_{xi}) = \frac{1}{2}(1+\tanh(U_{xi}/U_0)
\end{aligned} \tag{2.6}$$

where τ is set to 1 without loss of generality. Solving the equation sets, the outputs for the stable state give the desired matching.

2.4 Motion Estimation Using δ-Bound Matching Points

Any rigid motion can be uniquely represented by a rotation around an axis passing through the origin of a coordinate system followed by a translation, that is

$$\mathbf{p}' = \mathbf{R}\mathbf{p} + \mathbf{T} \tag{2.7}$$

where $\mathbf{p}$ and $\mathbf{p}'$ represent 3D coordinates of a point at time t and $t + \Delta t$ respectively, $\mathbf{R}$ represents the rotation matrix and $\mathbf{T}$ is the translation vector.

$$\mathbf{R} = \begin{bmatrix} (n_x^2-1)c+1 & n_xn_zc - n_ys & n_xn_zc + n_ys \\ n_yn_xc + n_zs & (n_y^2-1)c+1 & n_yn_zc - n_xs \\ n_zn_xc - n_ys & n_zn_yc + n_xs & (n_z^2-1)c+1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

$$\mathbf{T} = [\ T_x \quad T_y \quad T_z\ ]^T, \quad c = (1-\cos\theta), \quad s = \sin\theta$$

where $\mathbf{n} = [\ n_x \quad n_y \quad n_z\ ]^T$ is the unit vector of rotation axis, θ is the rotation angle and ^{T} denotes transpose.

Given two corresponding point sets with noise, $P = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n\}$ and $P' = \{\mathbf{p}'_1, \mathbf{p}'_2, \ldots, \mathbf{p}'_n\}$

$$\mathbf{p}'_i = \mathbf{R}\mathbf{p}_i + \mathbf{T} + \mathbf{N}_i \tag{2.8}$$

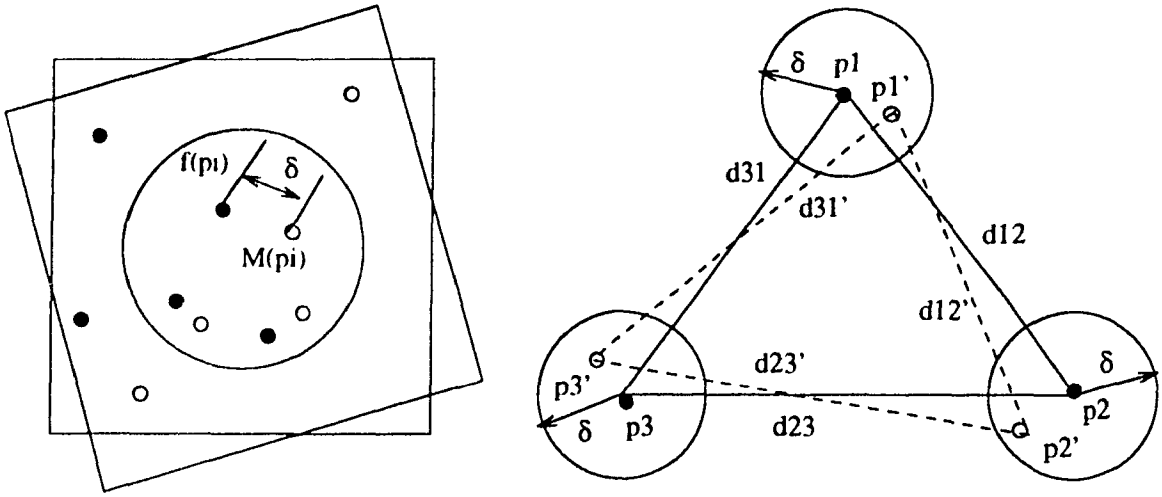


Figure 2.3: (a) A δ-bound matching. (b) Three δ-bound matching points.

where $\mathbf{N}_i$ is noise. $\mathbf{R}$ and $\mathbf{T}$ are found by minimizing

$$\varepsilon = 1/n \sum_{i=1}^{n} \|\mathbf{p}_i' - (\mathbf{R}\mathbf{p}_i + \mathbf{T})\|^2 \tag{2.9}$$

There are lots of algorithms [16, 72, 73, 9, 32, 52] proposed to solve this problem. A recent review of the various techniques can be found in [89]. Most of the techniques require that the good correspondence should be established, which is very strict considering the difficulty of the correspondence problem.

In order to increase estimation accuracy, the δ-bound matching concept given in [105] is introduced to select reliable matching pairs, which are then used to estimate motion parameters. Let f be a one-to-one mapping from point set P onto point set P', it is said that f is δ-bound matching if the following holds

$$d = \max_{\mathbf{p}_i \in P} \| f(\mathbf{p}_i) - (\mathbf{R}\mathbf{p}_i + \mathbf{T}) \| \leq \delta \tag{2.10}$$

where $f(\mathbf{p}_i)$ is a point in P' corresponding to $\mathbf{p}_i$ in P.

The definition of δ-boundness means that when a point in P' is superposed on a point in P by the optimal rotation and translation, the maximum distance between the corresponding points in P and P' is less than or equal to δ, see Figure 2.3(a), where $M(\mathbf{P}_i) = \mathbf{R}\mathbf{P}_i + \mathbf{T}$.

Umeyama and Kasvand [105] has shown that the least-mean squared error ε is given as follows

$$\varepsilon = S_{P'}^2 - [tr(\mathbf{DS})]^2 / S_P^2 \tag{2.11}$$

where S_P^2 and $S_{P'}^2$ are variances around the mean vectors of P and P', $\mathbf{S}_{PP'} = \mathbf{UDV}^T$ is a covariance matrix of P and P', $\mathbf{UDV^T}$ is singular value decomposition of $\mathbf{S}_{PP'}$ $\mathbf{D} = diag(d_i), d_1 \geq d_2 \geq \cdots \geq d_m \geq 0$

$$\mathbf{S} = \begin{cases} I & \text{if } det(\mathbf{S}_{PP'}) \geq 0 \\ diag(1, 1, \cdots, -1) & \text{if } det(\mathbf{S}_{PP'}) < 0 \end{cases}$$

When $rank(\mathbf{S}_{PP'}) = m$. The optimal $\mathbf{R}$ and $\mathbf{T}$ are uniquely determined as follows.

$$\begin{aligned} \mathbf{R} &= \mathbf{USV^T} \\ \mathbf{T} &= \mathbf{m}_{P'} - \mathbf{R}\mathbf{m}_P \end{aligned} \tag{2.12}$$

where $\mathbf{m}_P$ and $\mathbf{m}_{P'}$ are mean vectors of P and P'.

Starting from this basic theory, we have developed an algorithm as follows:

1. Find three δ-bound matching pairs $(\mathbf{p}_1, \mathbf{p}'_1), (\mathbf{p}_2, \mathbf{p}'_2), (\mathbf{p}_3, \mathbf{p}'_3)$ from given point correspondence under the following three constraints.

   $$(d_{12} - 2\delta) \leq d'_{12} \leq (d_{12} + 2\delta)$$

   $$(d_{23} - 2\delta) \leq d'_{23} \leq (d_{23} + 2\delta)$$

   $$(d_{31} - 2\delta) \leq d'_{31} \leq (d_{31} + 2\delta)$$

   where $d_{12}, d'_{12}, d_{23}, d'_{23}, d_{31}, d'_{31}$ are shown in Figure 2.3(b). R and T can be calculated from equation (2.12) using these three matching pairs.

2. Suppose k δ-bound matching pairs have been obtained, consider a new correspondence $(\mathbf{p}_{k+1}, \mathbf{p}'_{k+1})$, then $(\mathbf{m}_{P,k+1}, \mathbf{m}_{P',k+1}, S^2_{P,k+1}, S^2_{P',k+1}, \mathbf{S}_{PP',k+1})$ can be computed recursively from $(\mathbf{m}_{P,k}, \mathbf{m}_{P',k}, S^2_{P,k}, S^2_{P',k}, \mathbf{S}_{PP',k})$, that is

   $$\mathbf{m}_{P,k+1} = [k/(k+1)](\mathbf{m}_{P,k} + \mathbf{p}_{k+1}/k)$$

$$
\begin{aligned}
\mathbf{m}_{P',k+1} &= [k/(k+1)](\mathbf{m}_{P',k} + \mathbf{p}'_{k+1}/k) \\
S^2_{P,k+1} &= [k/(k+1)][S^2_{P,k} + \| \mathbf{m}_{P,k} - \mathbf{p}_{k+1} \|^2 /(k+1)] \\
S^2_{P',k+1} &= [k/(k+1)][S^2_{P',k} + \| \mathbf{m}_{P',k} - \mathbf{p}'_{k+1} \|^2 /(k+1)] \\
\mathbf{S}_{PP',k+1} &= [k/(k+1)][\mathbf{S}_{PP',k} + (\mathbf{m}_{P',k} - \mathbf{p}'_{k+1})(\mathbf{m}_{P,k} - \mathbf{p}_{k+1})/(k+1)]
\end{aligned}
$$

3. Find the singular value decomposition of $\mathbf{S}_{PP',k+1}$, and calculate ε from equation (2.11). If $\varepsilon \leq \delta^2$, then $(\mathbf{p}_{k+1}, \mathbf{p}'_{k+1})$ is a δ-bound matching and $\mathbf{R}$ and $\mathbf{T}$ are replaced by the new estimated values.

4. Repeat step 2 until all matching pairs have been considered.

5. The unit vector $\mathbf{n} = [\ n_x \quad n_y \quad n_z\ ]^T$ and θ can be easily computed from $\mathbf{R}$ using the algorithm given by Tsai and Huang in [101]

$$
\begin{aligned}
\sin\theta &= \pm d/2 \\
n_x &= \pm(r_{32} - r_{23})/d \\
n_y &= \pm(r_{13} - r_{31})/d \\
n_z &= \pm(r_{21} - r_{12})/d \\
\cos\theta &= \frac{d^2 r_{11} - (r_{32} - r_{23})^2}{d^2 - (r_{32} - r_{23})^2}
\end{aligned}
$$

where $d^2 = (r_{32} - r_{23})^2 + (r_{13} - r_{31})^2 + (r_{21} - r_{12})^2$. The signs of the rotation angle and rotation axis cannot be decided because these two sets of solutions are physically indistinguishable. If $d = 0$, then $\theta = 0, R = I$ and n_x, n_y, n_z can be anything since the rotation axis is meaningless without rotation.

2.5 Simulation

A network for matching two sets of 3D points using the connection matrix and the input bias terms defined in equation (2.5) was simulated on a digital computer. N points of set **a** were chosen at random (with uniform probability density) on the interior of a cube whose edge lengths were 256. Given motion parameters $\mathbf{n}, \mathbf{T}$ and θ, N corresponding points of set **a** were calculated using equation (2.7). Those N_2 points which were still in the cube

after motion were taken as part of point set **b**. The remaining $(N - N_2)$ points were considered as missing points which were invisible in the second time instance, and were thrown away. $M - N_2$ points which were chosen in the same way as for point set **a** were added to point set **b** (random choice). These points were considered as new feature points which were only visible in the second time instance or random noise points. Two Euclidean distance matrices were computed and normalized to make the selection of parameters in equation (2.3) easier.

The Runge-Kutta-Verner fifth-order and sixth-order method[2] was used to solve the differential equation set. The initial values were selected as given in [49], which is the following

$$U_{xi} = U_{00} + \delta U_{xi}$$

where $U_{00} = MN/N_1$ which gave an unbiased choice, δU_{xi} was added to break the symmetry which will lead to divergence. It is a random variable with uniform distribution in the interval

$$-0.1U_0 \leq \delta U_{xi} \leq 0.1U_0$$

It is reasonable to let $A = B = D, C < A, N_1 \geq N$ in equation (2.3). Chosing A, B, C, D is relatively easier than chosing N_1. Several sets of parameters were tried, and it has been found that the following set is suitable.

$$\begin{aligned} A &= 500, \quad B = 500 \\ C &= 200, \quad D = 500 \\ U_0 &= 0.01 \end{aligned}$$

Then, experiments for different values of N_1 were performed. Figure 2.4 shows the results, where curves give the speed in which neurons approach stable states. State error is defined as

$$e(t) = \max_{x,i} |U_{xi}(t) - U_{xi}(t-1)| \tag{2.13}$$

where t denotes iterative time. The results show that there exists an optimal N_1 which leads to the shortest converging time. For $N_1 = 10$, and 15, all of the correct matching

[2]Subroutine IVPRK in IMSL library was used to solve the differential equation set

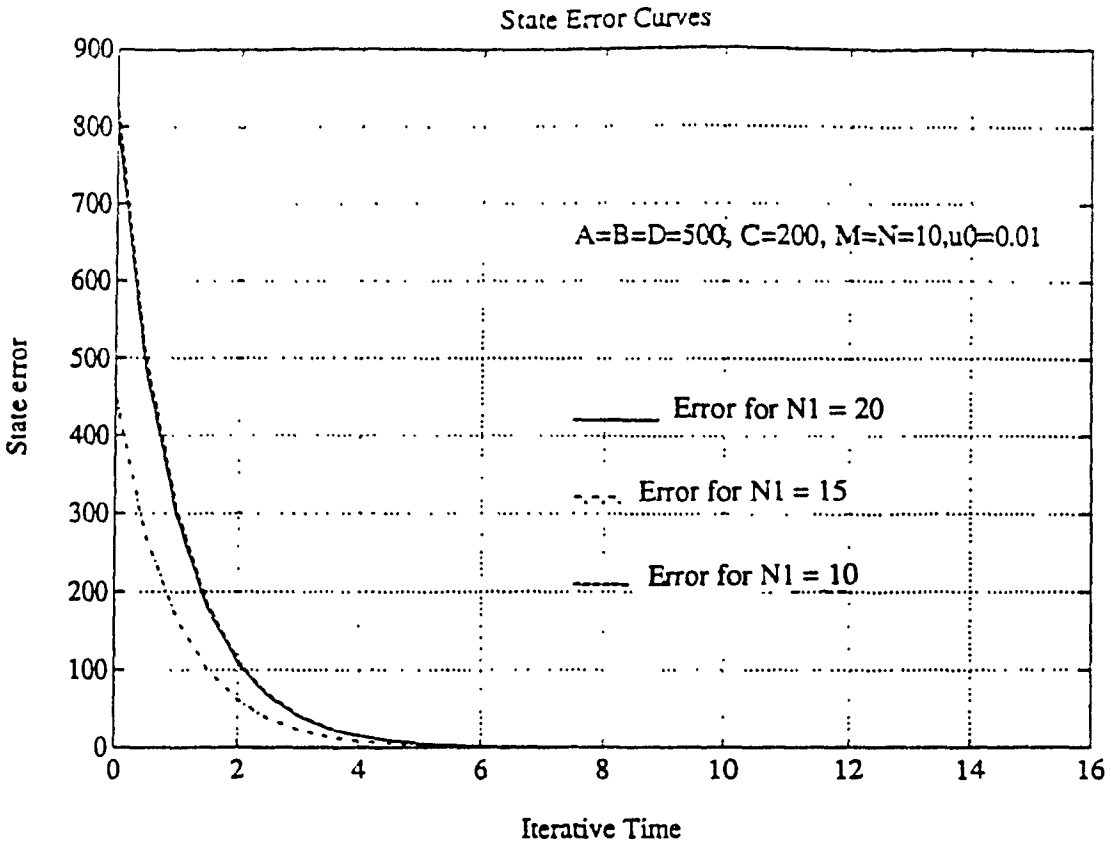


Figure 2.4: State error vs. iterative time.

pairs were found. For $N_1 = 20$, all of the correct matching pairs were also found , but some extra wrong matching pairs were obtained (see ⊞ in Figure 2.5). This was expected since the system was asked to find much more matching pairs (20) than the real number (10). However, these extra wrong matching pairs were singled out by the followed δ-bound matching verification. It is better to let N_1 be slightly larger than the actual matching number.

Since data sets were created randomly, the experiments were repeated 10 times for each set of given motion parameters. More results on matching can be found in tables 2.1 and 2.2. As shown in these tables, all correct matching pairs are found, but some extra unmatching pairs are also obtained because the exact number of matching pairs is unknown. However, these unmatching pairs can be distinguished from correctly matching pairs by the δ-bound matching test. In these experiments, it is assumed that there exists at least 70% correspondence between two point sets.

Tables 2.3 and 2.4 give the results of estimated motion parameters. Let $\tilde{\mathbf{R}}, \tilde{\mathbf{T}}, \tilde{\mathbf{n}}$ and $\tilde{\theta}$ denote the estimated motion parameters, $\epsilon_R, \epsilon_T, \epsilon_n$ and ϵ_θ represent the absolute errors

1	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	1	0	0	0	0
0	1	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1

Figure 2.5: The output V matrix for $A = B = D = 500$, $C = 200$, $M = N = 10$, $N_1 = 20$, $U_0 = 0.01$.

Table 2.1: Results of correspondence (noiseless) for $M = 10$, $N = 10$, $N_1 = 12$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [5\ 5\ 5]^T$, $\theta = 10^0$.

Trial Number	Input Number of Matching Points	Output Number of matching points	
		Correct matching	Wrong matching
1	9	9	0
2	10	10	0
3	9	9	0
4	9	9	0
5	10	10	0
6	9	9	0
7	9	9	0
8	10	10	0
9	10	10	0
10	9	9	1

Table 2.2: Results of correspondence (noiseless) for $M = 15$, $N = 15$, $N_1 = 17$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [10\ 10\ 10]^T$, $\theta = 20^0$.

Trial Number	Input Number of Matching Points	Output Number of matching points	
		Correct matching	Wrong matching
1	14	14	0
2	14	14	0
3	14	14	0
4	15	15	0
5	15	15	0
6	14	14	0
7	14	14	0
8	13	13	1
9	15	15	0
10	10	10	3

between the estimated values and the actual values, that is,

$$\begin{aligned} \epsilon_R &= \sum_{i,j=1}^{3} |R(i,j) - \tilde{R}(i,j)| \\ \epsilon_T &= \sum_{i=1}^{3} |T(i) - \tilde{T}(i)| \\ \epsilon_n &= |n_x - \tilde{n}_x| + |n_y - \tilde{n}_y| + |n_z - \tilde{n}_z| \\ \epsilon_\theta &= |\theta - \tilde{\theta}| \end{aligned}$$

The bias and standard deviation (denoted by 'standard dev.' in the tables) for each motion parameter are defined as the mean and standard deviation of absolute errors for repeated trials. The average percent error (denoted by 'avg % error' in the tables) is defined as the percentage of bias with respect to the real value. As can be seen from the tables, the results are quite satisfactory.

The algorithm's sensitivity to noise was tested by adding zero mean Gaussian random noise $N(0,5)$ to point set **b**. Table 2.5 shows the matching results and table 2.6 shows the statistics for the estimated motion parameters. The results are still very promising.

Table 2.3: Performance statistics for estimated parameters (noiseless) for $M = 10$, $N = 10$, $N_1 = 12$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [5\ 5\ 5]^T$, $\theta = 10^0$.

10 Trials	ϵ_R	ϵ_T	ϵ_θ	ϵ_n
bias	1.7293e-04	-0.0235	-4.2373e-04	-0.0312
standard dev.	8.9849e-04	0.1983	0.0059	0.0575
avg % error	0.64	3.35	0.86	0.5

Table 2.4: Performance statistics for estimated parameters (noiseless) for $M = 15$, $N = 15$, $N_1 = 17$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [10\ 10\ 10]^T$, $\theta = 20^0$.

10 Trials	ϵ_R	ϵ_T	ϵ_θ	ϵ_n
bias	3.1671e-04	-0.0924	-2.2559e-04	-0.0447
standard dev.	9.8695e-04	0.2191	0.0033	0.0311
avg % error	0.36	2.03	0.47	0.23

Table 2.5: Results of correspondence (with added Noise) for $M = 15$, $N = 15$, $N_1 = 17$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [10\ \ 0\ 10]^T$, $\theta = 20^0$.

Trial Number	Input Number of Matching Points	Output Number of matching points	
		Correct matching	Wrong matching
1	14	14	0
2	14	14	1
3	14	14	0
4	14	14	0
5	12	12	1
6	13	13	1
7	12	12	1
8	12	12	1
9	13	13	1
10	10	10	3

Table 2.6: Performance statistics for estimated parameters (with added noise) for $M = 15$, $N = 15$, $N_1 = 17$, $\mathbf{n} = [0.7\ 0.5\ 0.51]^T$, $\mathbf{T} = [10\ 10\ 10]^T$, $\theta = 20^0$.

10 Trials	ϵ_R	ϵ_T	ϵ_θ	ϵ_n
bias	2.9602e-04	-0.1707	-4.3970e-04	0.0527
standard dev.	0.0051	1.2076	0.0143	0.3410
avg % error	1.86	9.91	2.22	1.28

2.6 Extension of the Algorithm for Plane and Surface Correspondence

The algorithm for point correspondence can be easily extended to plane and surface matching. For plane matching, given two sets of planes P_a, P_b, constructing a permutation matrix in the same way as for point matching, computing normalized angles $\Theta_a(i,j)$ between planes i,j in plane set **a** and $\Theta_b(i,j)$ between planes i,j in plane set **b**, we then can construct the energy function as follows

$$E = \frac{A}{2}\sum_{j=1}^{M}\sum_{i=1}^{N}\sum_{j\neq i,j=1}^{N} V_{ij}V_{ij} + \frac{B}{2}\sum_{i=1}^{N}\sum_{x=i}^{M}\sum_{y\neq x,y=1}^{M} V_{xi}V_{yi} + \frac{C}{2}\sum_{x=1}^{M}\sum_{i=1}^{N}(V_{xi}-N_1)^2$$
$$+\frac{D}{2}\sum_{x=1}^{M}\sum_{y\neq x,y=1}^{M}\sum_{i=1}^{N}\sum_{j\neq i,j=1}^{N} V_{xi}V_{yj}|\Theta_b(x,y)-\Theta_a(i,j)| \qquad (2.14)$$

Using the Hopfield network, the plane correspondences can then be found. Once the plane correspondence has been established, the motion parameters can be estimated by the algorithm given in [105].

For surface matching, the principal axis of a region, or whichever attributes of a surface that are invariant to rigid motion, can be used to construct an energy function. Faugeras [32] gave an algorithm for estimating motion parameters using matching quadratic surfaces.

2.7 Conclusion

In this chapter, an algorithm has been presented to find the matching of features between two successive frames, and estimate rigid motion parameters. The match is found through a Hopfield neural network. An energy function is constructed based on the physical constraints between matching pairs such that the desired matching corresponds to the minimum value of the energy function, which happens to be the stable state of the Hopfield network. Therefore, the problem becomes to finding the stable state of Hopfield network.

In the energy function, a parameter which described the number of matching pairs

has to be predefined. This parameter is usually unknown before the matching is established. Fortunately, the experimental results show that it is better to let it be slightly larger than the actual number of matching pairs, therefore, it can be set to the maximum number of points among two given sets. When the real number of matching pairs are much smaller than this number, the neural network will also give some extra matching pairs, which are detected by the δ-bound matching test. In this way, a high accuracy of estimated motion parameters can be reached.

One of the remaining problems in the correspondence-based method is how to reliably extract features. Using the method presented in this chapter, this step may be bypassed. All the 3D coordinates provided by a range scanner can be simply taken as features. In this case, the number of features is enormous. It will take too long for a conventional matching algorithm to establish the proper matching. However, it will not dramatically increase the processing time for a neural network. This is a very interesting area. Unfortunately, there were no facilities available for the author to do the simulation for the data sets with large number of points since the available computers were too slow.

Further experiments on real data should be done to verify the theory. This involves developing algorithms needed to extract the necessary features.

Chapter 3

Motion Estimation of a Single Rigid Object

In this chapter, a gradient-based direct method is presented for recovering the motion of an observer in a static environment from range image sequences. This direct method, based on the *local velocity constraint equation* which is an extension of commonly used *brightness change constraint equation* in optical flow, involves only solving a set of linear equations with six rigid motion parameters as unknowns.

The necessary and sufficient conditions that an object must satisfy are discussed such that motion can be uniquely determined by the proposed method.

It will be shown that the method fails to provide an unique estimate of the rigid motion parameters only for some special types of 3D structures. Their motion can not be perceived even by a human eye, if there are no other sources of information available, such as intensity.

The sensitivity of the method against noise is analyzed. The performances of several least squares methods against noise are also compared. It will be demonstrated that a weighted least squares method has the best performance. Simulation results on both synthetic and real data are given.

3.1 Introduction

Single rigid body motion is the simplest, yet very important, case in motion analysis since many real world problems belong to this category, such as a moving observer in a static environment, which is very common for a mobile robot. So far, most work related to motion is concentrated on this type of motion. In chapter 2 we discussed an algorithm to estimate rigid motion parameters. Obviously, it can be applied to both long range and short range motions. However, for short range motion, other methods can be used to recover motion parameters without dealing with the complicated correspondence problem.

In this chapter, a direct method will be described. The method only involves solving a system of linear equations to obtain rigid motion parameters. In the next section, the basic formulas used will be derived. Section 3 will show how to solve the equations to obtain a solution. In section 4, the uniqueness of motion estimation will be investigated. Section 5 will be devoted to error analysis, which provides the measure of sensitivity of the method against noise. In section 6, the various techniques of implementing the algorithm will be compared. Finally, experimental results will be provided and discussed.

3.2 Formulation of Rigid Body Motion

Let a coordinate system shown in Figure 3.1 be fixed with respect to an observer (a range finder). The z-axis points along the optical axis. A range finder can measure the 3D coordinates (x, y, z) of points on surfaces in a scene. A time-varying range image can be considered as a function of time and space, $z = F(x, y, t)$. Suppose that the surfaces are differentiable. Taking the total derivative of z with respect to time t using the chain rule yields

$$\frac{dz}{dt} = \frac{\partial F}{\partial x}\frac{dx}{dt} + \frac{\partial F}{\partial y}\frac{dy}{dt} + \frac{\partial F}{\partial t} \tag{3.1}$$

Let $\mathbf{u}^T = [\ u_1 \quad u_2 \quad u_3\ ] = [\ \frac{dx}{dt} \quad \frac{dy}{dt} \quad \frac{dz}{dt}\ ]$, $\mathbf{n}^T = [\ -F_x \quad -F_y \quad 1\ ] = [\ -\frac{\partial F}{\partial x} \quad -\frac{\partial F}{\partial y} \quad 1\ ]$ and $F_t = \frac{\partial F}{\partial t}$. then equation (3.1) can be rewritten as

$$\mathbf{n} \bullet \mathbf{u} = F_t \tag{3.2}$$

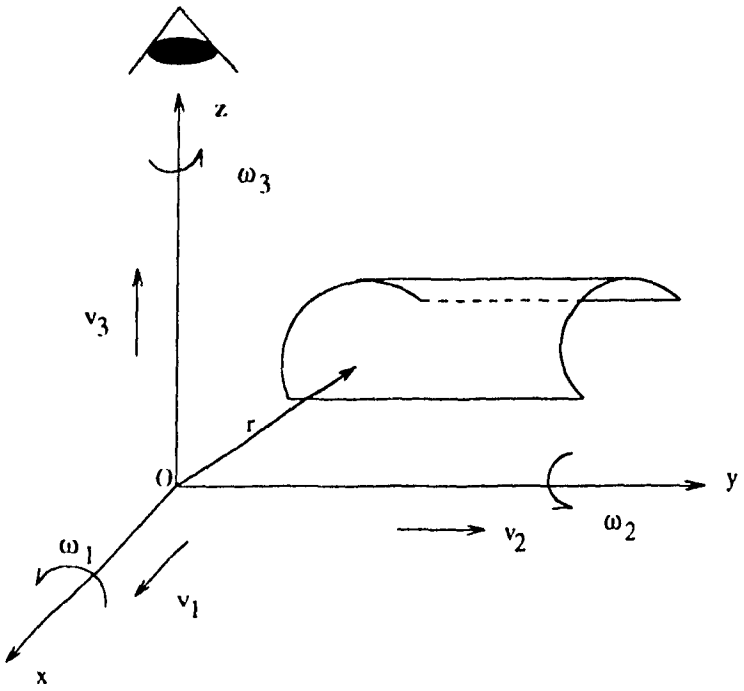


Figure 3.1: Coordinate system, where $(v_1, v_2, v_3), (\omega_1, \omega_2, \omega_3)$ are translational and rotational velocities in the x, y and z directions.

where F_x, F_y and F_t are the first order partial derivatives of z with respect to x, y and t, n is the normal vector, u_1, u_2 and u_3 are components of 3D instantaneous velocity u of point (x, y, z) in x, y and z directions, respectively, and $\bullet$ represents the vector dot product. We call this equation *a local velocity constraint equation.* This constraint restricts the 3D velocity vector of a point to a plane, which is parallel to its tangent plane (see Figure 3.2). It gives the velocity component $u^{\perp}$ in the normal direction

$$u^{\perp} = \frac{F_t}{\sqrt{1 + F_x^2 + F_y^2}}$$

Equation (3.2) is essentially very similar to the *brightness change constraint equation* used in the measurement of optical flow. Therefore, the well-known aperture problem illustrated in Figure 3.3 also exists here. Suppose that a moving plane P is analyzed by a local motion detector which examines a limited area of the image, represented by the aperture **A**. Such a detector can measure only the component of motion in the normal direction of the plane, indicated by **b**. The components of motion along the plane are invisible through this limited aperture. Thus, a local detector cannot distinguish the true

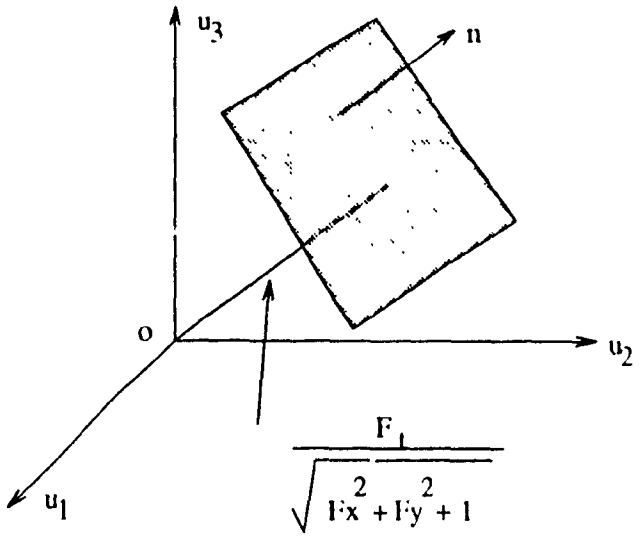


Figure 3.2: Local velocity constraint plane.

movement indicated by **a** with **b** or **c**.

Mathematically, this aperture problem is expressed by an underdetermined system of equations. In the local velocity constraint equation, the first order derivatives are measurable for each points, while three velocity components are unknown. If there are n points in an image, then we have n velocity constraint equations with $3n$ unknowns. Therefore, additional constraints have to be found in order to determine the unknown velocities. Similar to the computation of optical flow, it may be assumed that the velocity field is smooth and neighboring points have similar velocities, or velocities are constant over an entire segment of the image, or velocity field is the result of restricted motion, such as rigid body motion. In this chapter, we employ the assumption of rigid body motion. The following chapters will examine other assumptions.

Assuming that an observer is moving in a static environment, the whole image can be considered as a rigid object. As shown in Figure 3.1, any rigid body motion can be resolved into two components: translation and rotation about an axis through the origin. Let $\mathbf{v} = [\ v_1 \quad v_2 \quad v_3\ ]^T$ denote the translational velocity of the observer and $\omega = [\ \omega_1 \quad \omega_2 \quad \omega_3\ ]^T$ denote its rotational velocity. Let $\mathbf{r}^T = [\ x \quad y \quad z\ ]^T$ be the radius vector of a point on the surface, then according to classical mechanics, the instantaneous

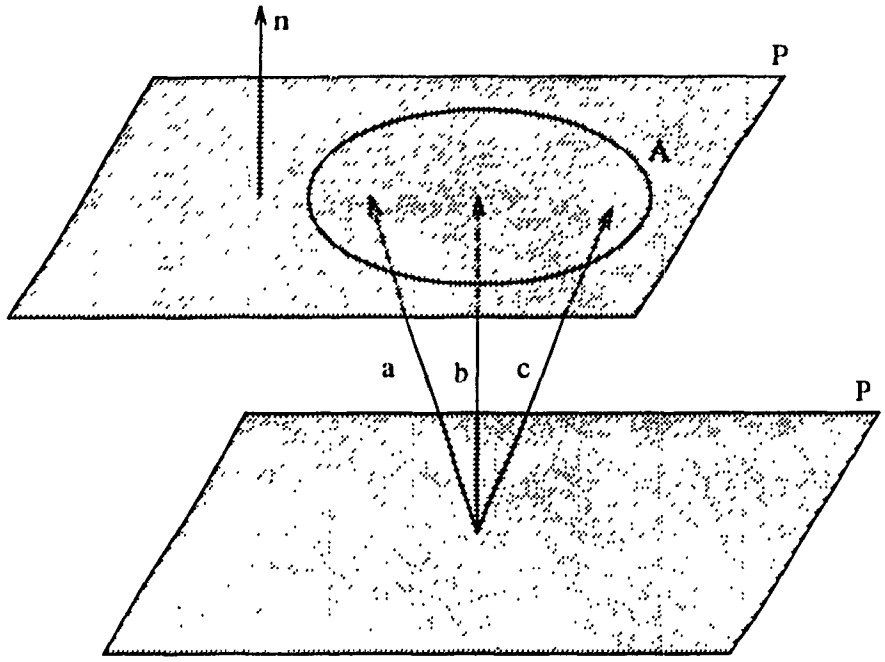


Figure 3.3: The aperture problem.

velocity of the point with respect to the observer is

$$\mathbf{u} = -(\mathbf{v} + \omega \times \mathbf{r}) \tag{3.3}$$

Substituting this relation into the velocity constraint equation produces

$$-\mathbf{n} \bullet (\mathbf{v} + \omega \times \mathbf{r}) = F_t \tag{3.4}$$

Rearranging the above equation and using equality [**abc**] = [**bca**] = [**cab**], we obtain

$$-\mathbf{n} \bullet \mathbf{v} - [\mathbf{r}\mathbf{n}\omega] = F_t \tag{3.5}$$

where [**abc**] = $\mathbf{a} \bullet (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \times \mathbf{b}) \bullet \mathbf{c}$ is vector triple product and × represent vector cross product. This is a linear equation with six rigid motion parameters, $\mathbf{v} and \omega$ as unknowns. The coefficients **r** and **n** of the equation are either measurable or computable. For each point, one such equation can be formed. If there are at least six points in the image, then it is possible to estimate the rigid motion parameters of the moving observer in a static environment or vice versa.

3.3 Solution of Rigid Body Motion

At least six points are needed to find motion parameters $\mathbf{v}$ and ω by solving a set of linear equations (see equation 3.5). In practice, more than six points are needed to combat noise in measured depth and calculated gradient values by using the least squares technique (LS). If there are n pixels in the image, the resulting n equations can be written in matrix form as

$$\mathbf{Am} = \mathbf{b} \tag{3.6}$$

where $\mathbf{A} \in \mathcal{R}^{n\times 6}, \mathbf{m} \in \mathcal{R}^{6}, \mathbf{b} \in \mathcal{R}^{n}$,

$$\mathbf{A} = \begin{bmatrix} F_{x_1} & F_{y_1} & -1 & -(F_{y_1}z_1 + y_1) & F_{x_1}z_1 + x_1 & F_{y_1}x_1 - F_{x_1}y_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ F_{x_i} & F_{y_i} & -1 & -(F_{y_i}z_i + y_i) & F_{x_i}z_i + x_i & F_{y_i}x_i - F_{x_i}y_i \\ F_{x_n} & F_{y_n} & -1 & -(F_{y_n}z_n + y_n) & F_{x_n}z_n + x_n & F_{y_n}x_n - F_{x_n}y_n \end{bmatrix}$$

$$\mathbf{m}^T = \begin{bmatrix} v_1 & v_2 & v_3 & \omega_1 & \omega_2 & \omega_3 \end{bmatrix}$$

$$\mathbf{b}^T = \begin{bmatrix} -F_{t_1} & \cdots & -F_{t_i} & \cdots & -F_{t_n} \end{bmatrix}$$

If the coefficient matrix $\mathbf{A^TA}$ is nonsingular, then equation (3.6) has an unique solution $\mathbf{m}$ which minimizes $(\mathbf{Am} - \mathbf{b})^T(\mathbf{Am} - \mathbf{b})$. The estimated motion parameters are

$$\mathbf{m} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b} \tag{3.7}$$

The requirement that the coefficient matrix $\mathbf{A}^T\mathbf{A}$ is nonsingular is not always satisfied as shown in the next section. For certain surfaces, $\mathbf{A}^T\mathbf{A}$ is singular. In this case, equation (3.6) does not have an unique solution. In other words, there exists infinite number of interpretations of the motion. What can be done in such situations? Obviously, one can simply give all the possible solutions. The interesting question is: can one reasonable solution be chosen among those infinite number of solutions? If it is possible, how can it be chosen?

Let us look at an example. If an image consists of a portion of a plane translating along the plane, it will be shown in the next section that in this case $\mathbf{A}^T\mathbf{A}$ is singular, there

exist an infinite number of interpretations of the motion. When looking at the two images in this example, the second image will be exactly the same as the first image if the plane is large enough compared to the movement between two frames. This means the apparent motion in the images is zero, thus zero motion should be a reasonable explanation although the true motion may or may not be equal to zero. Mathematically, this explanation is equivalent to selecting the solution with "minimum residue norm)". It has been proved [40] that there is only one solution to satisfy this requirement. This solution can be found using Singular Value Decomposition (SVD). From theorem A.1 in Appendix A, $\mathbf{m}$ satisfying this requirement is obtained by the following formula

$$\mathbf{m} = \mathbf{A}^{+}\mathbf{b} \tag{3.8}$$

where $\mathbf{A}^{+}$ is the pseudo-inverse of $\mathbf{A}$.

3.4 Ambiguity of Motion Perception

From the previous section, it is known that the true motion is indeterminable if the coefficient matrix $\mathbf{A}^T\mathbf{A}$ is singular. Since $\mathbf{A}^T\mathbf{A}$ is a function of the spatial gradients and position parameters of a range image, the uniqueness of the motion interpretation depends on the geometrical structure of a scene. It is interesting to investigate the types of special structures which are subject to ambiguous motion perception. In this section, the sufficient and necessary conditions for $\det(\mathbf{A}^T\mathbf{A}) = 0$ in terms of the geometrical structure of a scene will be discussed.

According to linear algebra,

$$\det(\mathbf{A}^T\mathbf{A}) = \sum_{i_1,i_2,\cdots,i_m} \det(\mathbf{A}(i_1, i_2, \cdots, i_m))^2 \tag{3.9}$$

where $\mathbf{A}(i_1, i_2, \cdots, i_m)$ is the minor composed of the $i_1^{th}, i_2^{th}, \cdots$ and i_m^{th} row of $\mathbf{A} = \mathbf{A}_{n\times m}$, the summation is taken over $\binom{m}{n}$ combinations of $(i_1, i_2, \cdots, i_m)$. Thus the necessary and sufficient condition for $\det(\mathbf{A}^T\mathbf{A}) = 0$ is that

$$\det(\mathbf{A}(i_1, i_2, \cdots, i_m)) = 0 \tag{3.10}$$

for any combination. Consequently, it is sufficient to examine the singularity of any six rows of matrix A for our algorithm. The following discussions will start from pure translation and pure rotation, then extend the results to general motion.

3.4.1 Pure Translation

For pure translation, the analysis is very simple. Without loss of generality, it is sufficient to consider the following matrix

$$\mathbf{A} = \begin{bmatrix} F_{x_1} & F_{y_1} & -1 \\ F_{x_2} & F_{y_2} & -1 \\ F_{x_3} & F_{y_3} & -1 \end{bmatrix} = - \begin{bmatrix} \mathbf{n_1}^T \\ \mathbf{n_2}^T \\ \mathbf{n_3}^T \end{bmatrix} \tag{3.11}$$

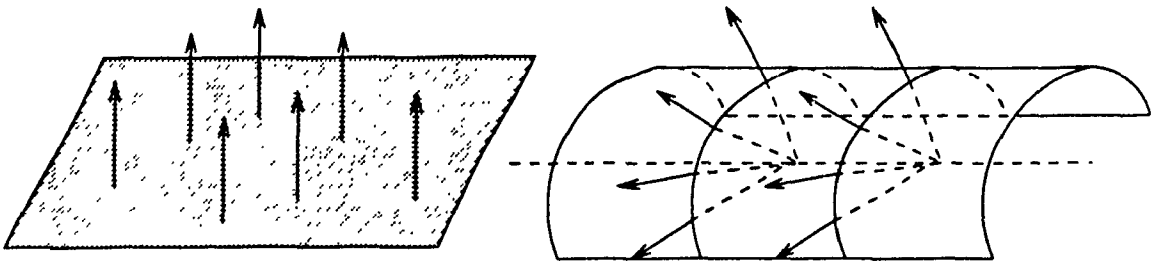
where $\mathbf{n_i}$ is normal vector at point $\mathbf{r_i}$. Points $\mathbf{r_1}, \mathbf{r_2}$ and $\mathbf{r_3}$ are any three points on a surface in a considered scene.

From linear algebra, it is known that the necessary and sufficient condition for $\det(\mathbf{A}) = 0$ is that $\mathbf{n_1}, \mathbf{n_2}$ and $\mathbf{n_3}$ are coplanar. Hence they are orthogonal to a straight line, i.e., the normal of the plane. On the other hand, if three normal vectors are orthogonal to a straight line, then $\mathbf{n_1} \times \mathbf{n_2}$ is parallel to the given straight line, therefore it is orthogonal to $\mathbf{n_3}$, then $\mathbf{n_1} \times \mathbf{n_2} \bullet \mathbf{n_3} = 0$. Since $\det(\mathbf{A}) = -\mathbf{n_1} \times \mathbf{n_2} \bullet \mathbf{n_3}$, thus, $\det(\mathbf{A}) = 0$. The following lemma has been proven.

Lemma 1 *The necessary and sufficient condition for* $\det(\mathbf{A}) = 0$ *is that all normals to a surface are orthogonal to a given line.*

In fact, if all the normals to a surface are parallel to each other, then the surface is a plane. If all the normals to a surface are orthogonal to a given line, the surface is cylindrical surface [38]. Therefore, the following corollary can be easily derived from the above discussions.

Corollary 1 *Pure translation can not be determined if and only if a surface in the scene is a plane or a cylindrical surface.*



(a) (b)

Figure 3.4: For a plane (a) and a cylindrical surface (b), their normal vectors are coplanar, therefore, pure translation cannot be determined.

3.4.2 Pure Rotation

In this case, it is sufficient to consider the matrix **A**

$$\mathbf{A} = - \begin{bmatrix} (\mathbf{r}_1 \times \mathbf{n}_1)^T \\ (\mathbf{r}_2 \times \mathbf{n}_2)^T \\ (\mathbf{r}_3 \times \mathbf{n}_3)^T \end{bmatrix} \tag{3.12}$$

We have the following conclusions:

Lemma 2 *The necessary and sufficient condition for* $\det(\mathbf{A}) = 0$ *is that one of the following conditions holds.*

- *All the normals*[1] *to a surface are parallel to each other.*
- *All the normals to a surface intercept at one common point.*
- *All the normals to a surface intercept a straight line passing through the origin.*

Proof:

$$\det(\mathbf{A}) = -(\mathbf{r}_1 \times \mathbf{n}_1) \times (\mathbf{r}_2 \times \mathbf{n}_2) \bullet (\mathbf{r}_3 \times \mathbf{n}_3) \tag{3.13}$$

Using the equalities

$$(\mathbf{a} \times \mathbf{b}) \times (\mathbf{c} \times \mathbf{d}) = [\mathbf{acd}]\mathbf{b} - [\mathbf{bcd}]\mathbf{a} = [\mathbf{abd}]\mathbf{c} - [\mathbf{abc}]\mathbf{d}$$

[1] The normal to a point is defined as a line passing through the point in the normal vector direction of the point

$$[\mathbf{abc}] = [\mathbf{bca}] = [\mathbf{cab}]$$

equation (3.13) is reduced to

$$\begin{aligned} \det(\mathbf{A}) &= -([\mathbf{r_1 r_2 n_2}]\mathbf{n_1} - [\mathbf{n_1 r_2 n_2}]\mathbf{r_1}) \bullet (\mathbf{r_3} \times \mathbf{n_3}) \\ &= [\mathbf{n_2 n_1 r_2}][\mathbf{r_1 r_3 n_3}] - [\mathbf{r_1 r_2 n_2}][\mathbf{n_3 n_1 r_3}] \end{aligned} \tag{3.14}$$

The normal at point $\mathbf{r_i}$ is

$$(\mathbf{r_i}' - \mathbf{r_i}) \times \mathbf{n_i} = 0 \tag{3.15a}$$

$$\mathbf{r_i}' \times \mathbf{n_i} = \mathbf{r_i} \times \mathbf{n_i} \tag{3.15b}$$

where $\mathbf{r_i}'$ is the radius vector of a point on the normal at $\mathbf{r_i}$. Substituting equation (3.15b) into equation (3.14) produces

$$\det(\mathbf{A}) = [\mathbf{n_2 n_1 r_2'}][\mathbf{r_1}'\mathbf{r_3}'\mathbf{n_3}] - [\mathbf{r_1}'\mathbf{r_2}'\mathbf{n_2}][\mathbf{n_3 n_1 r_3}'] \tag{3.16}$$

Since $\mathbf{r_i'}$ can be any point on the normal at $\mathbf{r_i}$, if $\det(\mathbf{A}) = 0$, then one of the following conditions has to be satisfied

1. $[\mathbf{r_1' r_2' n_2}] = [\mathbf{r_1' r_3' n_3}] = 0$
2. $[\mathbf{n_3 n_1 r_3'}] = [\mathbf{n_2 n_1 r_2'}] = 0$
3. $[\mathbf{r_1' r_2' n_2}] = [\mathbf{n_2 n_1 r_2'}] = 0$
4. $[\mathbf{n_3 n_1 r_3'}] = [\mathbf{r_1' r_3' n_3}] = 0$

Condition 1 requires that $\mathbf{r_i}', \mathbf{n_i}, i = 1,2,3$ are coplanar if we let normal vectors start at the origin, or $\mathbf{r_i}', i = 1,2,3$ are collinear. Remembering that $\mathbf{r_i}'$ is the radius vector starting from the origin and ending at the normal of point $\mathbf{r_i}$, we can see that there exist only three cases when $\mathbf{r_i}'$ are coplanar, see Figure 3.5(a), Figure 3.5(b) and Figure 3.5(c), that is, three normals intercept with xy or xz or yz plane, or at a common point, or with a straight line. Since $\mathbf{n_i}$ has to be coplanar with $\mathbf{r_i}'$, therefore, for cases shown in Figure 3.5(a) and Figure 3.5(c), three normals have to be on the plane which is impossible for a 3D surface. The case shown in Figure 3.5(b) is a special case that $\mathbf{r_i}'$ are collinear. Therefore, we can conclude that if condition 1 holds, then the $\mathbf{r_i}'$ have to be collinear,

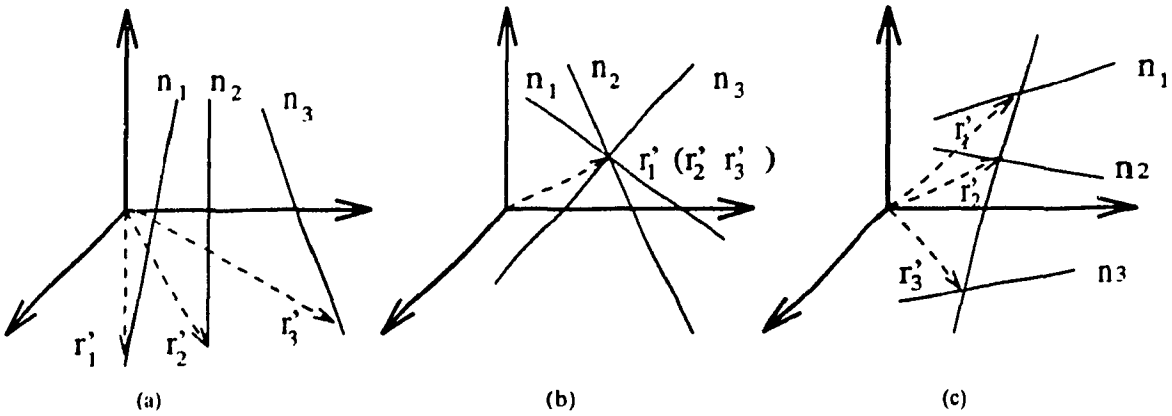


Figure 3.5: Three cases when $\mathbf{r_i}'$ are coplanar.

which means that three normals intercept either at a common point or with a straight line passing through the origin. Since $\mathbf{r_i}, i = 1,2,3$ are any combinations of three points on the surface, the conclusion has to be applied to all normals, otherwise, we can always find one combination to violate the condition. So far, we have proved that if $\det(\mathbf{A}) = 0$, then all normals to a surface either intercept at a common point or intercept with a straight line passing through the origin. Likewise, we can see that if condition 2 is satisfied, then the normals have to be parallel each other. Conditions 3 and 4 are actually the same if we consider that $\mathbf{r_1}, \mathbf{r_2}$ and $\mathbf{r_3}$ are any combination of three points on a surface. They require $\mathbf{r_i}', \mathbf{n_i}, i = 1,2,3$ are coplanar or normal passing through the origin, which is a special case that all normals intercept at a common point. All the statements above imply the necessary condition for $\det(\mathbf{A}) = 0$.

The sufficient conditions are easy to prove: if all normals are parallel to each other, then $[\mathbf{n_3 n_1 r_3'}] = [\mathbf{n_2 n_1 r_2'}] = 0$, and it is obvious from equation (3.14) that $\det(\mathbf{A}) = 0$.

If all normals intercept at a common point $\mathbf{r_t}$, then, taking $\mathbf{r_1}' = \mathbf{r_2}' = \mathbf{r_t}$, we have $[\mathbf{r_t r_t n_2}] = [\mathbf{r_t r_t n_3}] = 0$, then $\det(\mathbf{A}) = 0$.

If all normals intercept with a given line passing through the origin, assuming the direction vector of the line is l, and taking intercepting point as $\mathbf{r_i}'$, then $\mathbf{r_i}' = \lambda_i \mathbf{l}$, where λ_i is constant, then $[\mathbf{r_1' r_2' n_2}] = \lambda_1\lambda_2[\mathbf{l l n_2}] = 0$ and $[\mathbf{r_1' r_2' n_3}] = 0$, hence, $\det(\mathbf{A}) = 0$. Q.E.D.

In fact, if all the normals to a surface are concurrent, then the surface is a sphere or part of it. If all normals to a surface intercept with a given straight line, then it is a surface of revolution [38]. Hence, the following corollary can be obtained.

Corollary 2 *Rotation can not be determined if and only if a surface is a plane or a sphere, or a surface of revolution with its axis of revolution passing through the origin.*

3.4.3 General Rigid Motion

Since pure translation and pure rotation are two kinds of special motion of general rigid motion, the necessary conditions for general rigid motion have to include necessary conditions for pure translation and pure rotation. Now let us consider sufficient conditions for general rigid motion. Rigid motion parameters can always be obtained by the following iteration method: first assuming that translation is known (say $\mathbf{v}_0 = \mathbf{0}$), then from equation (3.5) the initial estimate ω_0 for rotation can be obtained by solving

$$[\mathbf{rn}\omega] = -\mathbf{n} \bullet \mathbf{v}_0 - F_t \tag{3.17}$$

This intitial estimate ω_0 can be used to obtain the new estimate for translation $\mathbf{v}_1$ by solving

$$\mathbf{n} \bullet \mathbf{v} = -[\mathbf{rn}\omega_0] - F_t \tag{3.18}$$

Repeat this procedure until a satisfactory result is achieved for translation and rotation. Therefore, as long as sufficient conditions for both translation and rotation are satisfied, the estimate for rigid motion parameters $\mathbf{v}$ and ω can be always found. Hence, the following conclusions can be drawn:

Lemma 3 *Rigid motion cannot be determined if and only if the surface in a scene is a plane, or a cylindrical surface, or a sphere, or a surface of revolution with its axis of revolution passing through the origin.*

In fact, for surfaces discussed above, it is impossible to perceive rigid motion based only on shape information, even by the human vision system. Other information such as intensity has to be incorporated with shape information in order to discern the motion.

3.5 Error Analysis

The theory of the existence of a solution does not provide any information about the reliability of the solution. In order to obtain a solution, gradient values have to be estimated from discrete images, which will be inaccurate due to the noise created during imaging process, quantization and sampling procedure. The inaccuracy in gradients and coordinates results in errors in the coefficients of the linear system for motion parameters. If these small errors in the coefficients of a linear system lead to totally incorrect results, then the linear system is very ill-conditioned. In other words, the system is very sensitive to small perturbations in the coefficients.

In this section, the factors that contribute to the gradient measurement errors and the factors that determine the conditioning of the linear system will be examined. The measurement errors in coordinates (x, y, z) are determined by the range scanner itself. Usually, it depends on how far the point is from the camera, the resolution of CCD position sensor, the accuracy of the rotating mirror, quantization errors etc. (for details, see [27]). The work in this section is inspired by Kearney and Thompson [66].

3.5.1 Gradient Measurement Error

The estimates of the gradients F_x, F_y, F_t will be corrupted by errors in the depth measurements, and inaccuracies introduced by sampling the surface discretely in time and space.

The errors in the depths are random and result from a variety of sources as we mentioned above. Usually, noise is proportional to the depth value, that is, the further the object is from the camera, the more noisy is the measured depth. It is assumed that the depth error is approximately additive and independent among neighboring pixels. The gradients, estimated from changes in the depth measurements, will contain a component of random error which is distributed like the error in the depth function. The random component of the gradient error will be additive and independent of the magnitude of the gradient to the extent that the depth noise is additive.

The depth function is sampled discretely in time and space. This will introduce a

systematic measurement error in the estimates $\tilde{F}_x, \tilde{F}_y, \tilde{F}_t$ of the gradients.

The gradient sampling error depends on the second and higher order derivatives of the surface. To examine the sampling error in $\tilde{F}_x$, the depth function evaluated at $(x + \Delta x, y, t)$ around the point (x, y, t) is expanded as

$$F(x + \Delta x, y, t) = F(x, y, t) + F_x \Delta x + \frac{1}{2} F_{xx} \Delta^2 x + O(\Delta^3 x) \tag{3.19}$$

where F_x, F_{xx} are the first and second order partial derivatives of the depth in the x direction evaluated at (x, y, t).

Rearranging terms, an estimate is obtained for the gradient in the x direction

$$\begin{aligned} \tilde{F}_x &= \frac{F(x + \Delta x, y, t) - F(x, y, t)}{\Delta x} \\ &= F_x + \frac{1}{2} F_{xx} \Delta x + O(\Delta^2 x) \end{aligned} \tag{3.20}$$

The sampling error ϵ_{F_x} is defined as $\tilde{F}_x - F_x$, the difference between the computed and true values. From equation (3.20), we obtain the approximate relationship

$$\epsilon_{F_x} \simeq \frac{1}{2} F_{xx} \Delta x \tag{3.21}$$

Similarly, the sampling error in the estimates $\tilde{F}_y$ and $\tilde{F}_t$ are approximated by

$$\epsilon_{F_y} \simeq \frac{1}{2} F_{yy} \Delta y \tag{3.22}$$

$$\epsilon_{F_t} \simeq \frac{1}{2} F_{tt} \Delta t \tag{3.23}$$

The sampling error of the spatial gradient depends on the spatial resolution of the camera Δx and Δy, and the second order spatial derivatives F_{xx}, F_{yy} of the surface. The sampling error for the temporal gradient ϵ_{F_t} is influenced by the frame rate Δt and the higher order derivatives of the depth function over time. In fact, ϵ_{F_t} can be expressed in terms of spatial derivatives and motion.

Differentiating the local velocity constraint equation (3.2) with respect to x, y and t respectively gives

$$-F_{xt} = F_{xx} u_1 + F_x \frac{\partial u_1}{\partial x} + F_{yx} u_2 + F_y \frac{\partial u_2}{\partial x} - \frac{\partial u_3}{\partial x} \tag{3.24a}$$

$$-F_{yt} = F_{xy} u_1 + F_x \frac{\partial u_1}{\partial y} + F_{yy} u_2 + F_y \frac{\partial u_2}{\partial y} - \frac{\partial u_3}{\partial y} \tag{3.24b}$$

$$-F_{tt} = F_{xt} u_1 + F_x \frac{\partial u_1}{\partial t} + F_{yt} u_2 + F_y \frac{\partial u_2}{\partial t} - \frac{\partial u_3}{\partial t} \tag{3.24c}$$

Rewriting these formulas produces

$$-F_{xt} = \begin{bmatrix} F_{xx} & F_{xy} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_2}{\partial x} & \frac{\partial u_3}{\partial x} \end{bmatrix} \begin{bmatrix} F_x \\ F_y \\ -1 \end{bmatrix} \tag{3.25a}$$

$$-F_{yt} = \begin{bmatrix} F_{xy} & F_{yy} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} \frac{\partial u_1}{\partial y} & \frac{\partial u_2}{\partial y} & \frac{\partial u_3}{\partial y} \end{bmatrix} \begin{bmatrix} F_x \\ F_y \\ -1 \end{bmatrix} \tag{3.25b}$$

$$-F_{tt} = \begin{bmatrix} F_{xt} & F_{yt} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} \frac{\partial u_1}{\partial t} & \frac{\partial u_2}{\partial t} & \frac{\partial u_3}{\partial t} \end{bmatrix} \begin{bmatrix} F_x \\ F_y \\ -1 \end{bmatrix} \tag{3.25c}$$

Substituting equation (3.25a) and equation (3.25b) into equation (3.25c), we get

$$F_{tt} = \begin{bmatrix} u_1 & u_2 \end{bmatrix} \begin{bmatrix} F_{xx} & F_{xy} \\ F_{xy} & F_{yy} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} u_1 & u_2 \end{bmatrix} \begin{bmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_2}{\partial x} & \frac{\partial u_3}{\partial x} \\ \frac{\partial u_1}{\partial y} & \frac{\partial u_2}{\partial y} & \frac{\partial u_3}{\partial y} \end{bmatrix} \begin{bmatrix} F_x \\ F_y \\ -1 \end{bmatrix}$$

$$+ \begin{bmatrix} \frac{\partial u_1}{\partial t} & \frac{\partial u_2}{\partial t} & \frac{\partial u_3}{\partial t} \end{bmatrix} \begin{bmatrix} F_x \\ F_y \\ -1 \end{bmatrix} \tag{3.26}$$

When Δt is small, motion may be considered uniform, therefore, $\frac{\partial u_1}{\partial t} = \frac{\partial u_2}{\partial t} = \frac{\partial u_3}{\partial t} = 0$, then

$$F_{tt} = \begin{bmatrix} u_1 & u_2 \end{bmatrix} \begin{bmatrix} F_{xx} & F_{xy} \\ F_{xy} & F_{yy} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} u_1 & u_2 \end{bmatrix} \begin{bmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_2}{\partial x} & \frac{\partial u_3}{\partial x} \\ \frac{\partial u_1}{\partial y} & \frac{\partial u_2}{\partial y} & \frac{\partial u_3}{\partial y} \end{bmatrix} \begin{bmatrix} F_x \\ F_y \\ -1 \end{bmatrix} \tag{3.27}$$

Now, let us first tilt the xyz coordinate system such that the z axis is parallel to the rotation axis (the direction cosines of the rotation axis are $(\frac{\omega_1}{|\omega|}, \frac{\omega_2}{|\omega|}, \frac{\omega_3}{|\omega|})$), then $\omega_1 = \omega_2 = 0$ and $\omega_3 = |\omega|$ in the new coordinate system, from equation (3.3) we have the following relation

$$u_1 = -v_1 + y|\omega| \tag{3.28a}$$

$$u_2 = -v_2 - x|\omega| \tag{3.28b}$$

$$u_3 = -v_3 \tag{3.28c}$$

Taking partial derivatives, we have

$$\begin{bmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_2}{\partial x} & \frac{\partial u_3}{\partial x} \end{bmatrix} = \begin{bmatrix} 0 & -\omega_3 & 0 \end{bmatrix} \tag{3.29a}$$

$$\begin{bmatrix} \frac{\partial u_1}{\partial y} & \frac{\partial u_2}{\partial y} & \frac{\partial u_3}{\partial y} \end{bmatrix} = \begin{bmatrix} \omega_3 & 0 & 0 \end{bmatrix} \tag{3.29b}$$

Substituting these partial derivatives into equation (3.27), we get

$$F_{tt} = \begin{bmatrix} u_1 & u_2 \end{bmatrix} \begin{bmatrix} F_{xx} & F_{xy} \\ F_{xy} & F_{yy} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} - u_1 F_y \omega_3 + u_2 F_x \omega_3 \tag{3.30}$$

Now let us rotate the coordinate system around z axis (rotation axis) again so that the x axis coincides with the direction $(u_1, u_2, 0)$, that is, the projection direction of u on the xy plane (xy plane is orthogonal to the rotation axis). This plane is called the *rotation plane*. In this new coordinate system, u_1 is the magnitude of the projection velocity of u on the rotation plane, denoted by $u_\perp$, and $u_2 = 0$, F_{tt} reduces to

$$F_{tt} = u_\perp^2 F_{xx} - u_\perp F_y |\omega| \tag{3.31}$$

It is clear that the magnitude of F_{tt} depends on the second order derivatives of the depth function along the projection direction of motion on the rotation plane, the first order derivatives in the perpendicular projection direction of motion on the rotation plane, the magnitudes of projection motion $u_\perp$ itself and the rotational velocity.

The temporal derivative will be well estimated only where the spatial depth function is nearly linear along the projection direction of motion on the *rotation plane* and is nearly constant along the perpendicular projection direction.

For pure translation,

$$F_{tt} = u_1^2 F_{xx} \tag{3.32}$$

The temporal derivative will be well estimated only if the partial derivative of the depth function is nearly linear along the projection direction of motion on the xy plane or if the motion is small.

3.5.2 Conditioning

The accuracy of the estimated motion parameters depends on the measurement errors in the gradients and the error propagation characteristics of the linear equation. When a

system of linear equations is very sensitive to small errors in the coefficients or right-hand side of equations, it is said to be ill-conditioned. The condition number of the coefficient matrix $\mathbf{A}$ is defined by

$$\kappa(\mathbf{A}) = \|\mathbf{A}\|\|\mathbf{A}^{-1}\| \tag{3.33}$$

where $\|\bullet\|$ denote norm of a matrix or a vector. The condition number roughly estimates the extent to which relative errors in the coefficients and the right-hand side magnifies the error in the estimate of the unknown. If

$$(\mathbf{A} + \epsilon\mathbf{F})\mathbf{m}(\epsilon) = \mathbf{b} + \epsilon\mathbf{f} \tag{3.34}$$

where $\epsilon\mathbf{F}$ and $\epsilon\mathbf{f}$ are noise in $\mathbf{A}$ and $\mathbf{b}$, the error in $\mathbf{m}$ caused by the noise is equal to

$$\frac{\|\mathbf{m}(\epsilon) - \mathbf{m}\|}{\|\mathbf{m}\|} \leq k(\mathbf{A})[\rho_a + \rho_b] + O(\epsilon^2) \tag{3.35}$$

where

$$\rho_a = \epsilon\frac{\|\mathbf{F}\|}{\|\mathbf{A}\|}, \qquad \rho_b = \epsilon\frac{\|\mathbf{f}\|}{\|\mathbf{b}\|}$$

represent the relative errors in $\mathbf{A}$ and $\mathbf{b}$, respectively. Thus, the relative error in $\mathbf{m}$ can be $\kappa(\mathbf{A})$ times the relative errors in $\mathbf{A}$ and $\mathbf{b}$. In this sense, the condition number $\kappa(\mathbf{A})$ quantifies the sensitivity of the $\mathbf{Am} = \mathbf{b}$ problem.

There are many ways to calculate the norm of a matrix and a vector. 2-norm for vectors and Frobenius norm for matrices will be used in this thesis. 2-norm of a vector $\mathbf{x} \in \mathcal{R}^n$ is defined as

$$\|\mathbf{x}\| = (|x_1|^2 + \cdots + |x_n|^2)^{1/2} = (\mathbf{x}^{\mathrm{T}}\mathbf{x})^{1/2}$$

Frobenius norm of a matrix $\mathbf{A} \in \mathcal{R}^{m\times n}$ is defined as

$$\|\mathbf{A}\| = \left[\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2\right]^{1/2}$$

Two kinds of special motion: pure translation and pure rotation, will be considered. As pointed out before, general rigid motion can be obtained through iteration. If the system of equations for both pure translation and rotation are well-conditioned, the

solution of general rigid motion will be not very sensitive to noise. Theoretically, three well-distributed points are enough to estimate motion for either pure translation or pure rotation. In practice, over-determined systems are used to reduce the effects of errors in the motion constraint equations. Square system will be examined.

Pure Translation

The following equation set is solved to estimate translation of an object.

$$\mathbf{A}\mathbf{v} = \mathbf{b} \tag{3.36}$$

where

$$\mathbf{A} = \begin{bmatrix} F_{x_1} & F_{y_1} & -1 \\ F_{x_2} & F_{y_2} & -1 \\ F_{x_3} & F_{y_3} & -1 \end{bmatrix} = - \begin{bmatrix} \mathbf{n_1}^T \\ \mathbf{n_2}^T \\ \mathbf{n_3}^T \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} -F_{t_1} \\ -F_{t_2} \\ -F_{t_3} \end{bmatrix}$$

where $n_i^T = [\ F_{x_i} \quad F_{y_i} \quad -1\ ]$ is the normal vector at point $\mathbf{r_i}$. It is obvious that

$$\|\mathbf{A}\| = (\|\mathbf{n_1}\|^2 + \|\mathbf{n_2}\|^2 + \|\mathbf{n_3}\|^2)^{1/2}$$

The calculation of $\|\mathbf{A}^{-1}\|$ is more complicated than that of $\|\mathbf{A}\|$. Using theorem A.2 given in Appendix A, the Frobenius norm of $\mathbf{A}^{-1}$ is

$$\|\mathbf{A}^{-1}\| = \left[\frac{\|\mathbf{n_2} \times \mathbf{n_3}\|^2}{(\mathbf{n_2} \times \mathbf{n_3} \bullet \mathbf{n_1})^2} + \frac{\|\mathbf{n_3} \times \mathbf{n_1}\|^2}{(\mathbf{n_3} \times \mathbf{n_1} \bullet \mathbf{n_2})^2} + \frac{\|\mathbf{n_1} \times \mathbf{n_2}\|^2}{(\mathbf{n_1} \times \mathbf{n_2} \bullet \mathbf{n_3})^2} \right]^{1/2} \tag{3.37}$$

Let θ_1 be the angle between vectors $\mathbf{n_2} \times \mathbf{n_3}$ and $\mathbf{n_1}$, θ_2 be the angle between vectors $\mathbf{n_3} \times \mathbf{n_1}$ and $\mathbf{n_2}$ and θ_3 be the angle between vectors $\mathbf{n_1} \times \mathbf{n_2}$ and $\mathbf{n_3}$, then

$$\|\mathbf{A}^{-1}\| = \left[\frac{1}{\|\mathbf{n_1}\|^2 \cos^2\theta_1} + \frac{1}{\|\mathbf{n_2}\|^2 \cos^2\theta_2} + \frac{1}{\|\mathbf{n_3}\|^2 \cos^2\theta_3} \right]^{1/2} \tag{3.38}$$

The condition number $\kappa(\mathbf{A})$

$$\kappa(\mathbf{A}) = \left[(\|\mathbf{n_1}\|^2 + \|\mathbf{n_2}\|^2 + \|\mathbf{n_3}\|^2) \left(\frac{1}{\|\mathbf{n_1}\|^2 \cos^2\theta_1} + \frac{1}{\|\mathbf{n_2}\|^2 \cos^2\theta_2} + \frac{1}{\|\mathbf{n_3}\|^2 \cos^2\theta_3} \right) \right]^{1/2} \tag{3.39}$$

As pointed out from previous discussions, the error in the solution of $\mathbf{Ax} = \mathbf{b}$ caused by errors in the coefficients $\mathbf{A}$ and $\mathbf{b}$ is proportional to $\kappa(\mathbf{A})$, therefore, it is desirable that $\kappa(\mathbf{A})$ is minimized. It is easy to prove the following lemma.

Lemma 4 *If three points* $\mathbf{r_1}, \mathbf{r_2}$ *and* $\mathbf{r_3}$ *are choosen such that their normal vectors to the surface* $\mathbf{n_1}, \mathbf{n_2}$ *and* $\mathbf{n_3}$ *are orthogonal to each other and* $\|\mathbf{n_1}\| = \|\mathbf{n_2}\| = \|\mathbf{n_3}\|$, *then* $\kappa(\mathbf{A})$ *reaches its minimum value.*

Proof:

Since $\kappa(\mathbf{A})$ and $\kappa^2(\mathbf{A})$ reach their minima at the same point, $\kappa^2(\mathbf{A})$ is considered. Taking partial derivatives of $\kappa^2(\mathbf{A})$ with respect to θ_1, θ_2, θ_3, $\|\mathbf{n_1}\|$, $\|\mathbf{n_2}\|$ and $\|\mathbf{n_3}\|$ respectively, and set them to zero

$$\frac{\partial \kappa^2(\mathbf{A})}{\partial \theta_1} = -\frac{c_1 \sin\theta_1}{\|n_1\|^2 \cos^3\theta_1} = 0 \tag{3.40}$$

$$\frac{\partial \kappa^2(\mathbf{A})}{\partial \theta_2} = -\frac{c_1 \sin\theta_2}{\|n_2\|^2 \cos^3\theta_2} = 0 \tag{3.41}$$

$$\frac{\partial \kappa^2(\mathbf{A})}{\partial \theta_3} = -\frac{c_1 \sin\theta_3}{\|n_3\|^2 \cos^3\theta_3} = 0 \tag{3.42}$$

$$\frac{\partial \kappa^2(\mathbf{A})}{\partial \|n_1\|^2} = -2\|\mathbf{n_1}\|c_2 - \frac{c_1}{\|n_1\|^3 \cos^2\theta_1} = 0 \tag{3.43}$$

$$\frac{\partial \kappa^2(\mathbf{A})}{\partial \|n_2\|^2} = -2\|\mathbf{n_2}\|c_2 - \frac{c_1}{\|n_2\|^3 \cos^2\theta_2} = 0 \tag{3.44}$$

$$\frac{\partial \kappa^2(\mathbf{A})}{\partial \|n_3\|^2} = -2\|\mathbf{n_3}\|c_2 - \frac{c_1}{\|n_3\|^3 \cos^2\theta_3} = 0 \tag{3.45}$$

where

$$c_1 = \|\mathbf{n_1}\|^2 + \|\mathbf{n_2}\|^2 + \|\mathbf{n_3}\|^2$$

$$c_2 = \frac{1}{\|\mathbf{n_1}\|^2 \cos^2\theta_1} + \frac{1}{\|\mathbf{n_2}\|^2 \cos^2\theta_2} + \frac{1}{\|\mathbf{n_3}\|^2 \cos^2\theta_3}$$

From equations (3.40, 3.41, 3.42),

$$\theta_1 = \theta_2 = \theta_3 = 0$$

Since θ_1 is the angle between $\mathbf{n_2} \times \mathbf{n_3}$ and $\mathbf{n_1}$, $\mathbf{n_1}$ is parallel to $\mathbf{n_2} \times \mathbf{n_3}$ when $\theta_1 = 0$. Hence, $\mathbf{n_1} \perp \mathbf{n_2}$ and $\mathbf{n_1} \perp \mathbf{n_3}$. Similarly, from $\theta_2 = \theta_3 = 0$, we can conclude that $\mathbf{n_1}, \mathbf{n_2}$ and $\mathbf{n_3}$ are orthogonal to each other.

Substituting $\theta_1 = \theta_2 = \theta_3 = 0$ in equations (3.43, 3.44, 3.45), and rearranging them produces

$$\|\mathbf{n_1}\|^4 c_2 - c_1 = 0 \tag{3.46}$$

$$\|\mathbf{n_2}\|^4 c_2 - c_1 = 0 \tag{3.47}$$

$$\|\mathbf{n_3}\|^4 c_2 - c_1 = 0 \tag{3.48}$$

We can find that

$$\|\mathbf{n_1}\| = \|\mathbf{n_2}\| = \|\mathbf{n_3}\|$$

Q.E.D.

Pure Rotation

Rotation can be obtained by solving

$$\mathbf{A}\omega = \mathbf{b} \tag{3.49}$$

where

$$\mathbf{A} = -\begin{bmatrix} (\mathbf{r_1} \times \mathbf{n_1})^T \\ (\mathbf{r_2} \times \mathbf{n_2})^T \\ (\mathbf{r_3} \times \mathbf{n_3})^T \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} -E_{t_1} \\ -E_{t_2} \\ -E_{t_3} \end{bmatrix}$$

Similar to the derivation of equation (3.39), the condition number for pure rotation is

$$\kappa(\mathbf{A}) = \left[\left(\|(\mathbf{r_1} \times \mathbf{n_1})\|^2 + \|(\mathbf{r_2} \times \mathbf{n_2})\|^2 + \|(\mathbf{r_3} \times \mathbf{n_3})\|^2 \right) \left(\frac{1}{\|\mathbf{r_1} \times \mathbf{n_1}\|^2 \cos^2 \phi_1} + \frac{1}{\|\mathbf{r_2} \times \mathbf{n_2}\|^2 \cos^2 \phi_2} + \frac{1}{\|\mathbf{r_3} \times \mathbf{n_3}\|^2 \cos^2 \phi_3} \right) \right]^{1/2} \tag{3.50}$$

where ϕ_1 is the angle between vectors $(\mathbf{r_2} \times \mathbf{n_2}) \times (\mathbf{r_3} \times \mathbf{n_3})$ and $(\mathbf{r_1} \times \mathbf{n_1})$, ϕ_2 is the angle between vectors $(\mathbf{r_3} \times \mathbf{n_3}) \times (\mathbf{r_1} \times \mathbf{n_1})$ and $(\mathbf{r_2} \times \mathbf{n_2})$, ϕ_3 is the angle between vectors $(\mathbf{r_1} \times \mathbf{n_1}) \times (\mathbf{r_2} \times \mathbf{n_2})$ and $(\mathbf{r_3} \times \mathbf{n_3})$.

According to Lemma 4, when $(\mathbf{r_1} \times \mathbf{n_1})$,$(\mathbf{r_2} \times \mathbf{n_2})$ and $(\mathbf{r_3} \times \mathbf{n_3})$ are mutually orthogonal and $\|\mathbf{r_1} \times \mathbf{n_1}\| = \|\mathbf{r_2} \times \mathbf{n_2}\| = \|\mathbf{r_3} \times \mathbf{n_3}\|$, $\kappa(A)$ is minimized. As shown in Figure 3.6, in order to reach minimum condition number, planes P_1, P_2 and P_3 should be orthogonal to each other, where plane P_i is constructed by vectors $\mathbf{r_i}$ and $\mathbf{n_i}$. This requirement implies that $\mathbf{r_i}$ should be orthogonal to each other, which means the field of view has to be very large. For a narrow viewing angle, $\mathbf{r_i}$ tend to be parallel to each other. Therefore, for small objects or objects far away from the observer, the estimation of rotation will be very sensitive to noise in the data.

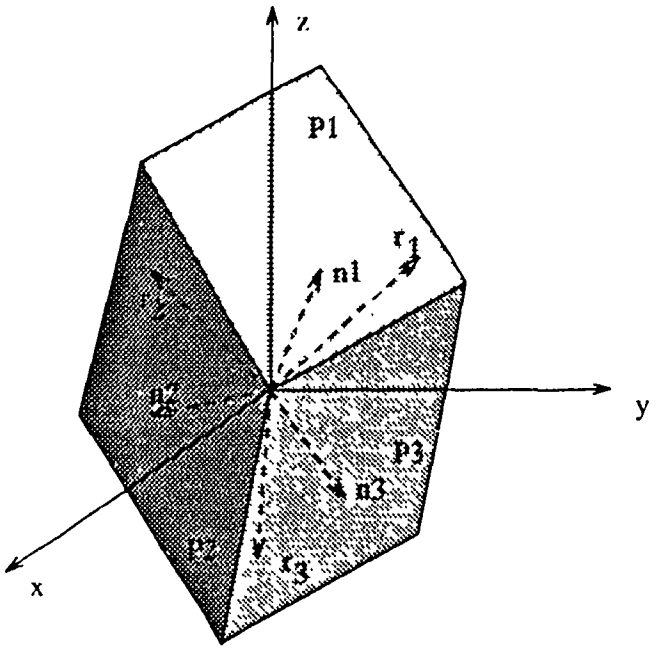


Figure 3.6: When three planes P_1, P_2 and P_3 are orthogonal to each other, the problem tend to be well-conditioned, where plane P_i is constructed by two vectors $\mathbf{r}_i$ ans $\mathbf{n}_i$.

3.6 Implementation

Standard LS usually works well for $\mathbf{Am} = \mathbf{b} + \delta\mathbf{b}$ when

1. The components $\delta\mathbf{b}$ are independent Gaussian noise with the same variance.

2. $\mathbf{A}$ is known without error.

Obviously, both conditions may be grossly violated in our problem. In this section, the problem of how to deal with these violations will be addressed.

3.6.1 Weighted Least Squares Method

It has been assumed that surfaces are differentiable in order to obtain the local velocity constraint equation. This assumption will often be violated for most of real world images. In fact, most range data contain several kinds of discontinuities such as jump edges (z is discontinuous) and roof edges (z is continuous but the first derivative is discontinuous) [61]. Although it may be justified that the local velocity constraint equation will still hold when an image contains discontinuities, as Schunck has proved for the brightness change

constraint equation [91], the first order partial derivatives still have to be approximated by differences. As shown before, the errors in the estimates will be proportional to the second order derivatives.

The first order approximation may result in large errors near jump edges, roof edges and high curvature points. These points are called *break points*, which simply means that they break the local velocity constraint equation. This kind of violation will contribute to $\delta\mathbf{b}$, therefore, the components of $\delta\mathbf{b}$ will have different variances.

Obviously, one possible way to avoid this problem is to locate the *break points* first and then remove them from further consideration. The problem of locating *break points* has been addressed by many researchers [61]. Unfortunately, there is no simple way to do it. Most of the methods require computation of the curvature, which is also very sensitive to noise. On the other hand, our purpose is not to precisely locate these points, but instead to reduce the effects of these points on the solution of equation (3.7). This suggests that a weighted least squares (WLS) method may be a better choice. A weight is assigned to each point in the image. *Break points* are given smaller weights than other points. Now the problem is how to determine these weights. Obviously if the tangent plane at a *break point* is fitted in its neighborhood, it will result in large fitting errors in that neighborhood, therefore, a point can be simply assigned a weight which is inversely proportional to the variance of fitting errors in the neighborhood of this point. That is,

$$w(x,y) = \begin{cases} \frac{1}{\lambda+\sigma^2(x,y)} & if\ \sigma^2(x,y) \le thvar \\ 0 & otherwise \end{cases} \tag{3.51}$$

where λ is a constant preventing $w(x,y)$ from becoming infinite when $\sigma^2(x,y)$ equals zero, *thvar* is a predefined threshold used to remove points on jump edges from further consideration by assigning it a zero weight, and $\sigma^2(x,y)$ is a fitting variance of point (x,y)

$$\sigma^2(x,y) = \sum_{x_i \in S} \sum_{y_j \in S} [\epsilon(x_i,y_j) - \epsilon(x,y)]^2 / N_s$$

where S is a predefined neighborhood of point (x,y), N_s is the number of pixels in the neighborhood, and $\epsilon(x_i,y_j)$ is the fitting error of point (x_i,y_j) defined as

$$\epsilon(x_i,y_j) = F(x_i,y_j,t) - [F(x,y,t) + F_x * (x - x_i) + F_y * (y - y_j)]$$

where $\epsilon(x, y)$ is the mean error in the neighborhood,

$$\epsilon(x, y) = \sum_{x_i \in S} \sum_{y_j \in S} \epsilon(x_i, y_j)/N_s$$

WLS method can be used to find the solution which minimizes

$$(\mathbf{Am} - \mathbf{b})^T \mathbf{W} (\mathbf{Am} - \mathbf{b})$$

where $\mathbf{W} = diag(w(x_1, y_1), w(x_2, y_2), ..., w(x_n, y_n))$ is the weight matrix. If $\mathbf{A}^T\mathbf{WA}$ is nonsingular, then the solution is

$$\mathbf{m} = (\mathbf{A}^T\mathbf{W}^2\mathbf{A})^{-1}\mathbf{A}^T\mathbf{W}^2\mathbf{b} \tag{3.52}$$

If $\mathbf{A}^T\mathbf{W}^2\mathbf{A}$ is singular, then using SVD, the solution is

$$\mathbf{m} = (\mathbf{WA})^+\mathbf{Wb} \tag{3.53}$$

As shown in Figures 3.15 and 3.18 in section 3.7, the weights chosen as in the above are smaller around those *break points* , hence the WLS method is expected to give better results than the LS method.

Choosing the variance of fitting errors instead of the sum of fitting errors is based on the following intuitive arguments. Suppose that the weight for a point P is estimated in terms of the fitting errors in the neighborhood S (see Figure 3.7(a)), if some of the neighboring points (for examples, P_i, P_j) of P happen to be subject to a large random noise, then the sum of the fitting errors in S will be very large, while the variance of the fitting errors tend to be small if the majority of neighboring points are not very noisy, therefore, a large weight will be assigned to point P by equation (3.51) as it should be. In this sense, the variance of fitting errors is less sensitive to random noise.

Another reason to choose the variance is that *thvar* is much easier to set. As shown in Figure 3.7(b), both P_1 and P_2 are two points on jump edges, which should be discarded from further consideration if *thvar* is set properly. The sum of the fitting errors is much more sensitive to the height of a jump edge than is the variance of the fitting errors. Therefore, it is harder to choose a proper *thvar* in order to single out both P_1 and P_2 if we use the sum of fitting errors. It has been found through experiments that *thvar*

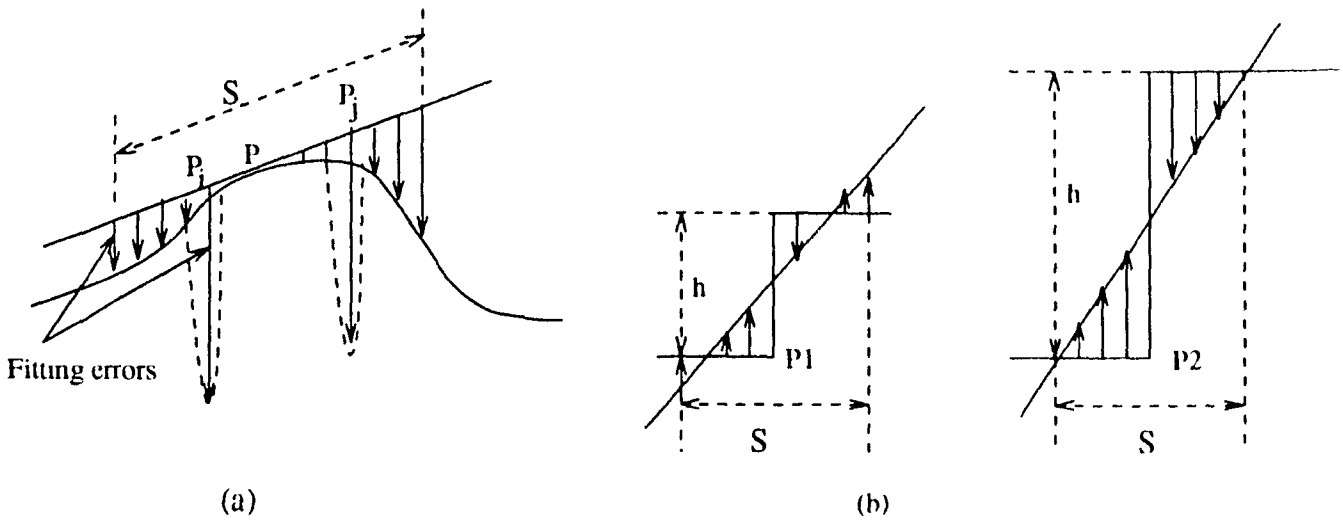


Figure 3.7: (a) Sum of fitting errors is more sensitive to random noise than variance of fitting errors. (b) Sum of fitting errors is more sensitive to the height of a jump edge than variance of fitting errors.

can change within a relatively large range without dramatically affecting the results if the variance is used, and a fixed *thvar* works very well for different kinds of images in the shown experiments.

3.6.2 Total Least Squares Method

The requirement that **A** is known without error is usually not held because the elements of **A** are functions of the first order partial derivatives, which are estimated from the noisy depth values. In this case, the total least squares (TLS) method as provided in [10, 39] may be used. The application of this method in motion analysis has been investigated in [23] where the author concluded that the TLS is much better than the LS. We here show how to solve our equations by the TLS method.

Assume that both **A** and **b** be noisy, equation (3.6) would be given by

$$(\mathbf{A} + \delta\mathbf{A})\mathbf{m} = (\mathbf{b} + \delta\mathbf{b}) \tag{3.54}$$

where $\delta\mathbf{A}$ and $\delta\mathbf{b}$ are noise. The TLS solution is given by minimizing

$$e_{TLS} = \|\delta\mathbf{A}\|^2 + \|\delta\mathbf{b}\|^2 \tag{3.55}$$

subject to $\mathbf{b} + \delta\mathbf{b} \in \Re(\mathbf{A} + \delta\mathbf{A})$, where $\Re(\bullet)$ denotes the range space of the argument. Equation (3.54) can be expressed as

$$[\mathbf{A} + \delta\mathbf{A}, \mathbf{b} + \delta\mathbf{b}] \begin{bmatrix} m \\ -1 \end{bmatrix} = 0 \tag{3.56}$$

where $[\mathbf{A} + \delta\mathbf{A}, \mathbf{b} + \delta\mathbf{b}]$ is an augmented matrix representation with dimension $n \times 7$. $\begin{bmatrix} \mathbf{m} \\ -1 \end{bmatrix}$ is a vector of length 7. Using the SVD, the augmented matrix can be decomposed as

$$[\mathbf{A}, \mathbf{b}] = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^{\mathbf{T}} \tag{3.57}$$

where $\mathbf{U} \in \mathcal{R}^{n\times 7}, \mathbf{V} \in \mathcal{R}^{7\times 7}$ are orthogonal matrices and $\mathbf{\Lambda} \in \mathcal{R}^{7\times 7}$ is a diagonal matrix ($\mathbf{\Lambda} = diag(\sigma_1, \sigma_2, \cdots, \sigma_7)$). In the noise free case, i.e. $\delta\mathbf{A} = 0, \delta\mathbf{b} = 0$, $\mathbf{b}$ is in the range space of $\mathbf{A}$, where only six of the seven singular values are nonzero. Due to the presence of noise, $rank[\mathbf{A}, \mathbf{b}] = 7$. Thus the dimensionality of the null space of $[\mathbf{A}, \mathbf{b}]$ is zero, and no exact solution for equation (3.56) exists. A rank-6 approximation for the augmented matrix is taken, i.e.,

$$[\tilde{\mathbf{A}}, \tilde{\mathbf{b}}] = \mathbf{U}\tilde{\mathbf{\Lambda}}\mathbf{V}^{\mathbf{T}} \tag{3.58}$$

where the smallest singular value of $\mathbf{\Lambda}$ has been set to zero in $\tilde{\mathbf{\Lambda}}$. The solution $\tilde{\mathbf{m}}_{\mathbf{TLS}}$ can be obtained from $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{b}}$ since $\tilde{\mathbf{b}}$ is in the range space of $\tilde{\mathbf{A}}$. The solution given in [39] by partitioning $\mathbf{V}$ as

$$\mathbf{V} = \begin{bmatrix} \mathbf{V_{11}} & \mathbf{V_{12}} \\ \mathbf{V_{21}} & \mathbf{V_{22}} \end{bmatrix}$$

where $\mathbf{V_{11}} \in \mathcal{R}^{6\times 6}, \mathbf{V_{12}} \in \mathcal{R}^{6\times 1}, \mathbf{V_{21}} \in \mathcal{R}^{1\times 6}, \mathbf{V_{22}} \in \mathcal{R}^{1\times 1}$, then

$$\tilde{\mathbf{\Lambda}} = -\mathbf{V_{12}}/\mathbf{V_{22}} \tag{3.59}$$

It has been proved [39] that if $\sigma_6 > \sigma_7$, then $\mathbf{V_{22}}$ will be guaranteed nonzero. For the case where this condition is not satisfied, the solution is nonunique, but a "minimum residue norm" solution may still be obtained as in the ordinary LS (pseudo-inverse) (see [39, 40] for details).

3.6.3 Normalizing the Coefficients

As discussed in section 3.5.2, the effect of $\delta\mathbf{A}$ and $\delta\mathbf{b}$ on the solution is proportional to the condition number $\kappa(\mathbf{A})$. If $\kappa(\mathbf{A})$ is small, $\delta\mathbf{A}$ and $\delta\mathbf{b}$ lead to small error in m. One of the conditions to minimize $\kappa(\mathbf{A})$ is that $\|\mathbf{a_i}\| = \|\mathbf{a_j}\|$, where $\mathbf{a_i}$ and $\mathbf{a_j}$ are i^{th} and j^{th} rows of $\mathbf{A}$. Therefore, it is better to normalize $\mathbf{a_i}$ before solving the set of linear equations. Equation (3.6) is rewritten as

$$\bar{\mathbf{A}}\mathbf{m} = \bar{\mathbf{b}} \tag{3.60}$$

where the i^{th} row $\bar{\mathbf{a}}_\mathbf{i}$ of $\bar{\mathbf{A}}$ and the i^{th} row $\bar{\mathbf{b}}_\mathbf{i}$ of $\mathbf{b}$ are

$$\bar{\mathbf{a}}_\mathbf{i} = \frac{\mathbf{a_i}}{(1 + F_{x_i}^2 + F_{y_i}^2 + (F_{y_i}z_i + y_i)^2 + (F_{x_i}z_i + x_i)^2 + (F_{y_i}x_i - F_{x_i}y_i)^2)^{1/2}}$$

$$\bar{b}_i = \frac{-F_{t_i}}{(1 + F_{x_i}^2 + F_{y_i}^2 + (F_{y_i}z_i + y_i)^2 + (F_{x_i}z_i + x_i)^2 + (F_{y_i}x_i - F_{x_i}y_i)^2)^{1/2}}$$

3.7 Simulation and Results

To verify the proposed theory and study the performance of different linear least squares methods in presence of noise, simulation experiments on both synthetic and real data have been performed. All experiments were carried out on the VAX-6410 using the C programming language and the IMSL library.

3.7.1 Experiment 1: Performances of the Various Least Squares against Noise

The goal of this experiment is to study the performances of the different least squares methods against noise. Eight algorithms are implemented, i.e., ordinary least squares (LS), weighted least squares (WLS), normalized least squares (NLS), normalized weighted least squares (NWLS), ordinary total least squares (TLS), weighted total least squares (WTLS), normalized total least squares (NTLS) and normalized weighted total least squares (NWTLS), where normalized least squares methods refer to least squares solutions of equation (3.60).

Spatial and temporal derivatives are found using the operator given by Horn and Schunck [54]: The 3D rigid motion is accomplished by a 3D motion simulation program. The details of the algorithm is found in Appendix C.

The test was done on a synthetic range image shown in Figure 3.8. The object in Figure 3.8 is defined by the equation:

$$z = \begin{cases} \sqrt{c^2 - x^2 - y^2} & if\ x < 0, (c^2 - x^2 - y^2) > 0 \\ c\sqrt{1 - \frac{x^2}{a^2} + \frac{y^2}{b^2}} = 0 & if\ x \geq 0, (1 - \frac{x^2}{a^2} + \frac{y^2}{b^2}) > 0 \\ 0 & otherwise \end{cases}$$

where $a = 52, b = 26, c = 52$. The defined object was moved to the center of the image and the image size was 128×128. This object was chosen because it contains jump edges, roof edges and corner points which violate the smooth surface assumption.

The inter-frame motion parameters are $\mathbf{t} = [\ t_x \quad t_y \quad t_z\ ]^T = [\ 1 \quad 1 \quad 1\ ]^T/pixel$, $\mathbf{n} = [\ n_x \quad n_y \quad n_z\ ]^T = [\ 0.5 \quad 0.5 \quad 0.7\ ]^T$, $\theta = 1^o$, where $\mathbf{t}$ is translation vector, $\mathbf{n}$ is rotation axis and θ is the rotation angle. Rotation is around the axis through the center point of the image.

Since the noise in range data taken by the range finder is proportional to the distance between the sensor and the point being measured, uniform noise, whose range was a certain percentage of the considered z to each point was added to the images. Let NSR denote the measure of added random noise. If a point has a depth z, then noisy $\tilde{z}$ is

$$\tilde{z} = z + NSR * \delta * z$$

where δ is a sample of an uniform random variable whose probability distribution is $(-1, 1)$. For different NSR, rigid motion parameters were estimated by different least squares methods. *thear* is fixed at 3.0. The set of translational and angular velocity parameters is converted to translation vector $\tilde{\mathbf{t}}$, rotation axis $\tilde{\mathbf{n}}$ and rotation angle $\tilde{\theta}$. The accuracy of estimated motion was measured by three error measures: ϵ_t, ϵ_n and ϵ_θ, which were defined as

$$\begin{aligned} \epsilon_t &= \frac{\|\mathbf{t} - \tilde{\mathbf{t}}\|}{\|\mathbf{t}\|} \\ \epsilon_n &= cos^{-1}(\mathbf{n} \bullet \tilde{\mathbf{n}}) \\ \epsilon_\theta &= \frac{|\theta - \tilde{\theta}|}{|\theta|} \end{aligned}$$

where ϵ_t measures the relative estimation error in the translation, ϵ_n measures how well the estimated rotation a is coincides with the real rotation axis, and ϵ_θ measures the relative estimation error in the rotation angle. The estimated errors ϵ_t, ϵ_n and ϵ_θ are plotted against the noise NSR in Figure 3.9, Figure 3.10 and Figure 3.11 for different least squares methods.

From these plots, it is surprisingly noted that the total least squares method has comparable performance for images with small noise, but definitely much worse performance for images with large noise compared with the least squares method. This is due to the fact that when the data is very noisy, such that any singular value associated with the noise subspace exceeds the corresponding singular value associated with the single subspace, the smallest singular value is comparable to other singular values, thus, the obtained TLS solution by setting this singular value to zero is no longer a good estimate. This is known as the threshold break. In our case, even for very small random noise, there still exist large systematic noise introduced by the 3D motion program, estimating errors in gradients etc.. TLS does not provide better performance than LS.

The errors against noise for the LS, WLS, NLS, and WNLS methods are replotted in Figure 3.12, Figure 3.13 and Figure 3.14 for a better visualization. Several conclusions may be drawn from these plots.

1. The WLS, NLS and NWLS methods give much better performance than the LS method.

2. When the noise is small, the WLS and NWLS methods are slightly better than NLS method, while for large noise, they are comparable.

3. The WLS and NWLS methods are comparable and have the best performance.

Conclusion 1 is expected. For conclusion 2, when random noise is small, *break points* in the image have stronger effect on the solution than random noise, the WLS and NWLS methods reduce the effect of *break points*, thus, they show better performance than the NLS method. For large noise, the *break points* are no longer dominant, therefore, the WLS, NWLS and NLS methods become comparable. As far as conclusion 3 is concerned, errors

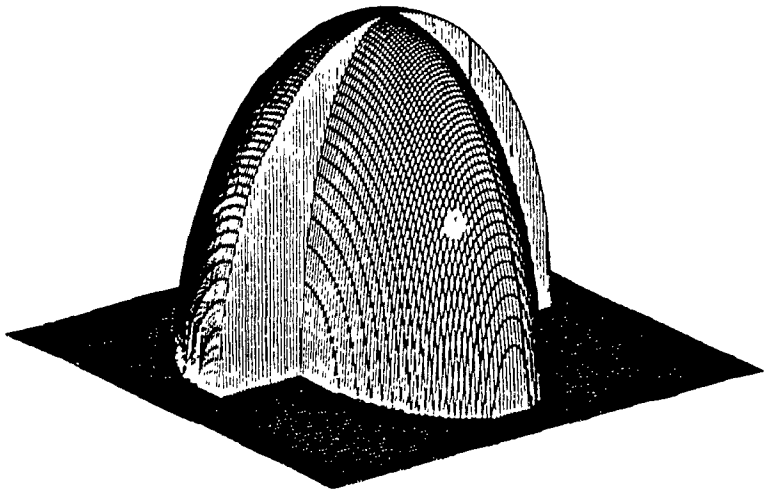


Figure 3.8: A synthetic range image.

in the solution are also proportional to errors in **A** and **b**. If weights are chosen properly, they equivalently reduce the errors in **A** and **b**, therefore, decreasing the condition number of **WA** through normalization does not have much effect on the solution, hence the WLS and the NWLS methods are comparable.

Weights and variances of fitting errors are shown in Figures 3.15 and 3.16. It is clear that the weights are small and the variances are large around the *break points*.

3.7.2 Experiment 2: Estimating Different Motion

The second experiment was done on a quasi-real data. The first frame of the sequence was taken using the laser finder at the National Research Council of Canada (NRC). The image is shown in Figure 3.17. The size of the image is reduced to 128×128. The second image was generated by the 3D movement simulation program. The purpose of this experiment is to get the performance of the algorithm for the various motion parameters. Table 3.1 shows the main results using the WLS method. It can be seen that the algorithm works quite well even for relatively large motion.

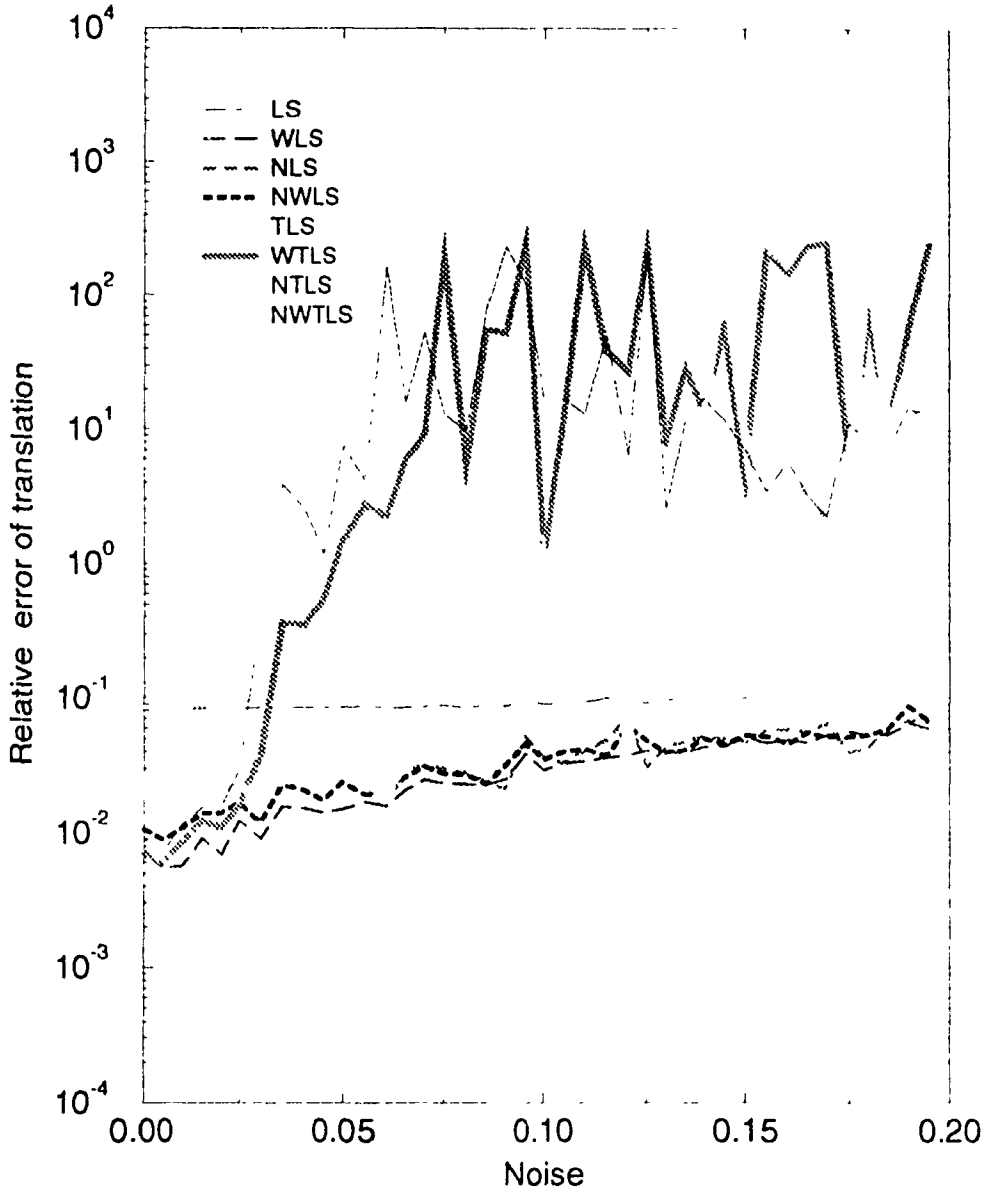


Figure 3.9: The relative error of translation e_t is plotted as a function of noise NSR for LS, WLS, NLS,NWLS, TLS, WTLS, NTLS and NWTLS methods.

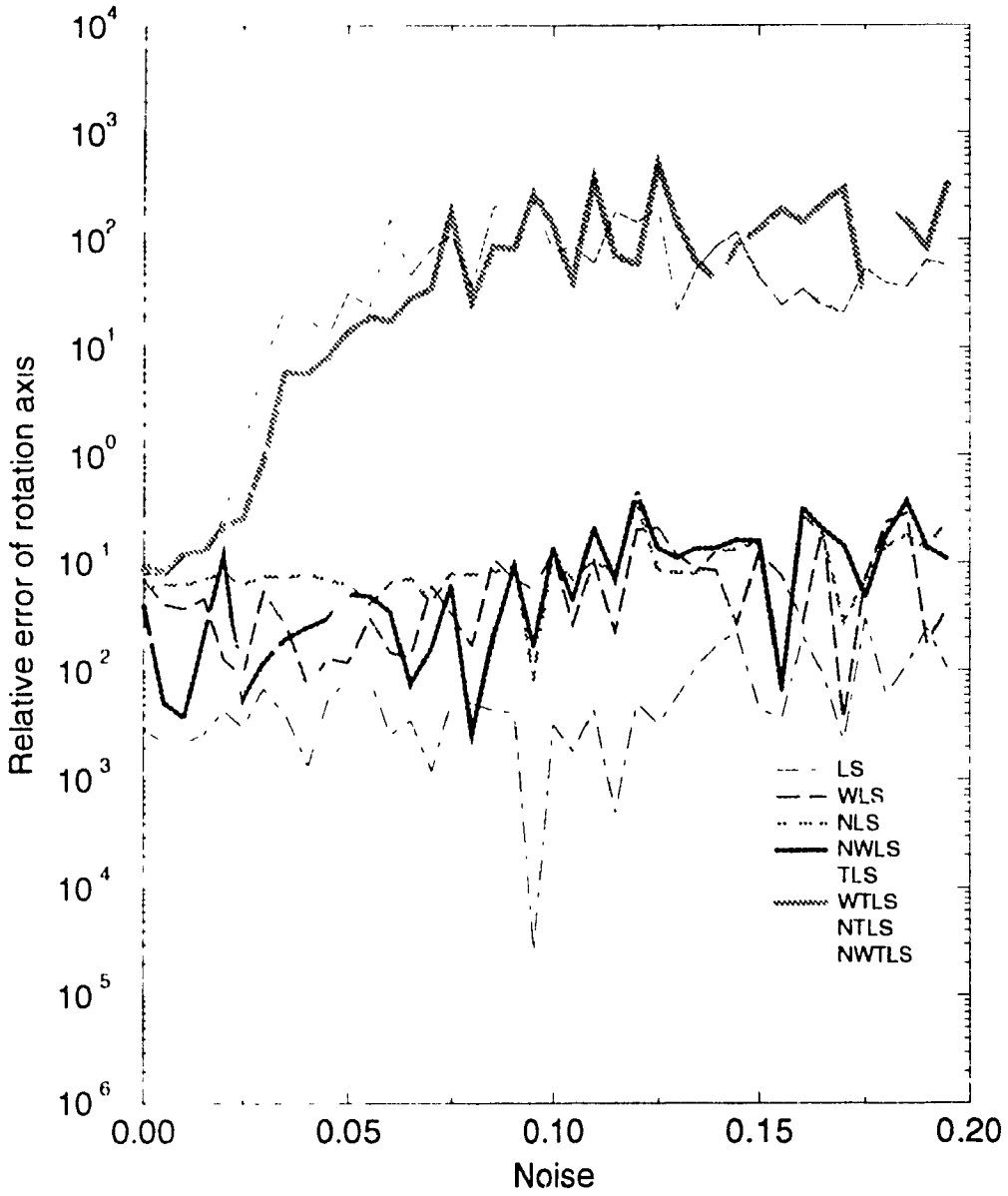


Figure 3.10: The relative error of rotation axis ϵ_n is plotted as a function of noise NSR for LS, WLS, NLS,NWLS, TLS, WTLS, NTLS and NWTLS methods.

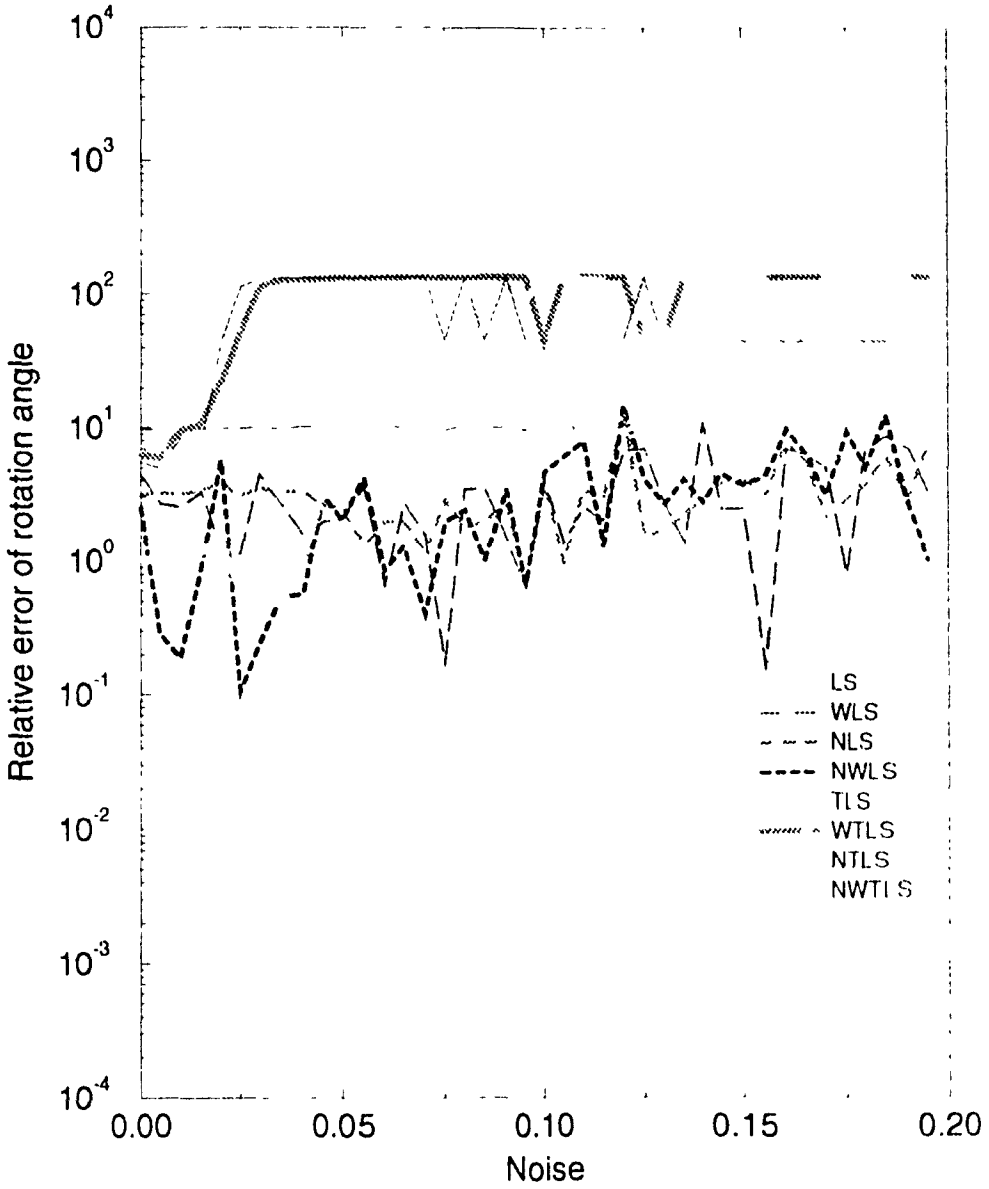


Figure 3.11: The relative error of rotation angle e_θ is plotted as a function of noise NSR for LS, WLS, NLS,NWLS, TLS, WTLS, NTLS and NWTLS methods

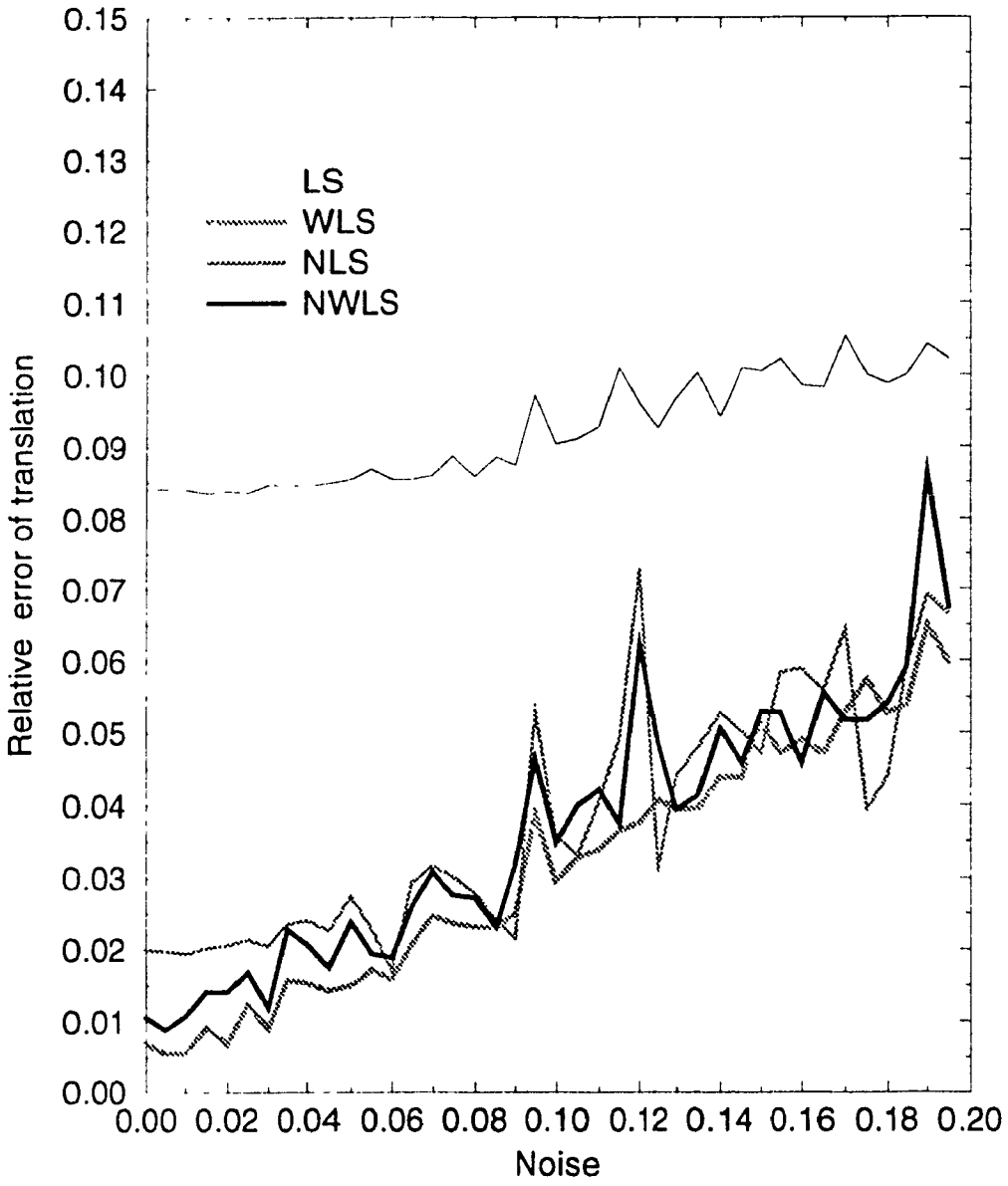


Figure 3.12: The relative error of translation ϵ_t is plotted as a function of noise NSR for LS, WLS, NLS, and NWLS methods.

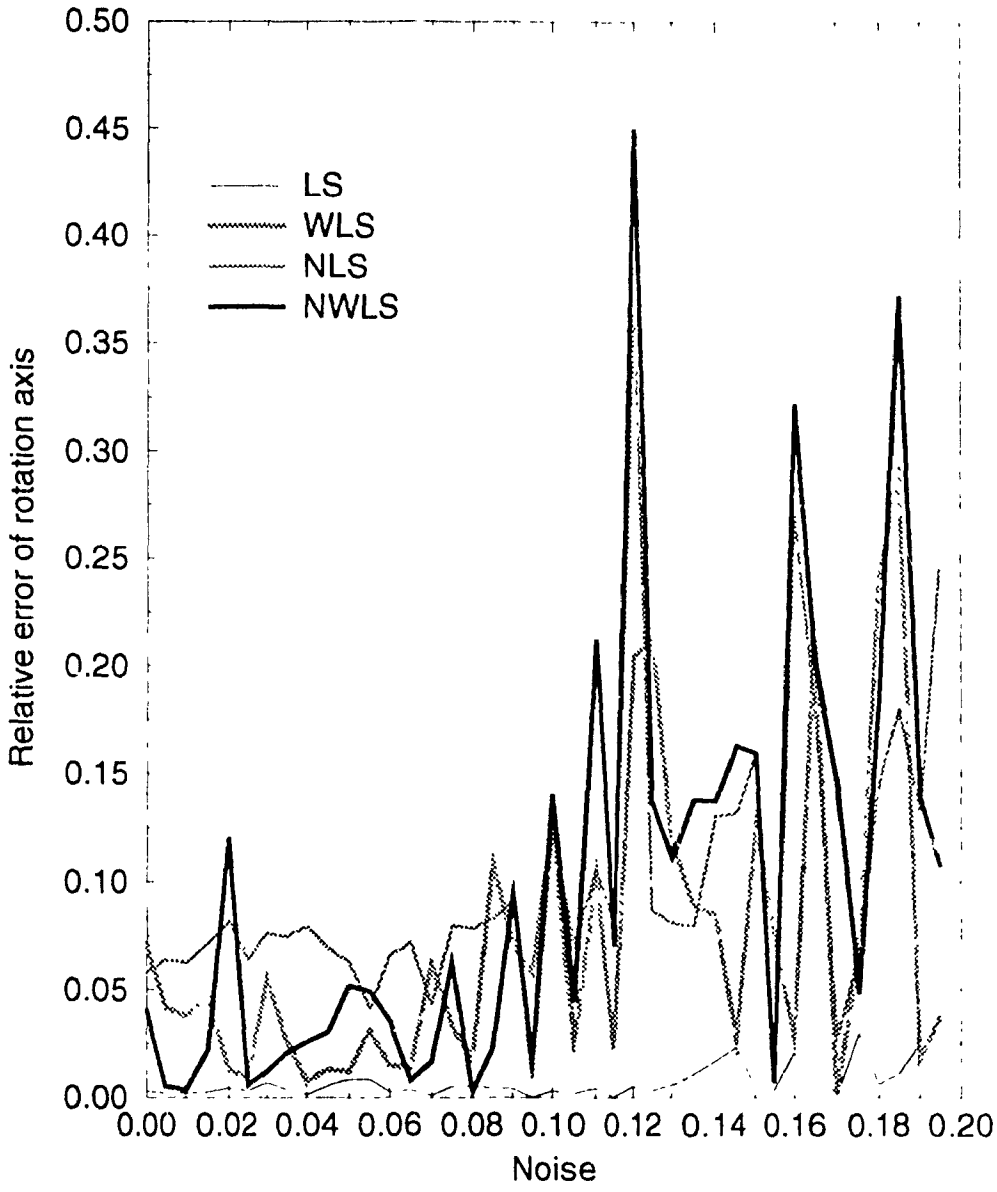


Figure 3.13: The relative error of rotation axis e_a is plotted as a function of noise NSR for LS, WLS, NLS and NWLS methods.

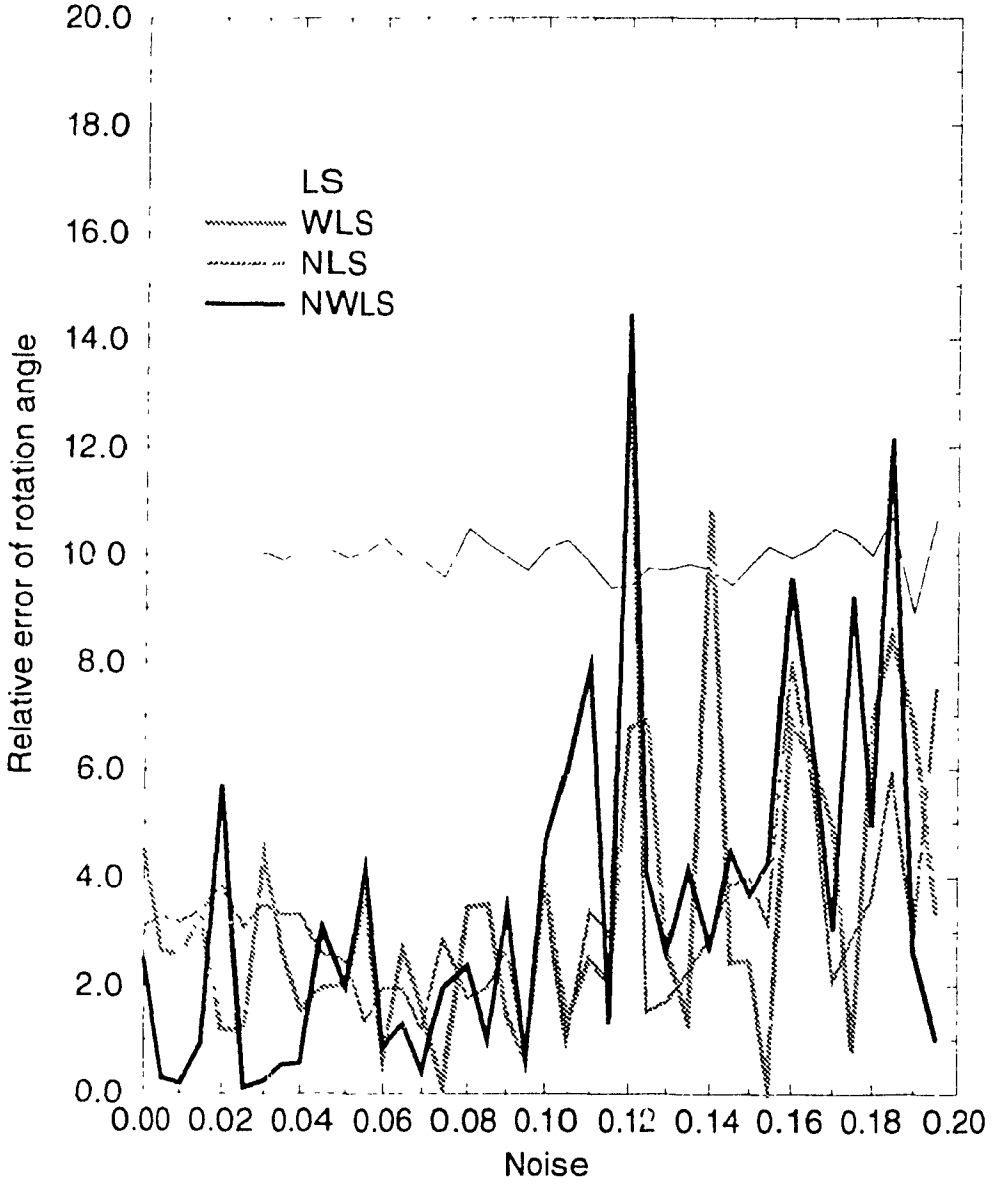


Figure 3.14: The relative error of rotation angle ϵ_θ is plotted as a function of noise NSR for LS, WLS, NLS and NWLS methods

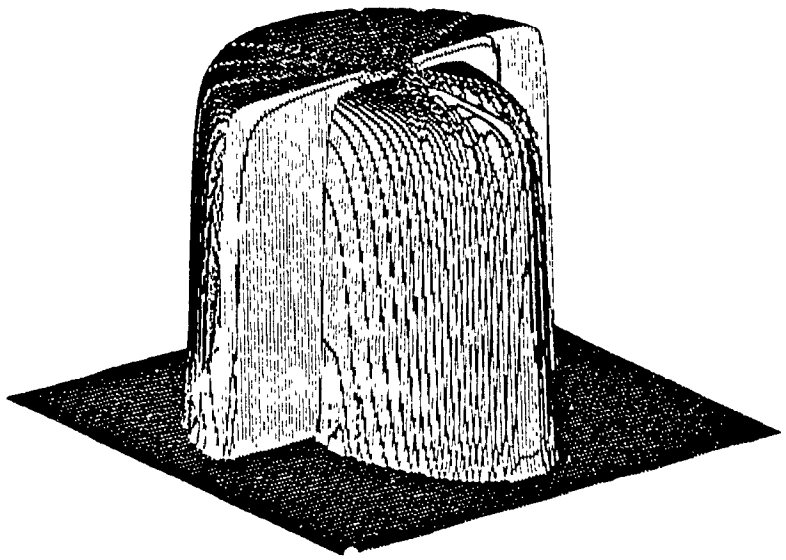


Figure 3.15: The weight image of the synthetic range image

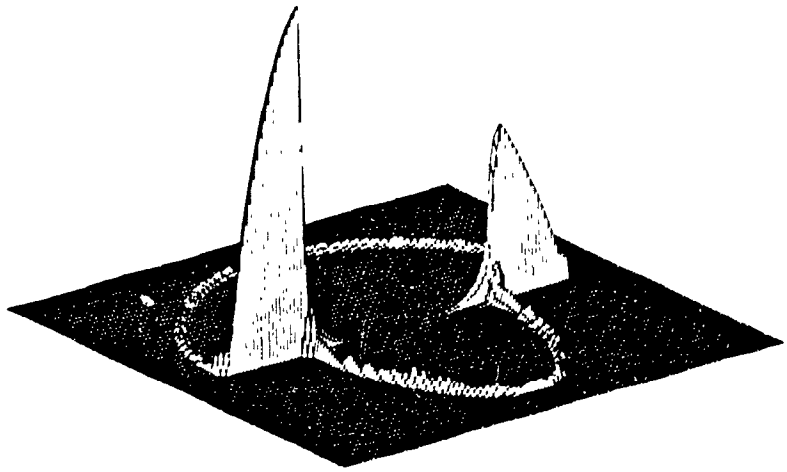


Figure 3.16: The variance image of fitting errors of the synthetic image.

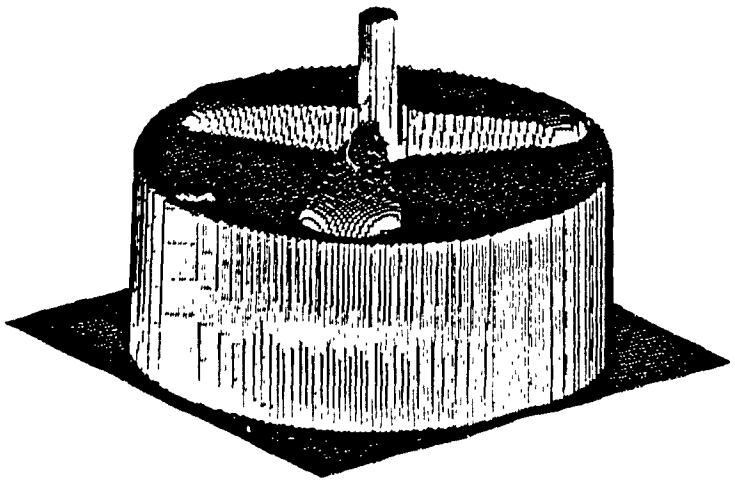


Figure 3.17: Grip5 range image taken by a range finder.

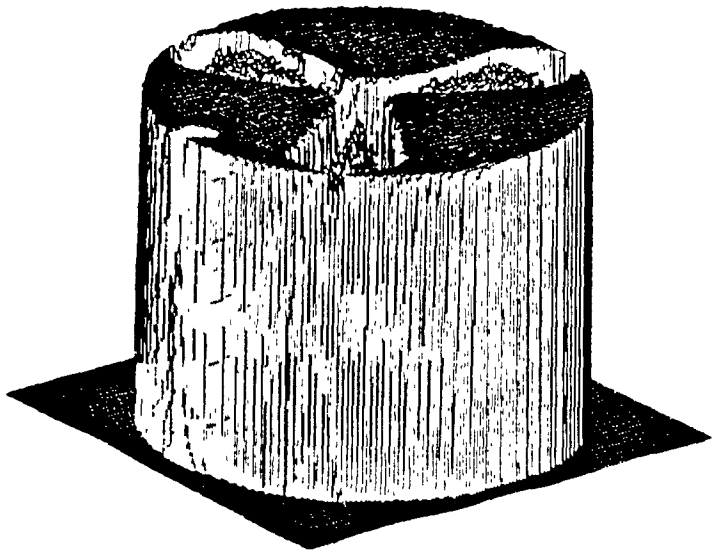


Figure 3.18: The weight image of Grip5.

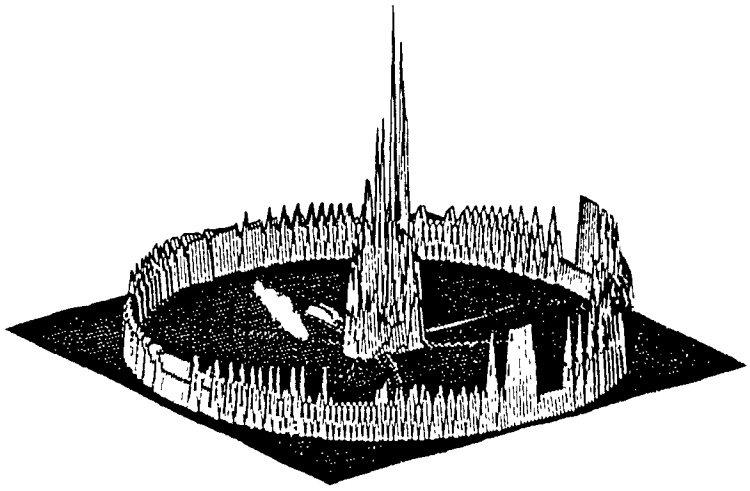


Figure 3.19: The image of variance of fitting errors of Grip5.

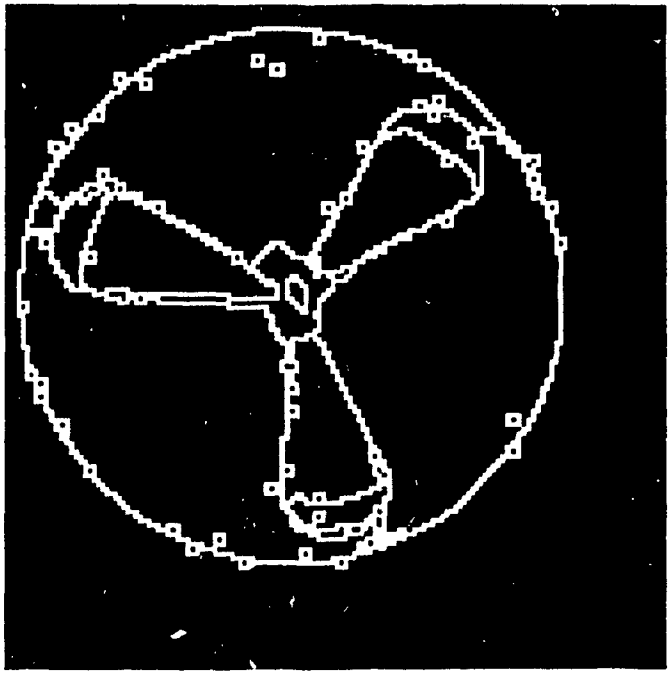


Figure 3.20: The *break points* in Grip5 range image.

Table 3.1: Estimated rigid motion parameters using WLS grip range image.

	t_1	t_2	t_3	n_1	n_2	n_3	θ
true	1.0000	1.0000	1.0000	0.0000	0.0000	1.0000	1.0000
est.	0.9642	0.9793	0.9976	0.0043	-0.0011	1.0000	0.9602
true	2.0000	2.0000	2.0000	0.0000	0.0000	1.0000	2.0000
est.	1.9942	2.1347	1.9512	0.0495	-0.0136	0.9987	2.0655
true	1.0000	1.0000	1.0000	0.2000	0.2000	0.9592	1.0000
est.	0.9611	0.9742	1.0028	0.2095	0.2067	0.9557	0.9594
true	2.0000	2.0000	2.0000	0.2000	0.2000	0.9592	2.0000
est.	2.5200	2.0236	2.0468	0.2676	0.0900	0.9593	1.9073
true	1.0000	1.0000	1.0000	0.5000	0.5000	0.7071	1.0000
est.	0.9328	0.9936	1.0063	0.5200	0.5206	0.6772	0.9593

Table 3.2: Estimated motion parameters by WLS and LS without *break points* for Grip5 range image, thvar = 3.

	v_1	v_2	v_3	ω_1	ω_2	ω_3	$\|\mathbf{v}\|$	θ
Given	1.0000	1.0000	1.0000	0.0000	0.0000	1.0000	0.0174	1.0000
WLS	0.9642	0.9793	0.9976	0.0024	-0.0043	1.0000	0.01627	0.9602
Nonedge	0.9174	0.9852	1.0235	0.0782	-0.0259	0.9966	0.0046	0.2649

Table 3.3: Estimated region motion parameters for shut sequence using WLS.

	t_1	t_2	t_3	n_1	n_2	n_3	θ
shutp810	-0.0514	-2.3296	0.0471	0.9999	0.0037	0.0061	-2.0225
shutp68	-0.0240	-2.2676	0.0312	0.9999	0.0013	0.0111	-1.9793
shutp46	0.0359	-2.2982	0.0266	0.9999	0.0000	0.0081	-1.9934
shutp24	-0.0490	-2.2972	0.0449	0.9999	0.0034	0.0073	-2.0071
shutp02	0.0222	-2.2333	0.0117	0.9999	-0.0109	0.0019	-2.0024
shutm02	0.0139	2.5286	0.0439	0.9999	-0.0071	-0.0090	2.0050
shutm24	-0.0242	2.4214	-0.0097	0.9999	0.0001	-0.0049	2.0145
shutm46	0.0517	2.4564	-0.0259	0.9999	-0.0053	0.0051	2.0145
shutm68	-0.0572	2.4152	0.0008	0.9999	0.0006	0.0055	2.0055
shutm810	-0.0796	2.4437	-0.0016	0.9999	-0.0016	0.0026	1.9959

Another test was done in this experiment as follows. First, the image is segmented using the algorithm given in [62], then the *break points* are located as shown in Figure 3.20, and motion is estimated without these points using the LS method. Similar results were obtained as with the WLS method (see Table 3.2).

3.7.3 Experiment 3: Results on Real Data

The third experiment was carried out on a real sequence of images, which were again taken at NRC. The image sequence contains 11 images: shut0.yin taken in the starting position, shut2p.yin, shut4p.yin, shut6p.yin, shut8p.yin and shut10p.yin taken at positions when the shuttle rolls 2, 4, 6, 8 and 10 degrees in clockwise direction, and shut2m.yin, shut4m.yin shut6m.yin, shut8m.yin and shut10m.yin taken at positions when the shuttle rolls 2, 4, 6, 8 and 10 degrees in anticlockwise direction. Figure 3.21 shows some of these images. In this case, the rotation axis is known in the x axis direction, and inter-frame rotation angle is $\pm$ 2 degrees, but the exact parameters for translation is unknown. The WLS method was applied to this sequence. Table 3.3 gives the estimated motion parameters, it can be seen that the relative errors of the estimated and real rotation angle are less than 2%, which are very promising results. Errors for the rotation axis are even smaller. Figure 3.22 gives the plots of table 3.3.

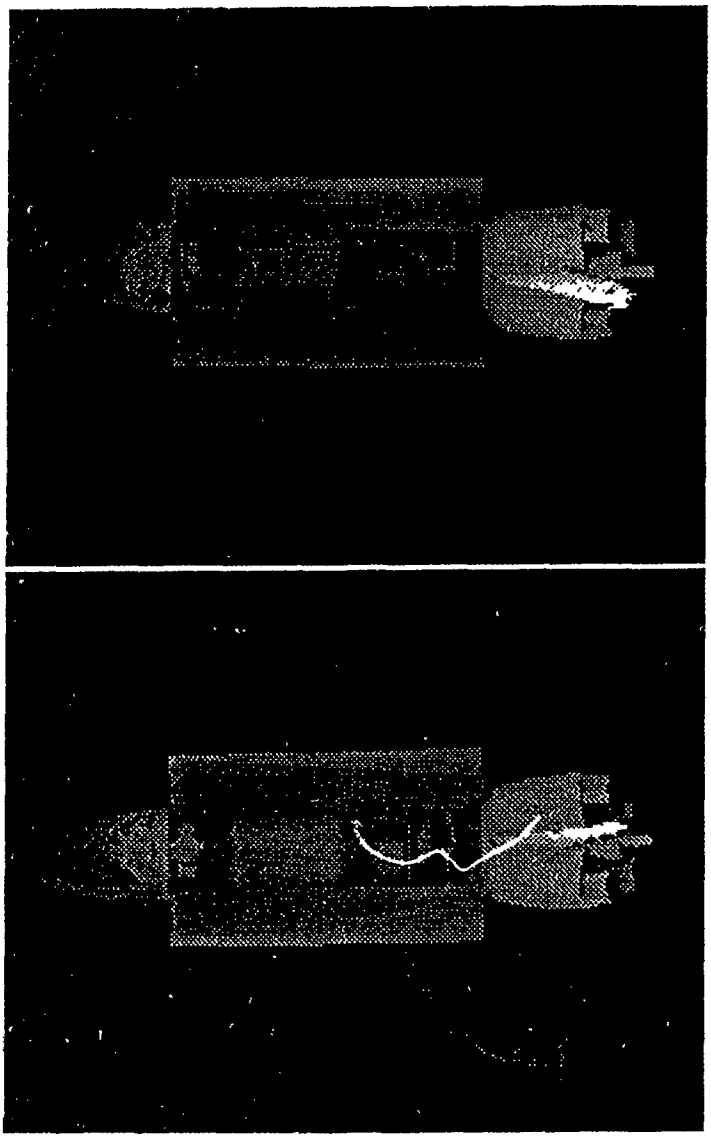


Figure 3.21: "Shut yin" sequence: top: shutp10.yin. bottom: shutm10.yin.

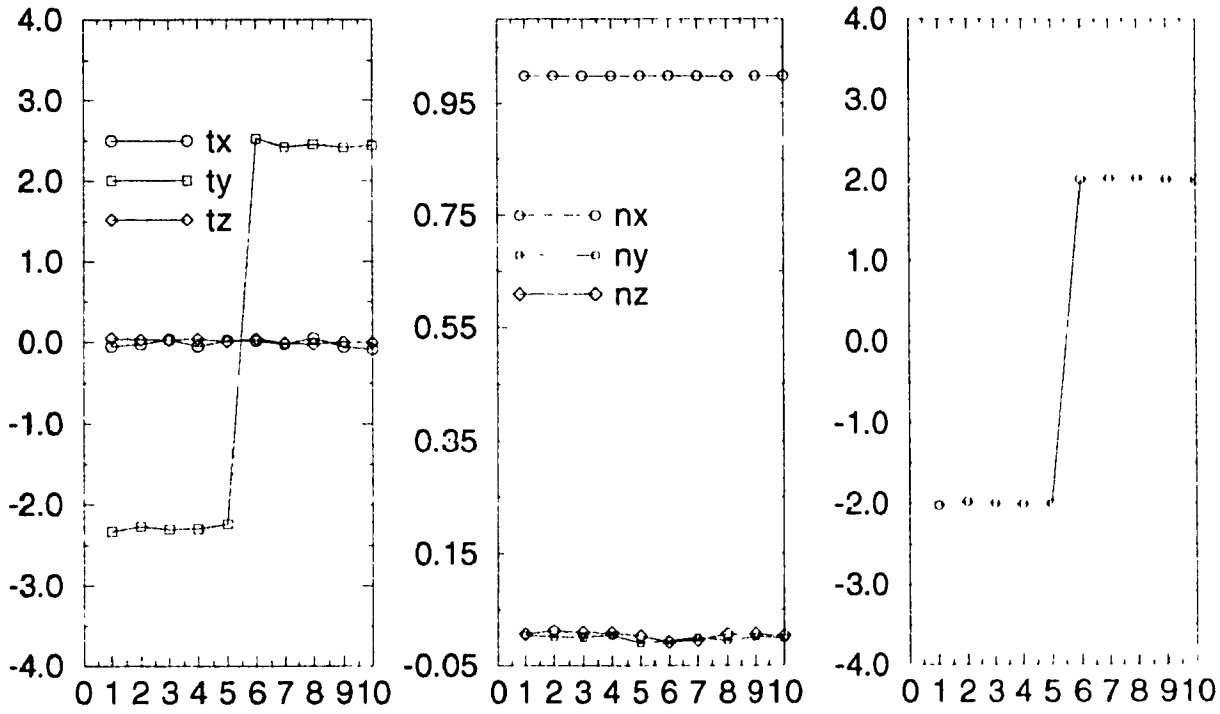


Figure 3.22: Estimated motion parameters for "shut" sequence using WLS.

3.8 Conclusion

In summary, a simple, straightforward, yet very powerful direct method has been presented to estimate 3D motion parameters of a rigid body from range image sequences.

A *local 3D velocity constraint equation* has been derived relating 3D velocity to spatial and temporal derivatives is derived. This equation is an extension of the commonly used *brightness change constraint equation.* Based on this equation, six rigid motion parameters of a single rigid moving object can be directly obtained by solving a set of linear equations using least squares techniques. Several least squares methods have been compared according to their performance against noise and the results show that a weighted least squares method provides the best performance among those different versions of tested least squares methods.

The sufficient and necessary conditions for the unique interpretation of motion using the proposed method have been discussed. The uniqueness of interpretation depends on the structure of the 3D object. It has been found that rigid motion cannot be determined if and only if the surface in a scene is a plane, or a cylindrical surface, or a sphere, or a surface of revolution with its axis of revolution passing through the origin.

The behavior of the algorithm has been analyzed in two steps. First error sources have been discussed, then sensitivity of the algorithm to these errors has been analyzed. It has been demonstrated that the major error, gradient measurement error, consists of two components: random error from random noise in measured depth and systematic error introduced by sampling surfaces discretely in time and space. The random error is determined by the range scanner. The systematic error depends on the spatial resolution of the camera, time interval between two successive frames, the second order partial derivatives of the surface, and motion itself. The error will be reduced if the spatial resolution of the camera and the sampling rate in time are increased. The slower the motion and the surface change in space, the smaller the systematic error.

The sensitivity of the algorithm to noise also depends on the condition number of the linear system. It has also been proven that if normal vectors of a surface tend to be orthogonal to each other, then the system will be less sensitive to noise if the motion is

translation, and for small objects or objects far away from the observer, the estimate of rotation will be very sensitive to noise in data.

Chapter 4

Measuring 3D Velocity Field through Spatial-Temporal Gradient Analysis

The analysis of the single rigid motion problem is only a small branch of motion understanding. There are many applications which require analysis of more complicated motion, such as nonrigid motion, multiple objects in motion, etc. In these cases, the processing can be carried out at two separate levels: low and high. Typically, low level processing locally extracts motion information. In high level processing, the final description of motion is derived.

In this chapter, attention will be paid to low level processing. A 3D velocity field has been chosen to be the output of this level. Several plausible techniques will be reviewed and compared. A new algorithm will be proposed. This algorithm is able to uniquely determine the 3D velocity of each point by using the first and second order spatial and temporal partial derivatives, except at parabolic points where Gaussian curvatures are equal to zero. For each calculated velocity, a measure of the reliability is calculated. For those parabolic points, their velocities are interpolated from the reliable velocities of their neighbors. It will also be shown that the same algorithm can be developed through similarity matching and a gradient-based method, under the assumption that a local surface can be approximated

by a second degree polynomial function.

4.1 Introduction

4.1.1 Low-Level Processing vs. High-Level Processing

It is well known that the motion of a rigid object can be uniquely expressed by a set of six parameters, as discussed in the previous chapter, while how to represent general motion of an unknown object remains an open question.

Looking out the window, a scene, which contains a swinging flag on the top of a building and a flying bird, may be perceived by a human vision system. This vision system is able to provide several levels of descriptions about the scene shown in Figure 4.1. At the lowest level, it detects that something in the scene is moving because the change in the scene is perceived from time to time, furthermore, it may discern that there are two moving objects, more precisely, it may notice that the smaller object is moving from left to right and the bigger object is moving "irregularly". Here "irregularly" means the kind of motion different from rigid motion. Finally, it may explain the scene in the following way: a white bird is flying in the sky from left to right, a red and white flag is swinging with the wind.

In terms of computer vision literature, there are two kinds of processes, i.e., low level process and high-level process, involved in visual systems. The task of the low level process is to measure motion, and that of the high-level is to interpret the measured motion.

As can be seen from the example, the low-level process usually does not require information about the types of motion (rigid or nonrigid, chaotic, etc.), the number of moving objects (single or multiple), overlapping or nonoverlapping objects, etc. Because of the nature of the low-level processes, the output provided by these processes should be pixel-based or local neighborhood-based. One possible choice for the output is the velocities of all pixels.

The high-level process can provide different interpretations of motion depending on the purpose of the vision process (or the applications of the vision systems). Therefore, the representation of motion at this stage is also goal oriented or application dependent

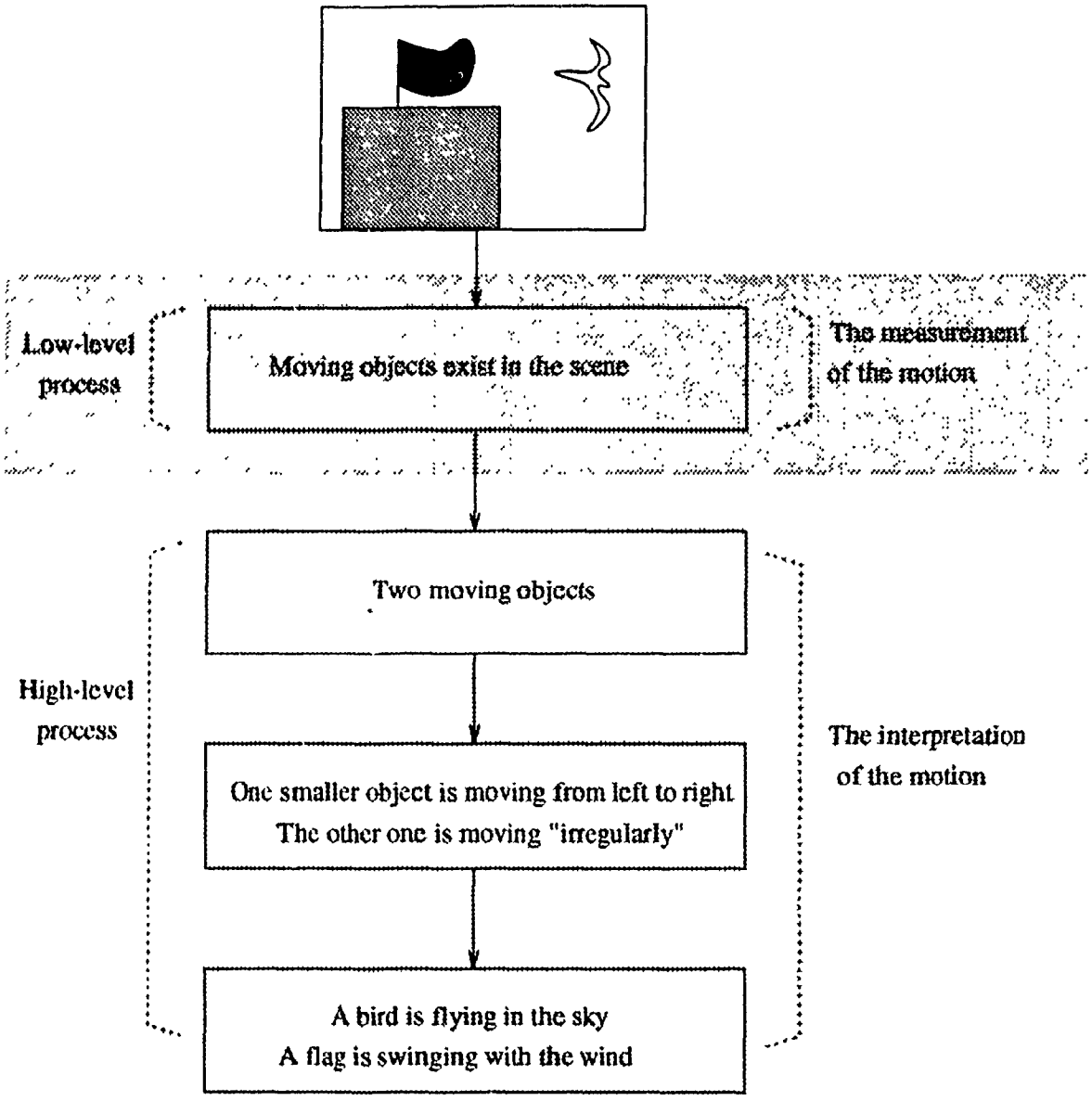


Figure 4.1: An example of perceived motion by our visual system.

At some level, the integration of motion analysis with object recognition is needed.

This chapter will concentrate on the low-level process, specifically, the measurement of a 3D velocity field. A velocity vector field is important. At the basic level, it allows scene segmentation into moving and stationary objects from the perception of motion boundaries. It may extract something worth noticing, which is useful, for example, for moving object tracking, for collision avoidance, etc. Furthermore, it allows scene segmentation into different coherent moving regions as will be discussed later in chapter 6, which may also be helpful in object recognition.

4.1.2 3D Velocity Vector Field vs. 3D Displacement Vector Field

Assuming the same coordinate system as in Figure 3.1, given a point $\mathbf{r} = [\ x \quad y \quad z\ ]^T$ on some surface in a 3D scene, its image is then located at $\mathbf{x} = [\ x \quad y\ ]^T$. Let a point, which is located at time t at position $\mathbf{r}$ on a surface in the 3D scene, move during the interval Δt relative to the camera to a new position $\mathbf{r}' = [\ x' \quad y' \quad z'\ ]^T$, with an image $\mathbf{x}'$. The difference $\mathbf{r} - \mathbf{r}'$ between positions of this 3D point at times t and $t + \Delta t$ is denoted by a 3D displacement vector $\Delta\mathbf{r}$. Dividing the position difference $\Delta\mathbf{r}$ by the time interval Δt, when $\Delta t \to 0$, one obtains the 3D instantaneous velocity vector $\mathbf{u}$.

The range image $z = F(\mathbf{x}, t)$, observable as a function of the image location $\mathbf{x}$ and time t, encodes the local variation of depth value structure, not the image position of identifiable mathematical points in 3D space. It is in general not possible, therefore, to solve the so-called correspondence problem, i.e. for each image location $\mathbf{x}$ at time t, to determine exactly the corresponding image location $\mathbf{x}'$ at time $t + \Delta t$. This happens when a 3D point is occluded or because its new position at time $t + \Delta t$ is not on the sampling grid. However, when both spatial and temporal sampling resolutions are high enough, this problem can be ignored. We will approximate the displacement vector by the shift of image positions and depth values ignoring the difference.

The collection of such displacement vectors or instantaneous velocity vectors for the entire image at all pixel positions constitutes the 3D displacement vector field or the 3D velocity vector field. If the time interval Δt is not too large, then $\Delta\mathbf{r} = \mathbf{u}\Delta t$. Unless the distinction between the displacement vector and velocity vector needs to be made

explicitly, the time interval in question will be set to 1 and the symbol **u** will also be used for the displacement vector. The displacement field will not be distinguished from the velocity field.

4.2 Related Work

A 3D velocity field can be estimated using algorithms which are similar to those used for 2D visual motion (optical flow). The most common methods proposed to compute the optical flow fall into two broad classes:

- gradient-based schemes
- token tracking and matching schemes

Gradient-based schemes rely on the well-known brightness change constraint equation which relates optical velocities to spatial and temporal changes in an image. Token tracking schemes first detect distinctive image features (tokens) and their correspondences are tracked from frame to frame. In this chapter, it will be shown that a 3D velocity field can be estimated from the local velocity constraint equation in a way such that the optical flow is computed based on the brightness change constraint equation. Token tracking schemes are studied in the next chapter.

The well-known brightness change constraint equation is derived from the assumption that image brightness does not change with time, that is,

$$\frac{dE}{dt} = E_x u + E_y v + E_t = 0 \tag{4.1}$$

where $E(x, y, t)$ denotes the image brightness at point (x, y) in the image plane at time t. E_x, E_y and E_t represent the partial derivatives of image brightness with respect to x, y and t, respectively. $u = dx/dt$ and $v = dy/dt$ are the image velocity components in the directions x and y, associated with point (x, y). The collection of such velocity vectors for the entire image constitutes the optical flow for the image.

Equation (4.1) embodies two unknowns u and v, and is not sufficient by itself to specify the optical flow uniquely. This problem is also referred to by some researchers as

aperture problem, that is, the local measurement of movement in the changing images generally provides only one component of local velocity, rather than the full two-dimensional velocity vector. Additional constraints have to be found in order to uniquely compute optical flow together with the brightness change constraint equation. Various additional constraints have been proposed in the literature. Some of the heuristic ones used include:

1. the optical flow is constant over an area of the image [71, 34, 77, 99, 90],
2. the optical flow is smooth, and exhibits the least variation among the set of velocity fields consistent with the changing image [54, 80, 47],
3. and the optical flow is the result of restricted motion, for example, planar rigid motion.

Different techniques have been developed to solve equation (4.1) subject to certain constraints. Among the techniques are least squares methods [99], clustering [99, 93], modified Hough transform [34], maximum likehood estimation [90], which use the assumption of locally constant optical flow. Horn and Schunck [54] and Hildreth [47] utilize a smoothness constraint on (u, v). Waxman et al [111, 112] impose a polynomial model on (u, v) for this purpose. A Fourier domain method for image flow extraction is found in [36, 45]. The following sections will discuss the possibility of applying some of these techniques to compute a 3D velocity field.

4.3 Least Squares Methods

In many situations, 3D velocities are approximately constant in a small but finite neighborhood. Examples include: a moving object that is far away, a nearby object rotating around a point which is far away, an object translating in 3D space, an object undergoing a rigid motion with large translational velocity and small rotational velocity. Let S_n be a small but finite neighborhood of a point (x_0, y_0). If the surface in S_n only undergoes a translation or a translation with a relatively small rotation with respect to the translation, then the former case is called a locally pure translation and the latter a locally dominant

translation. In these two cases, 3D instantaneous velocities are either the same or almost the same in S_n so that they can be treated as equal.

As discussed in the previous chapter, for each point (x_i, y_i) in S_n, its 3D velocity is confined to the velocity constraint plane, which is

$$F_{x_i} u_1 + F_{y_i} u_2 + F_{t_i} = u_3 \tag{4.2}$$

where $F_{x_i}, F_{y_i}, F_{t_i}$ are the first order partial derivatives at point (x_i, y_i), u_1, u_2, u_3 are translational velocities of the surface in S_n. Assuming that there exist n points in S_n, then estimating 3D velocity can be viewed as a problem of fitting a plane to a set of n noisy data $F_{x_i}, F_{y_i}, F_{t_i}, i = 1, \cdots, n$, where u_1, u_2, u_3 are the parameters of the fitted plane.

The simplest way to deal with this problem is the least squares method. There are n linear equations with 3 unknowns (u_1, u_2, u_3). Any standard least squares or weighted least squares or total least squares method can then be used for this problem. The standard least squares solution is

$$\mathbf{u} = \mathbf{A}^{+}\mathbf{b} \tag{4.3}$$

where

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} -F_{x_1} & -F_{y_1} & 1 \\ & \cdots & \\ -F_{x_n} & -F_{y_n} & 1 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} F_{t_1} \\ \cdots \\ F_{t_n} \end{bmatrix}$$

Problems with this approach are obvious. First, the least squares method is appropriate only if a single translational surface is present in S_n. If more than one moving object appears in the neighborhood S_n, the least squares methods approximately provide the average of translational velocities of these objects if the sizes of these objects are comparable (see Figure 4.2). Second, when one or more points in S_n are corrupted by impulse noises or the calculated data $F_{x_i}, F_{y_i}, F_{t_i}$ are unreliable, the least squares methods give unreliable solutions.

The first problem is not very serious if the size of S_n is not very large. Besides, if there is more than one moving object in S_n, then very likely there exist depth discontinuities (object boundaries) in S_n. If the residuals of the least squares solution are large and

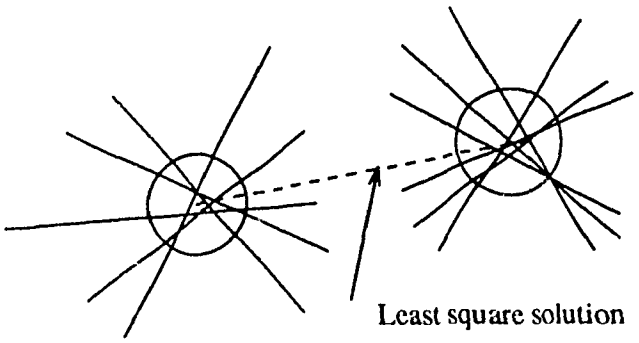


Figure 4.2: If more than one moving object is present in S_n, least squares methods provide the average of the translational velocities of these objects.

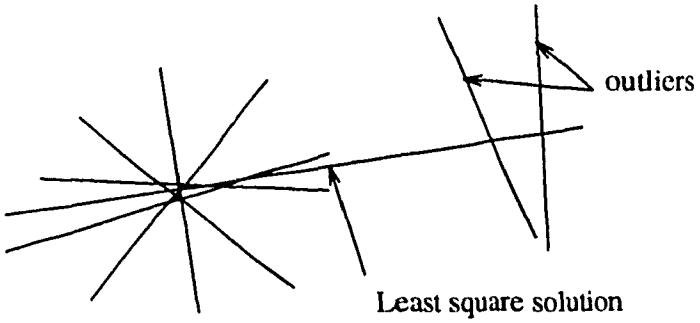


Figure 4.3: When one or more points in S_n are affected by large noise such that their velocity constraint planes are far away from the correct intercepting point, least squares methods give a unreliable solutions.

there are object boundaries, these indicate the presence of multiple moving objects. Low confidence can be assigned to the estimates of velocities at these points or S_n should be changed so that no boundaries are included in S_n.

The second problem is severe considering that the size of S_n is not very large and images are always subject to various kinds of random and systematic noise. Mathematically, those points which create large residuals are called *outliers*. Three kinds of outliers exist in our problem: first, those points subject to large impulse noise during sampling process, second, those points whose first order derivatives can not be reliably estimated, third, the points which are not on the same object to which the considered point belongs if there is more than one moving object in S_n. Usually, the effect of the first kind of outliers on the solution can be reduced by increasing the size of S_n, however, the size of S_n is limited by other factors discussed above. The WLS can reduce the effect of the second kind of outliers on the result by assigning small weights to possible *outliers* as was done for estimating motion of a single rigid object. Unfortunately, WLS does not work very well for the first and third kind of outliers. Robust regression methods such as M-estimators [44], R-estimators, L-estimators [55], random sample consensus (RANSAC) [35, 17], Least Median Squares method (LMS) [88] and Median of the Intercepts (MI) [60] etc., have been developed to reduce the effect of outliers.

4.4 Robust Regression Methods

A review of robust regression methods for computer vision can be found in [79]. The *breakdown point* of a regression method is the smallest amount of outlier contamination that may force the value of the estimate outside an arbitrary range. For example, the standard least squares algorithm has a breakdown point of 0 since one outlier may corrupt the result. Robust estimation methods are designed to increase the breakdown point. *Robust* is in the sense that parameter estimation can tolerate the *outliers*.

The M-estimators minimize the sum of a symmetric, positive definite function $\rho(r_i)$ of the residual r_i, with an unique minimum at $r_i = 0$. A residual is defined as the difference between the data point and the fitted value. In our problem, $r_i = F_{x_i} u_1 + F_{y_i} u_2 + F_{t_i} - u_3$.

Instead of using $\rho(r_i) = r_i^2$ as in the standard least squares method, several ρ functions have been proposed to reduce the influence of large residual values on the estimated fit, for example, the squared error for small residuals and the absolute error for large residuals [55] and a squared sine function for small residuals and a constant for large residuals [8]. The M-estimates of the parameters are then obtained by converting the minimization

$$min \sum_i \rho(r_i) \tag{4.4}$$

into a weighted least squares problem. The weights depend on the assumed ρ function and the data.

The R-estimators are based on ordering the set of residuals. Jaeckel [57] proposed to obtain the parameter estimate by solving the minimization problem

$$min \sum_i a_n(R_i) r_i \tag{4.5}$$

where R_i is the location of the residual in the ordered list, that is, its rank and a_n is a score. Scale invariance (independence from the variance of the noise) is an important advantage of R-estimators over M-estimators.

The L-estimators employ combinations of order statistics and various simulations have shown that L-estimators give less satisfactory results than the other two classes. The M-, R-, and L-estimators have breakdown points that are less than 0.25 for planar surface fitting.

RANSAC, LMS and MI have breakdown close to 0.5. The basic ideas behind those robust regression methods are very similar. Instead of using all points for fitting, they choose some subset of data to estimate parameters. Taking plane fitting as an example, RANSAC works as follows: given a set of n points P ($n > 3$), randomly select a 3-tuple of data points from P, solve a system of 3 linear equations to obtain a fitted plane defined by this 3-tuple, determine subset S of all points in P (consensus set) whose residuals based on the fitted plane are within some error tolerance, if the number of points in S is greater than threshold T, use S to estimate new parameters. If the number of points in S is less than threshold T, randomly select a new 3-tuple, repeat the remaining process. After some predetermined number of trials or exhaustively chosing all combinations of three points, no consensus set with more than T points has been found, either estimate

the parameters using the largest consensus set found, or terminate in failure. In order to use RANSAC, the error tolerance and the consensus set acceptance threshold T must be set a priori. The parameters are hard to predetermine because they depend on the noise levels and the types of fitting models used.

Similar to RANSAC, LMS randomly selects a 3-tuple of data points from P to obtain a fitted plane defined by the 3-tuple, computes the squared residuals of all points in P related to this plane, finds the median of these square residuals (the middle element of the sorted squared residuals), and take this median as the residual for this particular 3-tuple. After repeating the above process K times, the 3-tuple with the least residual is finally used to determine the fitted plane. The points in P can be classified into inliers and outliers relative to this final set. The final fitted plane can be found from those inliers.

MI searches all possible 3-tuples and estimates the parameters of the plane for each 3-tuple, the medians of the estimated parameters are taken as the final estimated values.

All these three methods are based on the assumption that at least one subset of data carry the correct model. When noise corrupts all the data (e.g. Gaussian noise) the quality of the initial model estimates degrades and could lead to incorrect decisions. Besides, they are usually time consuming. For the small number of outliers, L_1 solution (minimization of the sum of the absolute values of the residuals) may be a better choice since the L_1 solution is obtained in one shot [1].

4.5 Clustering Approaches

Instead of treating velocity estimation as regression analysis, we can also look at this problem from another point of view. The motion constraint equation of one point in S_n forms a plane in velocity space. For n points, we have n such constraint planes. Ideally, these planes will intercept at a common point in velocity space as shown in Figure 4.4 if S_n undergoes pure translation, or they intercept in the vicinity of a common point as illustrated in Figure 4.5 if S_n undergoes dominant translation. This common point can be used as the estimate of 3D velocity of point (x_0, y_0).

Three well behaved planes should intercept at one point. If we take any three planes

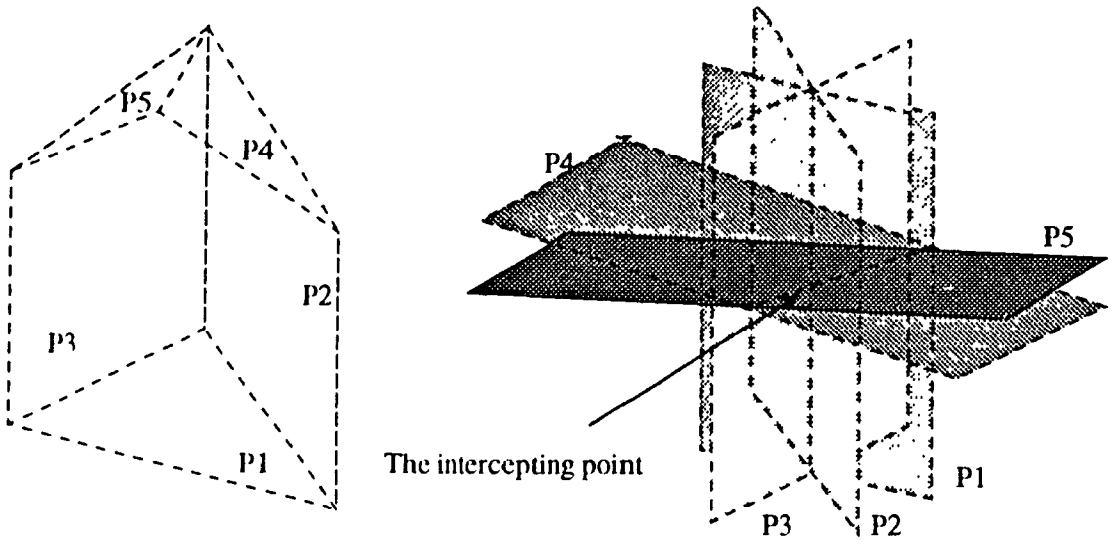


Figure 4.4: Velocity constraint planes intercept at a common point if S_n undergoes pure translation.

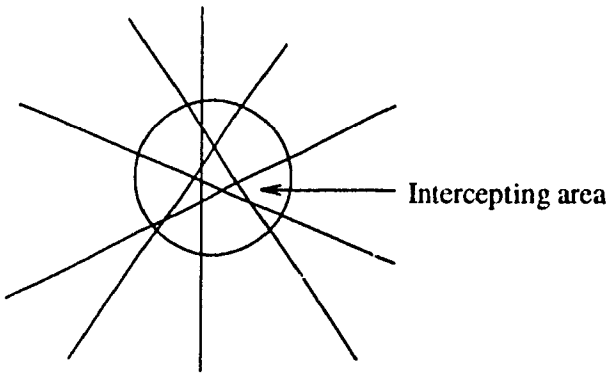


Figure 4.5: Velocity constraint planes intercept in the vicinity of a common point if S_n undergoes dominant translation.

from n motion constraint planes formed by points in S_n, the intercepting point of those three planes forms a point in 3D velocity space. There are $n(n-1)(n-2)/3$ combinations of three constraint planes from n points in S_n, the intercepting points for all combinations may have the following possible distributions in velocity space:

- One blob as shown in Figure 4.6(a) if only a single surface is translating in S_n.
- More than one blob as shown in Figure 4.6(b) if there are more than one moving surface in S_n.
- One blob and some scattered points as shown in Figure 4.6(c) if part of the surface in S_n is subject to noise.
- Scattered points as shown in Figure 4.6(d) if the surface is very noisy.
- Scattered points along one axis as shown in Figure 4.6(e) if the surface in S_n is a cylindrical.

Any clustering algorithms used for pattern classification [28] could be used to locate cluster centers. If more than one cluster center are found, the one closest to the velocity constraint plane of point (x_0, y_0) should be chosen to be the estimate of 3D velocity for point (x_0, y_0). The principal axis can be used as the measure of confidence of the estimated velocities. The advantages of this method are that it is able to deal with multiple moving objects in S_n, and is less sensitive to "outliers". The disadvantages are that much more computations are involved compared with the least squares method and to correctly locate the cluster centers may become a problem.

A related technique is Hough transformation. A three-dimensional histogram of possible velocities is computed as follows. An array of accumulators is created, indexed by discrete values of u_1, u_2 and u_3. Accumulator cells are initially set to zero. If possible u_1, u_2 and u_3 values are on one of the constraint plane. Increment the accumulator cell corresponding to the possible values by one, then the cell with the largest value is detected, and velocities associated with this cell can be used as the estimates. More than one peak in the histogram indicates the possibility of multiple moving objects. The method can also filter out outliers, but reliable location of the peaks may be another problem.

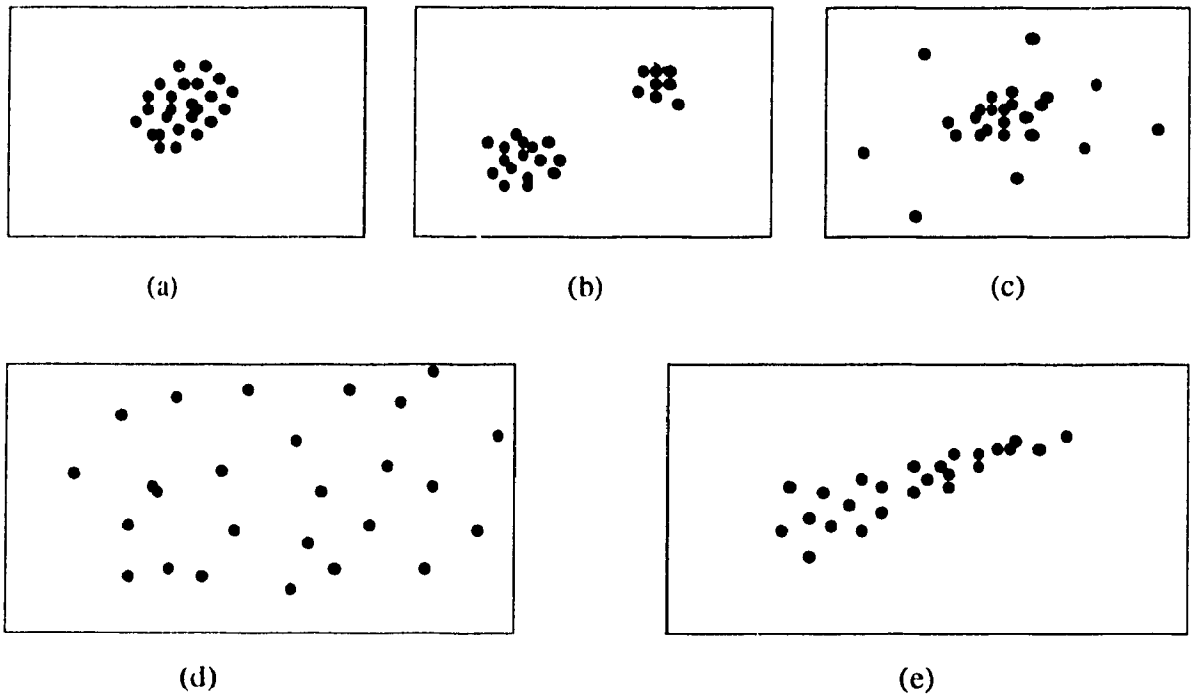


Figure 4.6: Possible distributions of clusters: (a) one blob if only a single surface is translating in S_n, (b) more than one blob if more than one moving surfaces in S_n, (c) one blob and some scattered points if part of the surface in S_n is subject to noise, (d) scattered points if the surface is very noisy, (e) scattered points along one axis if the surface in S_n is cylindrical.

4.6 Least Squares, Robust Regression vs. Clustering Approaches

As discussed before, both robust regression methods and clustering approaches improve the accuracy of the estimated velocities by increasing the cost of computation through extensively searching for the most potential candidates in the parameter space. The advantages of the standard least squares method are its simplicity and ease of computation. However, LS might provide inaccurate results if there are outliers in the data. Therefore, there is a compromise between the accuracy and the time affecting the choice between LS and other methods. In fact, very often, the discontinuities (edges) in range images indicate the possible existence of outliers in the data, hence, LS may be used for smooth surfaces and the other methods for edges.

The basic ideas behind the robust regression methods and clustering approaches are very similar, therefore, their performances are almost equivalent. The choice between them depends on the particular application. As have been already shown, the successes of both methods relies on whether there exists at least one correct subset of data. If images are subject to large random noise, they will fail without an appropriate filter such as a preprocessor to reduce the random noise.

All the methods discussed so far utilize the velocity constraint equation as the model for further processing. This equation is derived under the assumption that a continuous surface in the $xyzt$ space is available. In practice, only a sequence of images is obtained, which are discrete in both space and time. In this case, the velocity constraint equation holds only at the points which are on a plane translating in the 3D space, and is approximately valid for curved surfaces. For edge points it might be far from the truth. The fact that the performance of robust regression and clustering approaches will deteriorate imply the inaccuracy of the model.

4.7 Using the Second Order Derivative Information

The biggest difficulty with the least squares method, regression analysis, and the clustering methods is that they require the image gradients to vary slowly. In this section, an

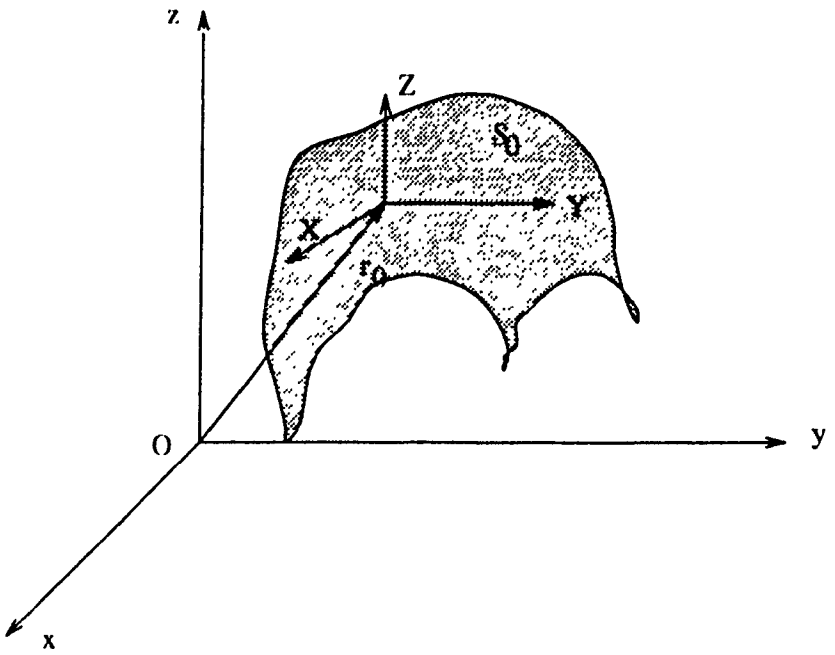


Figure 4.7: Coordinate system.

algorithm will be proposed, which estimates velocities by using the second order derivative information.

4.7.1 Derivation of the Formula

Assume that the first image of a sequence has been taken at time t_0, and without loss of generality, we may always set $t_0 = 0$. Consider the local surface S_0 around the point $\mathbf{r_0} = (x_0, y_0, z_0)^T$ at time t_0. Let XYZ be the local coordinate system with the $\mathbf{r_0}$ as the origin and X, Y and Z axes parallel to x, y and z axes (see Figure 4.7). If S_0 is smooth and small enough, then it can be well approximated by a second degree polynomial function using Taylor's expansion in the local coordinate system. Taking time as an additional dimension to the XYZ space, as the surface S_0 translates in the 3D space, it forms a surface in the $XYZt$ coordinate system. It has been proved in Appendix E that the resultant surface can be expressed by

$$\begin{aligned} Z &= F_{x_0}X + F_{y_0}Y + F_{t_0}t + F_{xy_0}XY + F_{xt_0}Xt + F_{yt_0}Yt \\ &\quad + 1/2F_{xx_0}X^2 + 1/2F_{yy_0}Y^2 + 1/2F_{tt_0}t^2 \end{aligned} \tag{4.6}$$

where $F_{x_0}, F_{y_0}, F_{t_0}, F_{xy_0}, F_{yt_0}, F_{xt_0}, F_{xx_0}, F_{yy_0}$ and F_{tt_0} are the first and second order partial derivatives of S_0 calculated at point (x_0, y_0, z_0, t_0). Taking the total derivative of Z with respect to t, we get

$$\begin{aligned}\frac{dZ}{dt} &= (F_{x_0} + F_{xy_0}Y + F_{xt_0}t + F_{xx_0}X)\frac{dX}{dt} + (F_{y_0} + F_{xy_0}X + F_{yt_0}t \\ &\quad + F_{yy_0}Y)\frac{dY}{dt} + (F_{t_0} + F_{xt_0}X + F_{yt_0}Y + F_{tt_0}t) \qquad (4.7)\end{aligned}$$

Because the $XYZt$ is the translation of the $xyzt$, $\frac{dX}{dt} = \frac{dx}{dt}, \frac{dY}{dt} = \frac{dy}{dt}, \frac{dZ}{dt} = \frac{dz}{dt}$. Let $(u_{1_0}, u_{2_0}, u_{3_0})$ represent the translational velocity of S_0 at time $t = t_0 = 0$, substituting them into equation (4.7) yields

$$\begin{aligned}u_{3_0} &= (F_{x_0} + F_{xy_0}Y + F_{xx_0}X)u_{1_0} + (F_{y_0} + F_{xy_0}X + F_{yy_0}Y)u_{2_0} \\ &\quad + (F_{t_0} + F_{xt_0}X + F_{yt_0}Y) \qquad (4.8)\end{aligned}$$

Rearranging equation 4.8 yields

$$\begin{aligned}&(F_{x_0}u_{1_0} + F_{y_0}u_{2_0} + F_{t_0} - u_{3_0}) + (F_{xx_0}u_{1_0} + F_{xy_0}u_{2_0} + F_{xt_0})X \\ &+ (F_{xy_0}u_{1_0} + F_{yy_0}u_{2_0} + F_{yt_0})Y = 0 \qquad (4.9)\end{aligned}$$

This equation is valid for any point (X, Y) on the surface S_0, thus the coefficients of X and Y must equal zero since $F_{x_0}, F_{y_0}, F_{t_0}, F_{xy_0}, F_{yt_0}, F_{xt_0}, F_{xx_0}, F_{yy_0}, F_{tt_0}$ and $u_{1_0}, u_{2_0}, u_{3_0}$ are constants. From this requirement, a system of linear equations can be formed as follows

$$F_{xx_0}u_{1_0} + F_{xy_0}u_{2_0} + F_{xt_0} = 0 \qquad (4.10a)$$

$$F_{xy_0}u_{1_0} + F_{yy_0}u_{2_0} + F_{yt_0} = 0 \qquad (4.10b)$$

$$F_{x_0}u_{1_0} + F_{y_0}u_{2_0} + F_{t_0} - u_{3_0} = 0 \qquad (4.10c)$$

The solution of this equation set gives us the estimate of the 3D velocity of point (X_0, Y_0) at time $t = t_0$. Similarly, we can find the estimates for any other points. In general, a 3D velocity field (u_1, u_2, u_3) can be uniquely estimated, except at points where $F_{xy}^2 - F_{xx}F_{yy}$ equals zero since

$$u_1 = \frac{F_{xy}F_{yt} - F_{yy}F_{xt}}{F_{xx}F_{yy} - F_{xy}^2} \qquad (4.11a)$$

$$u_2 = \frac{F_{xy}F_{xt} - F_{xx}F_{yt}}{F_{xx}F_{yy} - F_{xy}^2} \qquad (4.11b)$$

$$u_3 = F_x u_1 + F_y u_2 + F_t \qquad (4.11c)$$

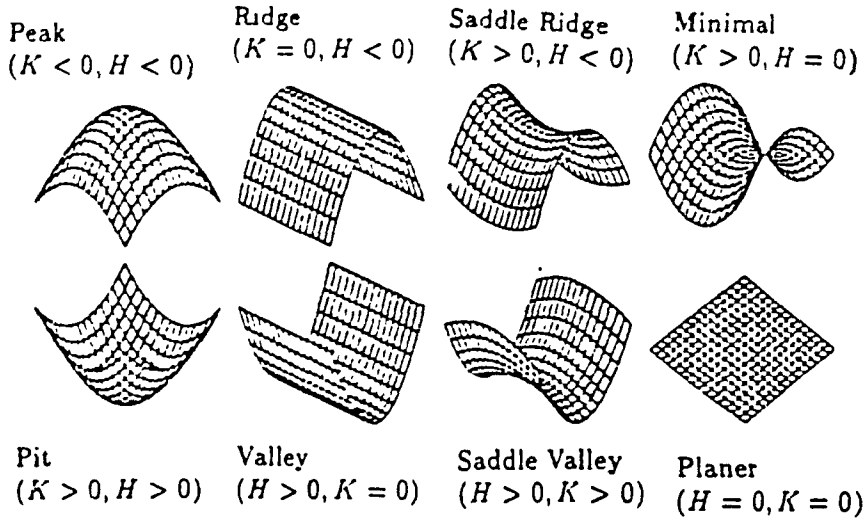


Figure 4.8: Eight types of fundamental surfaces.

where $F_{\cdot}$ and $F_{\cdot\cdot}$ are the first and the second order derivatives of point (x, y) computed at time t_0.

According to differential geometry[1] $F_{xy}^2 - F_{xx}F_{yy}$ is proportional to Gaussian curvature[2]. There exist eight types of fundamental local surfaces [13] as shown in Figure 4.8, among which four types of local surfaces have zero Gaussian curvature[3]. They are: planar, cone-shaped, ridge-shaped, or valley-shaped. Looking at such surfaces locally, certain motions cannot be perceived: a translation in the direction which is orthogonal to the normal vector of a planar surface, a translation along the ridge of a ridge-shaped surface, if no other information is available. Thus, an unique interpretation of motion cannot be locally obtained for such surfaces. For any zero Gaussian curvature points (parabolic points), their motions have to be inferred from their neighboring points.

If the local coordinate system is transformed such that it is aligned with the principal axis, say, X, Y and Z axis are aligned with the maximum curvature, minimum curvature and normal directions (principal coordinates), respectively, as illustrated in Figure 4.9, equations (4.11a,4.11b,4.11c) are simplified to

$$u_1 = -F_{xt}/F_{xx} \tag{4.12a}$$

[1] An introduction to differential geometry is included in the next chapter.

[2] For simplicity, $F_{xy}^2 - F_{xx}F_{yy}$ will be referred as Gaussian curvature as long as it does not cause any confusion.

[3] A point with zero Gaussian curvature is called *a parabolic point*.

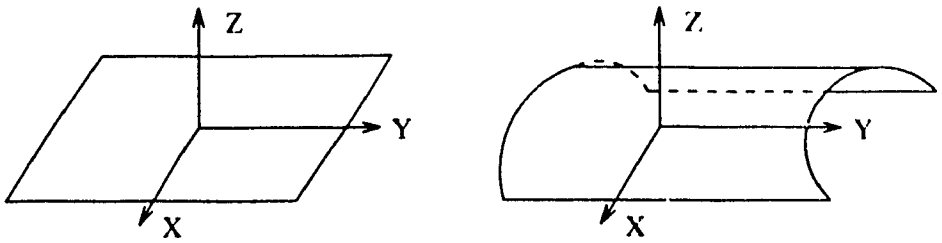


Figure 4.9: Principal coordinate system for planar and cylindrical surfaces.

$$u_2 = -F_{yt}/F_{yy} \tag{4.12b}$$

$$u_3 = -F_x F_{xt}/F_{xx} - F_y F_{yt}/F_{yy} + F_t \tag{4.12c}$$

since F_{xy} equals zero in this new coordinate system.

From the above equation set, some clear explanation of the motion of a plane or a cylinder can be obtained. For a plane in the principal coordinate system, x, y axis are on the plane, z axis is in the normal direction, $F_{xt} = F_{yt} = 0$, $F_{xx} = F_{yy} = 0$, hence any arbitrary values of u_1 and u_2 are valid because they are velocities orthogonal to the normal vector, but $u_3 = F_t$ can be uniquely determined. Similarly, for a cylinder, $F_{yy} = 0$ because the minimum curvature is zero for a cylinder surface, so we can uniquely determine u_1 and u_3, while u_2 cannot be determined uniquely because it represents the velocity component along its symmetry axis.

4.7.2 Geometric Explanation

Equations (4.11a,4.11b,4.11c) can be explained in another way. In terms of the assumption of pure translation or dominant translation in a local area, the direction and magnitude of the spatial gradient of the surface in the local area do not change with time. If F_x is not equal zero, then

$$\frac{d}{dt}\left(\frac{F_y}{F_x}\right) = 0 \tag{4.13a}$$

$$\frac{d}{dt}(F_x^2 + F_y^2) = 0 \tag{4.13b}$$

Expanding them yields

$$\frac{\frac{dF_y}{dt}F_x - \frac{dF_x}{dt}F_y}{F_x^2} = 0 \tag{4.14a}$$

$$2\frac{dF_x}{dt}F_x + 2\frac{dF_y}{dt}F_y = 0 \quad (4.14b)$$

where

$$\frac{dF_x}{dt} = F_{xx}\frac{dx}{dt} + F_{xy}\frac{dy}{dt} + F_{xt} \quad (4.15a)$$

$$\frac{dF_y}{dt} = F_{xy}\frac{dx}{dt} + F_{yy}\frac{dy}{dt} + F_{yt} \quad (4.15b)$$

Since F_x is not zero, equations (4.14a) and (4.14b) are equivalent to

$$\frac{dF_y}{dt}F_x - \frac{dF_x}{dt}F_y = 0 \quad (4.16a)$$

$$\frac{dF_x}{dt}F_x + \frac{dF_y}{dt}F_y = 0 \quad (4.16b)$$

Summing equations (4.16a) and (4.16b) produces

$$\frac{dF_x}{dt}F_y = 0 \quad (4.17)$$

Subtracting equation (4.16a) from equation (4.16b) gives

$$\frac{dF_y}{dt}F_x = 0 \quad (4.18)$$

If both F_x and F_y are not equal to zero, then equations (4.17) and (4.18) can be reduced to

$$\frac{dF_x}{dt} = 0 \quad (4.19a)$$

$$\frac{dF_y}{dt} = 0 \quad (4.19b)$$

that is,

$$F_{xx}u_1 + F_{xy}u_2 + F_{xt} = 0 \quad (4.20a)$$

$$F_{xy}u_1 + F_{yy}u_2 + F_{yt} = 0 \quad (4.20b)$$

These two equations together with the 3D velocity constraint equation give the same solution as equations (4.11a,4.11b,4.11c).

4.7.3 3D Displacement Field: The Discontinuous Case

The above formula are derived under the assumption that a continuous surface in $xyzt$ coordinate system is given. In practice, only a sequence of images are available. They

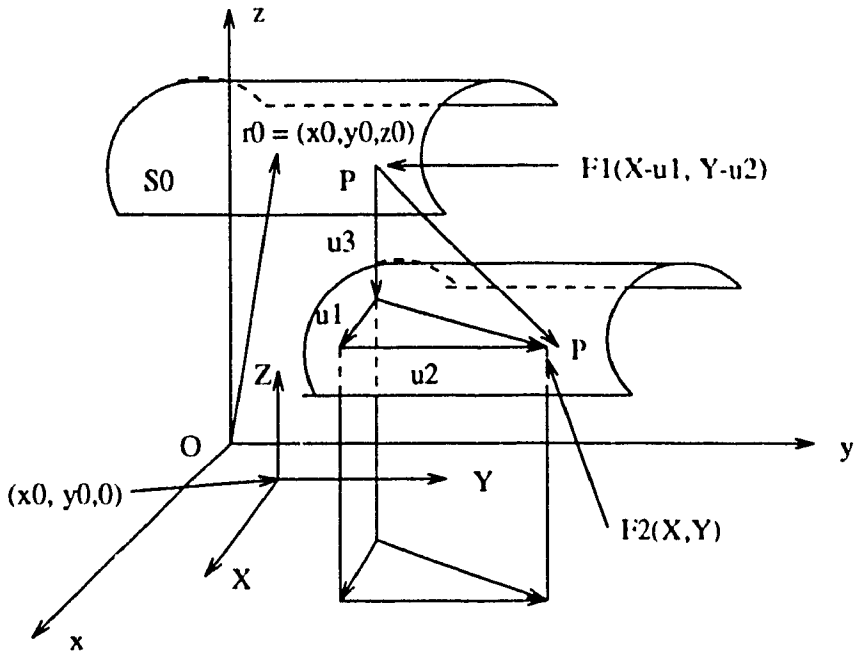


Figure 4.10: Coordinate system for 3D displacement.

are discrete in both space and time. In this case, only the 3D displacement field can be estimated.

Assuming that two range images have been recorded at times t_1 and t_2, taking consideration of a local surface S_0 in the neighborhood of the image position (x_0, y_0), under the assumption of small inter-frame movement, the surface S_0 at time t_1 can be considered to be translated by a 3D displacement vector $\mathbf{d} = (u_1\Delta t, u_2\Delta t, u_3\Delta t)^T$ (see Figure 4.10), where $\mathbf{u} = (u_1, u_2, u_3)^T$ is the 3D instantaneous velocity of point (x_0, y_0, z_0) and $\Delta t = t_2 - t_1$. Without loss of generality, Δt may be set to 1. Let XYZ be the local coordinate system with the point $(X_0, Y_0, 0)$ as the origin, and $F_1(X, Y)$ denote the range value at image position (x, y) in the first image, then this point will be at image position $(x + u_1, y + u_2)$ with range value $F_1(X, Y) + u_3$ in the second image. Let $F_2(X, Y)$ denote the range value at image position (x, y) in the second image, if u_1 and u_2 are small enough such that the depth at image position $(x + u_1, y + u_2)$ remains well approximated by the same function used for image position (x, y) in the second image, then range value at position $(x + u_1, y + u_2)$ in the second image is $F_2(X + u_1, Y + u_2)$. Obviously,

$$F_2(X + u_1, Y + u_2) = F_1(X, Y) + u_3 \tag{4.21}$$

Equivalently, the range value at image position (x, y) is

$$F_2(X, Y) = F_1(X - u_1, Y - u_2) + u_3 \tag{4.22}$$

where it is assumed that u_1 and u_2 are small enough such that the depth at image position $(x - u_1, y - u_2)$ remains well approximated by the same analytic function used for image position (x, y) in the first image. u is determined from the following requirement that

$$E = \sum_{X,Y} (F_2(X, Y) - F_1(X - u_1, Y - u_2) - u_3)^2 \Rightarrow min \tag{4.23}$$

summed over the neighborhood of (x_0, y_0). As discussed before, S_0 can be well approximated by a second degree polynomial function

$$F_1(X, Y) = F_{1_0} + F_{1_x} X + F_{1_y} Y + F_{1_{xy}} XY + 1/2 F_{1_{xx}} X^2 + 1/2 F_{1_{yy}} Y^2 \tag{4.24}$$

Let the neighborhood of (x_0, y_0) be chosen as a square window centered at (x_0, y_0) with length $2k + 1, k = 1, 2, \ldots$. Then E can now be rewritten as

$$\begin{aligned} E \;=\; & \sum_{X,Y=-k}^{k} [F_2(X, Y) - F_{1_0} - F_{1_x}(X - u_1) - F_{1_y}(Y - u_2) - F_{1_{xy}}(X - u_1) \\ & (Y - u_2) - 1/2 F_{1_{xx}}(X - u_1)^2 - 1/2 F_{1_{yy}}(Y - u_2)^2 - u_3]^2 \end{aligned} \tag{4.25}$$

Taking the partial derivative of E with respect to u_1, u_2 and u_3 yields

$$\frac{\partial E}{\partial u_1} = -2 \sum_{X,Y=-k}^{k} [F_{1_x} + F_{1_{xy}}(Y - u_2) + F_{1_{xx}}(X - u_1)] e \tag{4.26a}$$

$$\frac{\partial E}{\partial u_2} = -2 \sum_{X,Y=-k}^{k} [F_{1_y} + F_{1_{xy}}(X - u_1) + F_{1_{yy}}(Y - u_2)] e \tag{4.26b}$$

$$\frac{\partial E}{\partial u_3} = -2 \sum_{X,Y=-k}^{k} e \tag{4.26c}$$

where

$$\begin{aligned} e \;=\; & F_2(X, Y) - F_{1_0} - F_{1_x}(X - u_1) - F_{1_y}(Y - u_2) - F_{1_{xy}}(X - u_1)(Y - u_2) \\ & - 1/2 F_{1_{xx}}(X - u_1)^2 - 1/2 F_{1_{yy}}(Y - u_2)^2 - u_3 \end{aligned}$$

There are many ways to calculate the first and the second partial derivatives from a given image. Here the method described in AppendixD is used. Let

$$S_{kpqr} = \sum_{X,Y=-k}^{k} X^p Y^q F_k(X, Y)^r$$

If $r = 0$, the following notation is used

$$S_{pq0} = \sum_{X,Y=-k}^{k} X^p Y^q$$

Because the selected neighborhood is symmetrical, for any odd p and q

$$S_{pq0} = 0$$

Setting $\frac{\partial E}{\partial u_1}$, $\frac{\partial E}{\partial u_2}$ and $\frac{\partial E}{\partial u_3}$ to zero, and using the notation S_{kpqr} yields

$$\begin{aligned} 0 &= (F_{1_{xx}}^2 S_{200} + F_{1_{xy}}^2 S_{020})u_1 + F_{1_{xy}}(F_{1_{xx}} S_{200} + F_{1_{yy}} S_{020})u_2 \\ &\quad + F_{1_{xx}}(S_{2101} - F_{1_x} S_{200}) + F_{1_{xy}}(S_{2011} - F_{1_y} S_{020}) \end{aligned} \tag{4.27a}$$

$$\begin{aligned} 0 &= F_{1_{xy}}(F_{1_{xx}} S_{200} + F_{1_{yy}} S_{020})u_1 + (F_{1_{yy}}^2 S_{020} + F_{1_{xy}}^2 S_{200})u_2 \\ &\quad + F_{1_{xy}}(S_{2101} - F_{1_x} S_{200}) + F_{1_{yy}}(S_{2011} - F_{1_y} S_{020}) \end{aligned} \tag{4.27b}$$

$$\begin{aligned} 0 &= S_{2001} - S_{000}F_{1_0} - 1/2F_{1_{xx}} S_{200} - 1/2F_{1_{yy}} S_{020} + S_{000}(F_{1_x} \\ &\quad - 1/2F_{1_{xx}} u_1)u_1 + S_{000}(F_{1_y} - 1/2F_{1_{yy}} u_2)u_2 - F_{1_{xy}} S_{000} u_1 u_2 - S_{000} u_3 \end{aligned} \tag{4.27c}$$

Dividing both sides of equations (4.27a,4.27b) by S_{200} and equation (4.27c) by S_{000} yields

$$\begin{aligned} 0 &= (F_{1_{xx}}^2 + F_{1_{xy}}^2)u_1 + F_{1_{xy}}(F_{1_{xx}} + F_{1_{yy}})u_2 \\ &\quad + F_{1_{xx}}(S_{2101}/S_{200} - F_{1_x}) + F_{1_{xy}}(S_{2011}/S_{200} - F_{1_y}) \end{aligned} \tag{4.28a}$$

$$\begin{aligned} 0 &= F_{1_{xy}}(F_{1_{xx}} + F_{1_{yy}})u_1 + (F_{1_{yy}}^2 + F_{1_{xy}}^2)u_2 \\ &\quad + F_{1_{xy}}(S_{2101}/S_{200} - F_{1_x}) + F_{1_{yy}}(S_{2011}/S_{200} - F_{1_y}) \end{aligned} \tag{4.28b}$$

$$\begin{aligned} 0 &= S_{2001}/S_{000} - (S_{000}F_{1_0} + 1/2F_{1_{xx}} S_{020} + 1/2F_{1_{yy}} S_{002})/S_{000} \\ &\quad + (F_{1_x} - 1/2F_{1_{xx}} u_1)u_1 + (F_{1_y} - 1/2F_{1_{yy}} u_2)u_2 \\ &\quad - F_{1_{xy}} u_1 u_2 - u_3 \end{aligned} \tag{4.28c}$$

Summing both sides of equation (4.24) over the neighborhood, it can be easily proved that

$$S_{1001} = S_{000}F_{1_0} + 1/2F_{1_{xx}} S_{020} + 1/2F_{1_{yy}} S_{002}$$

From equations (D.15, D.21, D.22) in Appendix D, we know $S_{020} = S_{200}$ and $F_{2_x} = S_{2101}/S_{200}$, $F_{2_y} = S_{2011}/S_{200}$. Let F_2 denote S_{2001}/S_{000} and F_1 denote S_{1001}/S_{000}, substituting them into equations (4.28a,4.28b, 4.28c) produces

$$\begin{aligned} 0 &= (F_{1_{xx}}^2 + F_{1_{xy}}^2)u_1 + F_{1_{xy}}(F_{1_{xx}} + F_{1_{yy}})u_2 \end{aligned}$$

$$+ F_{1xx}(F_{2x} - F_{1x}) + F_{1xy}(F_{2y} - F_{1y}) \tag{4.29a}$$

$$0 = F_{1xy}(F_{1xx} + F_{1yy})u_1 + (F_{1yy}^2 + F_{1xy}^2)u_2 + F_{1xy}(F_{2x} - F_{1x}) + F_{1yy}(F_{2y} - F_{1y}) \tag{4.29b}$$

$$0 = \bar{F}_2 - \bar{F}_1 + (F_{1x} - 1/2F_{1xx}u_1)u_1 + (F_{1y} - 1/2F_{1yy}u_2)u_2 - F_{1xy}u_1u_2 - u_3 \tag{4.29c}$$

Now multiplying equation (4.29a) by F_{1yy} and equation (4.29b) by F_{1xy}, and subtracting the results, then multiplying equation (4.29a) by F_{1xy} and equation (4.29b) by F_{1xx}, and subtracting the results, we obtain

$$0 = (F_{1xx}F_{1yy} - F_{1xy}^2)(F_{1xx}u_1 + F_{1xy}u_2 + F_{2x} - F_{1x}) \tag{4.30a}$$

$$0 = (F_{1xx}F_{1yy} - F_{1xy}^2)(F_{1xy}u_1 + F_{1yy}u_2 + F_{2y} - F_{1y}) \tag{4.30b}$$

When $(F_{1xx}F_{1yy} - F_{1xy}^2)$ is nonzero, then these two equations are equivalent to

$$0 = F_{1xx}u_1 + F_{1xy}u_2 + F_{2x} - F_{1x} \tag{4.31a}$$

$$0 = F_{1xy}u_1 + F_{1yy}u_2 + F_{2y} - F_{1y} \tag{4.31b}$$

Equation (4.31a) and equation (4.31b) are two linear equations of u_1 and u_2. Solving these equations and substituting u_1, u_2 into equation (4.29c), if the Gaussian curvature $K = F_{1xx}F_{1yy} - F_{1xy}^2$ at point (x, y) in the first image does not equal zero, the displacement vector **u** can be calculated as follows

$$u_1 = \frac{(F_{2y} - F_{1y})F_{1xy} - (F_{2x} - F_{1x})F_{1yy}}{F_{1xx}F_{1yy} - F_{1xy}^2} \tag{4.32a}$$

$$u_2 = \frac{(F_{2x} - F_{1x})F_{1xy} - (F_{2y} - F_{1y})F_{1xx}}{F_{1xx}F_{1yy} - F_{1xy}^2} \tag{4.32b}$$

$$u_3 = \bar{F}_2 - \bar{F}_1 + u_1(F_{1x} - 1/2F_{1xx}u_1) + u_2(F_{1y} - 1/2F_{1yy}u_2) - F_{1xy}u_1u_2 \tag{4.32c}$$

In the above equations, if we approximate $F_{2y} - F_{1y}$ by F_{yt} and $F_{2x} - F_{1x}$ by F_{xt}, and omit the second order term u_1u_2, u_1^2, u_2^2 when $t \to 0$, then equation (4.32a,4.32b,4.32c) and equation (4.11a,4.11b,4.11c) are equivalent, which means that similarity matching gives rise to the same result as the gradient-based method does if the sampling rate is high enough.

4.7.4 Implementation

The implementation of the proposed algorithm is very easy, the most time consuming computations are the calculations of the second order derivatives based on the method described in Appendix D, yet each item can be obtained by a close-form formula. This is the most elegant characteristic of the algorithm.

Given two images, the basic algorithm is as follows

1. Preprocess the images using a noise filter (e.g., a median filter or a Gaussian filter).
2. For each considered pixel (x, y), estimate the first order and second order derivatives for the first image and the first order derivatives for the second image using equations (D.21-D.25).
3. Calculate the Gaussian curvature K at the pixel in the first image, if K is larger than a predefined threshold T, then go to the next step, otherwise, mark this pixel and go to the previous step to process the next pixel.
4. Calculate u_1 and u_2 from equations (4.32a,4.32b).
5. Calculate $\bar{F}_1$ and $\bar{F}_1$, then find u_3 from equation (4.32c), where u_1 and u_2 are substituted by the results calculated from the previous step.
6. If all the pixels have been processed, then terminate, otherwise, go back to the second step to process the next pixel.

Two parameters: the size N of a neighborhood for estimating the derivatives, and the threshold T, have to be predefined in the algorithm. At least two problems exist for this basic algorithm. First, how to deal with areas where the assumption of local quadratic surface is not valid? Second, how to estimate velocities of pixels whose Gaussian Curvatures are close to zero? These details will be discussed in the following sections.

The Neighborhood Size for Surface Fitting

The selection of the neighborhood size should consider the following facts. If the neighborhoods is small, then the surface in the neighborhood will be closer to a quadratic

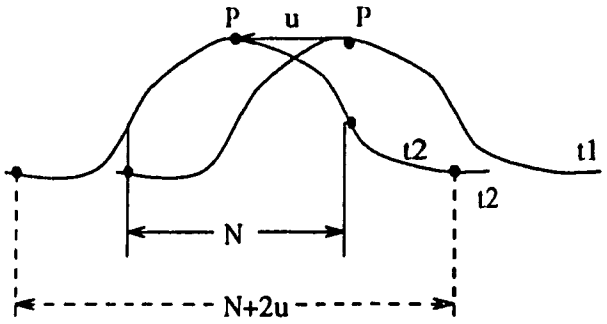


Figure 4.11: The neighborhood size for surface fitting.

surface, which satisfies the assumption for the algorithm. Theoretically, the minimum size of a local surface is six pixels. However, it should be always larger than this minimum size because the bigger the local surface is, the less sensitive to random noise the fitted surface is. On the other hand, the velocities can not be treated as approximately constant if the neighborhood is too large and the motion is not pure translation. Recall that when the algorithm was derived , it has been assumed that the depth at image position $(x - u_1, y - u_2)$ remains well approximated by the same analytic function used for image position (x, y). Therefore, if N is the size for which the assumption of constant motion is valid, then the size of the neighborhood in which the second degree polynomial function is fitted should be $(N + 2u_1) \times (N + 2u_2)$. This idea is illustrated in Figure 4.11. Let u_1max and u_2max be the maximum velocities for u_1 and u_2, then the size for surface fitting should be $(N + 2u_1max) \times (N + 2u_2max)$.

Threshold for Gaussian Curvature

One intuitive argument for the selection of the threshold T is that it should be related to the noise level in the estimated Gaussian Curvature K, which in fact is dependent on the noise level in the images. The threshold T should be at least larger than the standard deviation of the estimated Gaussian Curvature K. Let

$$K = F_{xx}F_{yy} - F_{xy}^2$$

represent the estimated Gaussian Curvature. The formula derived by Nagel in [80] is used for the estimate of the variance of K. According to error propagation theory, the variance

of K may be approximated by

$$\begin{aligned}(\Delta K)^2 &= \left(\frac{\partial K}{\partial F_{xx}}\right)^2(\Delta F_{xx})^2 + \left(\frac{\partial K}{\partial F_{yy}}\right)^2(\Delta F_{yy})^2 + \left(\frac{\partial K}{\partial F_{xy}}\right)^2(\Delta F_{xy})^2 \\ &= (F_{yy})^2(\Delta F_{xx})^2 + (F_{xx})^2(\Delta F_{yy})^2 + (2F_{xy})^2(\Delta F_{xy})^2 \end{aligned} \tag{4.33}$$

Since we are only interested in a threshold value, the squared parameters $(F_{xx})^2, (F_{yy})^2$ and $(F_{xy})^2$ can be replaced by $(\Delta F_{xx})^2, (\Delta F_{yy})^2$ and $(\Delta F_{xy})^2$, which represents a reasonable lower limit for them. Now, assuming that the measurement errors for image pixels are independent of each other, and distributed according to a zero-mean normal distribution $N(0, \sigma)$, then from [80],

$$(\Delta F_{xx})^2 = (\Delta F_{yy})^2 = 4\sigma^2/(S_{400} - S_{220}) \tag{4.34a}$$

$$(\Delta F_{xy})^2 = \sigma^2/S_{220} \tag{4.34b}$$

Substituting these formulas into equation (4.33) yields

$$(\Delta K)^2 = 2(4\sigma^2/(S_{400} - S_{200}))^2 + 4(\sigma^2/S_{220})^2 \tag{4.35}$$

For example, if 3×3 neighborhood is chosen, then $S_{400} = 6$ and $S_{220} = 4$,

$$(\Delta K)^2 = 2.5\sigma^4$$

The threshold T should therefore be

$$T > \Delta K$$

Confidence Measure

As mentioned before, there exist some areas where the estimates of velocities from the basic algorithm are not reliable. The accuracy of estimates of velocities is affected by:

- How well a local surface fits a second degree polynomial function
- The condition number of the system of equations (4.29a, 4.29b)
- How well the motion of a considered local surface can be approximated by a single translational motion

The first factor can be measured by a fitting error defined by

$$e_{fit} = \sum_{X,Y} (F(X,Y) - \tilde{F}(X,Y))^2 \tag{4.36}$$

where $F(X,Y)$ is the measured surface by a sensor and $\tilde{F}(X,Y)$ is the fitted surface.

The second factor characterizes the error propagation, details of which can be found in the last chapter. The condition number of the system of equations (4.31a, 4.31b) can be easily calculated, if Frobenius norm is used, by

$$cond = \frac{(F_{1_{xx}}^2 + 2F_{1_{xy}}^2 + F_{1_{yy}}^2)}{|F_{1_{xx}} F_{1_{yy}} - F_{1_{xy}}^2|} \tag{4.37}$$

The combination of these two measures can be used as a measure of confidence for estimated velocities from the basic algorithm. If the condition number or the fitting error for one pixel is larger than a certain value, then the estimated velocity for the point is not reliable, and the estimate should be discarded from further consideration.

Coping with Parabolic Points

For parabolic points, it is impossible to locally determine their velocities. These velocities have to be interpolated from their neighboring points. A similar problem appears in the computation of optical flow if the intensity of an image is uniform. To overcome this problem, additional information must be available. The most common assumption in optical flow is that the optical flow is smooth and the resultant velocity field should vary least [54]. The problem is that the smoothness assumption is always violated in real images and it is very hard to locate the areas where the assumption is not valid [78]. However, for range images, the discontinuities of a velocity field can be located in most cases. It is safe to assume that the discontinuities of a velocity field only occur where depth is discontinuous. Therefore, the velocity field can be smoothed within areas where there are no discontinuities. Based on this idea, edges are first detected using a Sobel operator. Then the smoothing problem is formulated as a minimization problem in a similar way as in [93].

Let u_1, u_2 and u_3 denote the x, y and z components of the estimated velocity field obtained after reliability checking. Let $\bar{u_1}, \bar{u_2}$ and $\bar{u_3}$ denote the smoothed velocity field. A

smoothed velocity field is found such that a cost for the lack of smoothness in the velocity field between motion boundaries and a cost on the deviation of the smoothed velocity field from the estimated velocity field will be minimized. The cost assigned to the deviation of the smoothed velocity field from the original velocity field estimate is

$$(u_1 - \bar{u}_1)^2 + (u_2 - \bar{u}_2)^2 + (u_3 - \bar{u}_3)^2$$

The cost assigned to the lack of smoothness is the sum of the squares of the magnitudes of the gradients of the x and y velocity field components which is

$$\bar{u}_{1x}^2 + \bar{u}_{1y}^2 + \bar{u}_{2x}^2 + \bar{u}_{2y}^2 + \bar{u}_{3x}^2 + \bar{u}_{3y}^2$$

The combined optimization criterion is

$$\int\int [k^2(\bar{u}_{1x}^2+\bar{u}_{1y}^2+\bar{u}_{2x}^2+u_{2y}^2+\bar{u}_{3x}^2+\bar{u}_{3y}^2)+(u_1-\bar{u}_1)^2+(u_2-\bar{u}_2)^2+(u_3-u_3)^2]dxdy \quad (4.38)$$

where k^2 weighs the degree of smoothing versus the degree of deviation from the original estimated velocity field. The optimization problem is easily solved by the calculus of variation. The result is a set of linear partial differential equations:

$$\begin{aligned} k^2\Delta^2\bar{u}_1 - u_1 &= 0 \\ k^2\Delta^2\bar{u}_2 - u_2 &= 0 \\ k^2\Delta^2\bar{u}_3 - u_3 &= 0 \end{aligned}$$

The equation can be solved iteratively by approximating the Laplacian with a nine point mask

$$\Delta^2 \simeq \begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix}$$

The iterative solution is

$$\bar{u}_1^{n+1} = \frac{1}{1+20k^2}[\bar{u}_1^n + k^2 S(\bar{u}_1^n)] \quad (4.39a)$$

$$\bar{u}_2^{n+1} = \frac{1}{1+20k^2}[\bar{u}_2^n + k^2 S(\bar{u}_2^n)] \quad (4.39b)$$

$$\bar{u}_3^{n+1} = \frac{1}{1+20k^2}[\bar{u}_3^n + k^2 S(\bar{u}_3^n)] \quad (4.39c)$$

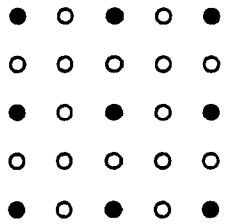


Figure 4.12: An example of subsampling scheme for the LMS method, where the neighborhood is 5×5, it is subsampled at black dots such that the effective window is 3×3.

where the function $S(\cdot)$ represents the computation at the eight surrounding points for calculating the approximation of the Laplacian. If any of the eight points are on the motion boundary, then the velocity field value at the center of the Laplacian replaces the velocity field value at any point that is a boundary point.

Algorithm

After all these considerations, the final algorithm is as follows:

1. The basic algorithm to create the initial estimate of a velocity field (u_1, u_2, u_3).
2. Reliability check to obtain the reliable velocity field (ru_1, ru_2, ru_3).
3. Iteratively smooth velocity field (ru_1, ru_2, ru_3) to get the smoothed velocity field (su_1, su_2, su_3).

4.8 Experimental Results

In this section, the experimental results carried out on both real and synthetic images will be presented. The algorithms discussed in the previous section and a robust regression algorithm (LMS) were tested on the same data in order to compare their performances. One thing that should be mentioned, is that, the solution space was exhaustively searched for the LMS method. For example, for 3×3 neighborhood, all 24 possible 3-tuples were used to obtain the possible estimates of velocities. However, if the size of the neighborhood is larger, the complete search is very slow. The neighborhood was subsampled such that the effective size is 3×3. Figure 4.12 shows an example of subsampling.

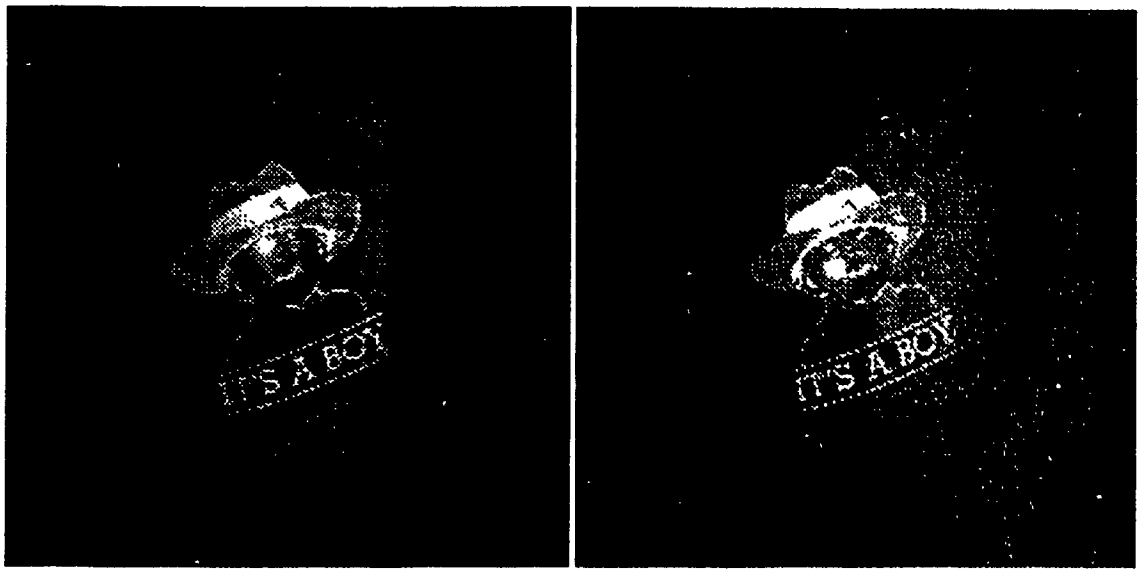


Figure 4.13: Left: the first balloon image, right: the second balloon image

Six image sequences were used. Only two successive images were used in each experiment. Among the images, two sequences are real images, two are synthetic images and the remaining two are quasi-real images. By the term quasi real image sequences, it is meant that the second image in a sequence is obtained by moving the first image using a 3D motion simulation program, while the first image was taken by a range sensor.

Rubber Balloon

The balloon images were taken by a video rate range finder at NRC. The details of the video rate range finder can be found in [12]. The image size was 200×200. The camera also provided an intensity image which is in registration with the range image. Figure 4.13 shows two successive intensity images. It is clear that the balloon is shrinking toward a middle point at bottom of the balloon.

The estimated 3D velocity field is displayed by three projected 2D velocity fields on the three planes. u_1u_2-, u_1u_3- and u_2u_3-planes. The velocity field is reduced to 50×50 in order to get a better visualization. Figures 4.14, 4.15 and 4.16 show the estimated velocity field by the basic algorithm, where a velocity is set to 20 if it is larger than 20. It is obvious that the estimated velocity field is correct except near the boundaries of

the balloon and the boy's image on the balloon, where the surface is very flat and the measured depths are noisy because of the reflection of the shining parts of the boy.

The reliable velocity field is shown in Figures 4.17, 4.18 and 4.19, where "thgau, thfit, thcond, thvel" denote the thresholds for the Gaussian curvature, fitting error, condition number and the maximum possible velocity. Most of the unreliable estimates are removed from the original estimates. Figure 4.20 shows four binary images, where white pixels locate unreliable estimates of velocities based on corresponding confidence measures. The upper left image is obtained by checking the Gaussian curvature, where a white pixel means that the Gaussian curvature at the point is smaller than "thgau". The upper right image is obtained by checking the condition number, where a white pixel means that the condition number at the point is larger than "thcond". The bottom left image is created by checking the sum of the fitting errors in the first image and the second image, where a white pixel means that the sum of the fitting errors at the point is larger than "thfit". The bottom right image is created by checking the maximum allowable velocity for each component, where a white pixel means that one of velocity components at the point is larger than "thvel".

From these images, it can be seen that the measure of the fitting error is very important. Notice that there is a black area between two white areas on the top of the binary image obtained from the fitting error. This area also has unreliable estimates of velocities but it is not possible to discern by checking the fitting error alone. The reason for this is that the pixels in that black area belong to the smooth surface of the balloon in the first image, and they belong to the background in the second image. The fitting errors in the black area of both images are small, but the estimated velocities are unreliable because the algorithm is asked to relate two totally different surfaces to each other. This is an extreme example. The movement around the black area is too great for the small motion assumption. In normal case the measure of the fitting error alone can locate most of the unreliable estimated velocities.

The thresholds for reliability checking can be chosen from the histograms of the calculated values. Figure 4.21 shows the histograms of the Gaussian curvature, condition number, the fitting errors of the first image and the second image from upper left, upper

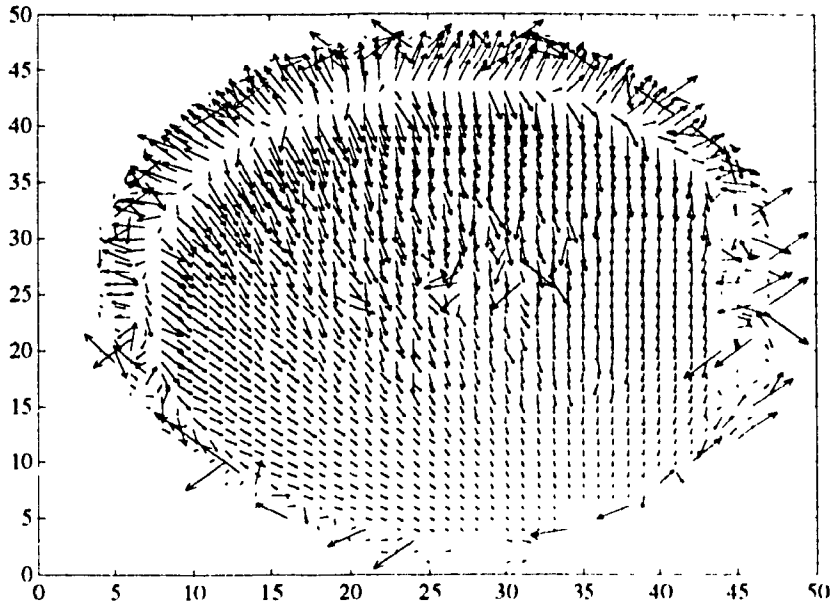


Figure 4.14: The projected 2D velocity field on u_1u_2-plane of the estimated velocity from the basic algorithm, where the neighborhood is 9×9.

right, bottom left and bottom right, respectively. Histograms were created by using 64 equally spaced accumulators. If a value is less than the lowest accumulator, then it is considered to belong to the lowest accumulator. In a similar manner, a value which is larger than the highest accumulator is considered to belong to the highest accumulator. For the histogram of the Gaussian curvature, the accumulators are in the range $[-0.1, 0.1]$. For the condition number, the accumulators are in the range $[0, 50]$, where zero condition number is assigned to a point with zero Gaussian curvature. For the fitting error, the accumulators are in the range [0, 50]. For each histogram, the number of pixels is set to 1000 if exceeded.

The smoothed velocity field after 5 iterations is shown in Figures 4.22, 4.23 and 4.24. It does provide a more uniform velocity field.

Figure 4.25 shows the projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method. 7×7 Gaussian smoothing was first applied to both images since the LMS method is sensitive to random noise. Compared with Figure

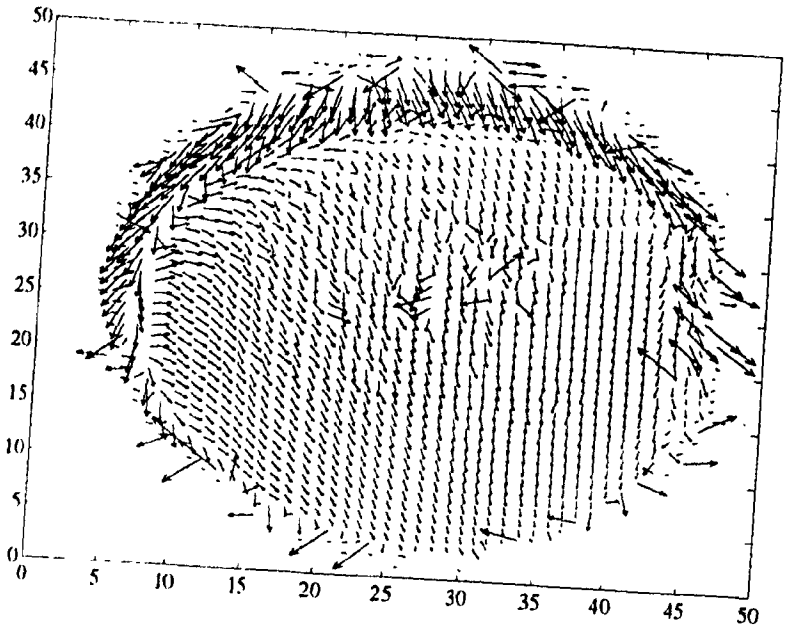


Figure 4.15: The projected 2D velocity field on u_1u_3-plane of the estimated velocity from the basic algorithm, where the neighborhood is 9×9.

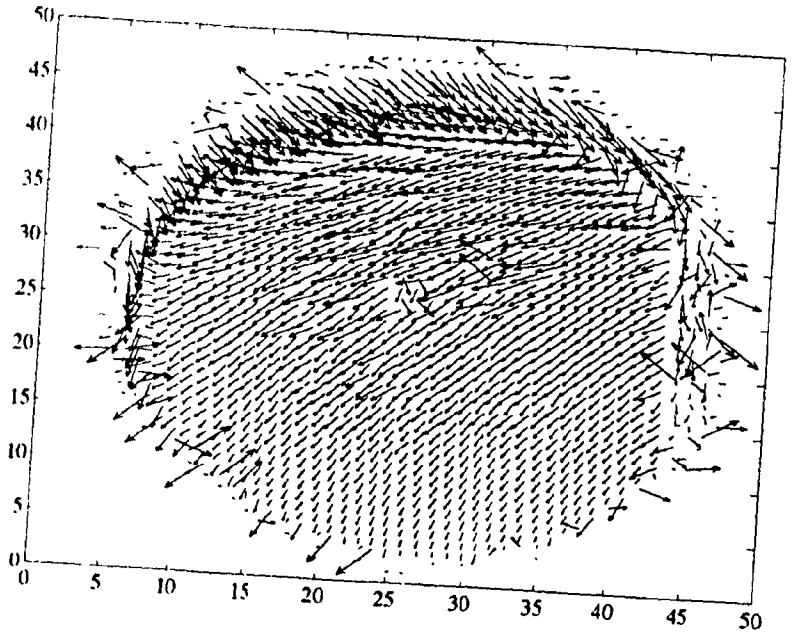


Figure 4.16: The projected 2D velocity field on u_2u_3-plane of the estimated velocity from the basic algorithm, where the neighborhood is 9×9.

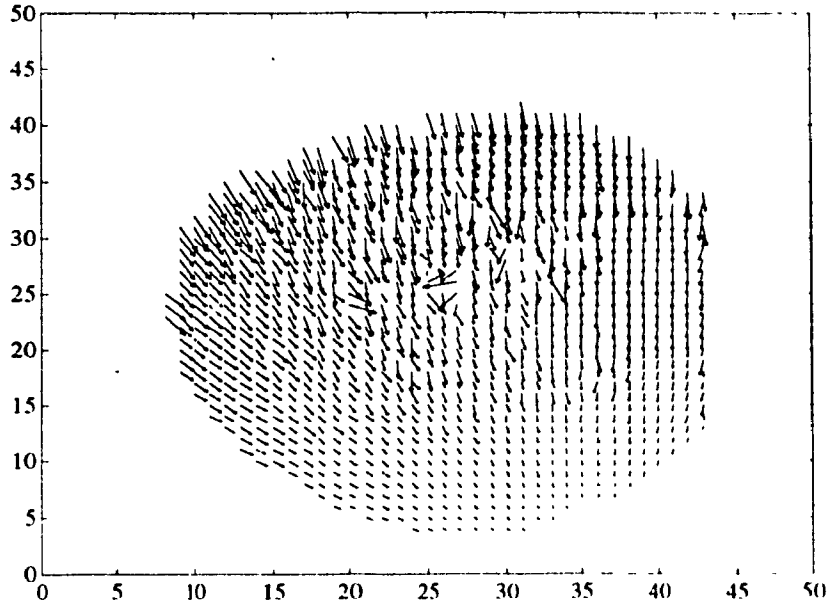


Figure 4.17: The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where $thgau = 0.00001$, $thfit = 20.0$, $thcond = 20.0$, $thvel = 20.0$.

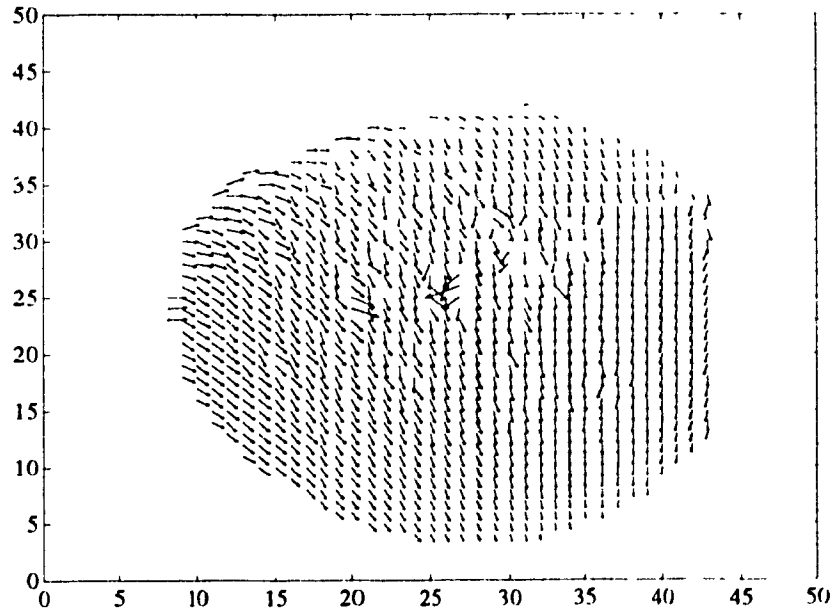


Figure 4.18: The projected 2D velocity field on u_1u_3-plane of the reliable velocity field, where $thgau = 0.00001$, $thfit = 20.0$, $thcond = 20.0$, $thvel = 20.0$.

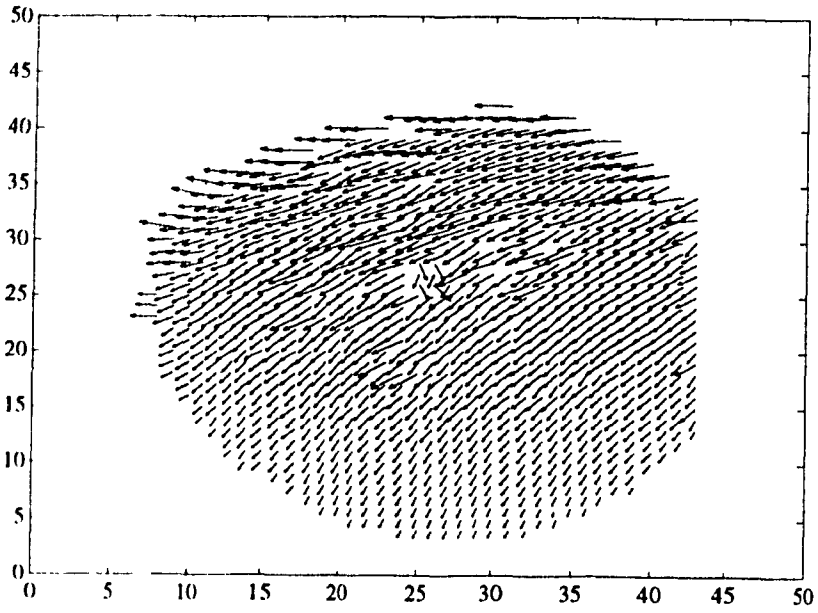


Figure 4.19: The projected 2D velocity field on u_2u_3-plane of the reliable velocity field, where $thgau = 0.00001$, $thfit = 20.0$, $thcond = 20.0$, $thvel = 20.0$.

4.14, the result from the LMS method is slightly worse than that from the proposed algorithm, especially near the boy's area, where the depth values contain more noise. It does not show any improvement near the boundaries since the movement is so large in this image sequence that there does not even exist one 3-tuple which well satisfies the velocity constraint equation near the boundaries. Besides, the LMS method is much slower than the proposed algorithm even after subsampling.

4.8.1 Doll

The doll image sequence was taken by the range sensor at McGill University's Research Center for Intelligent Machines. The range sensor was mounted on the end of the Puma robot arm. The robot arm can move freely so that we can simulate a 3D camera motion. Two images are shown in Figure 4.26. They are range images displayed as gray level images. The image size was 256 x 256. Three "cones" were used as markers. Two of them were positioned at the top shoulders of the doll, and the third one was put between one

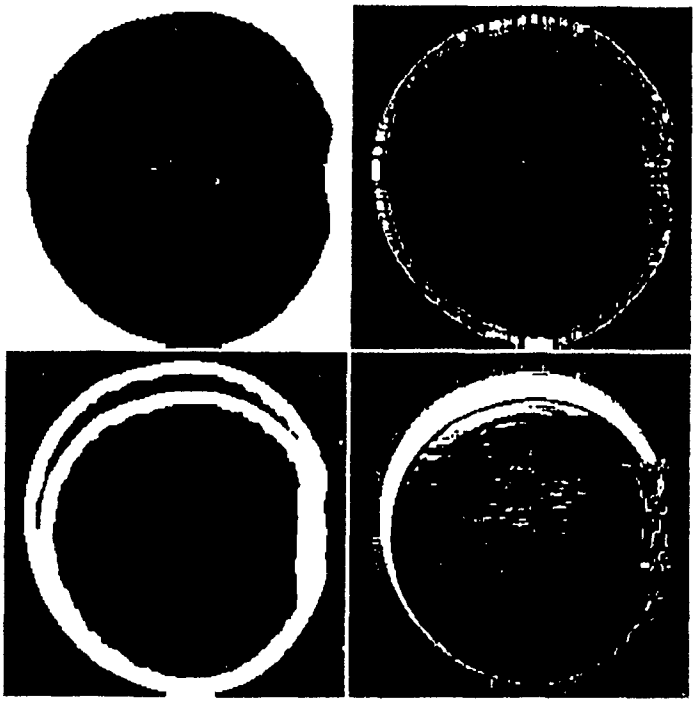


Figure 4.20: Upper left: binary image of Gaussian curvature. Upper right: binary image of condition number. Bottom left: binary image of fitting error. Bottom right: binary image of maximum allowable velocity.

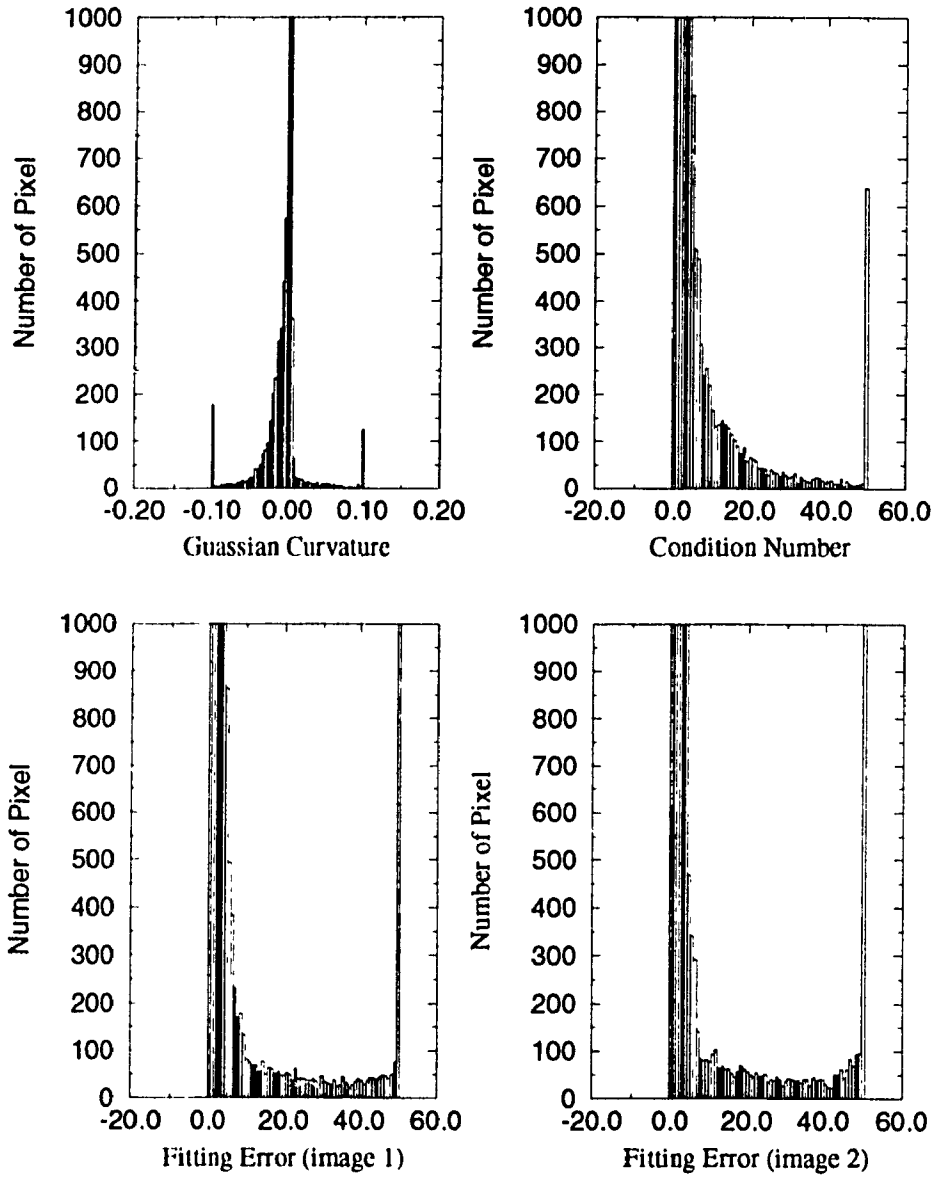


Figure 4.21: Histograms of the Gaussian curvature, condition number and fitting error (see text for details). Upper left: Gaussian curvature, threshold is 0.1. Upper right: condition number. Bottom left: fitting error of the first image. Bottom right: fitting error of the second image.

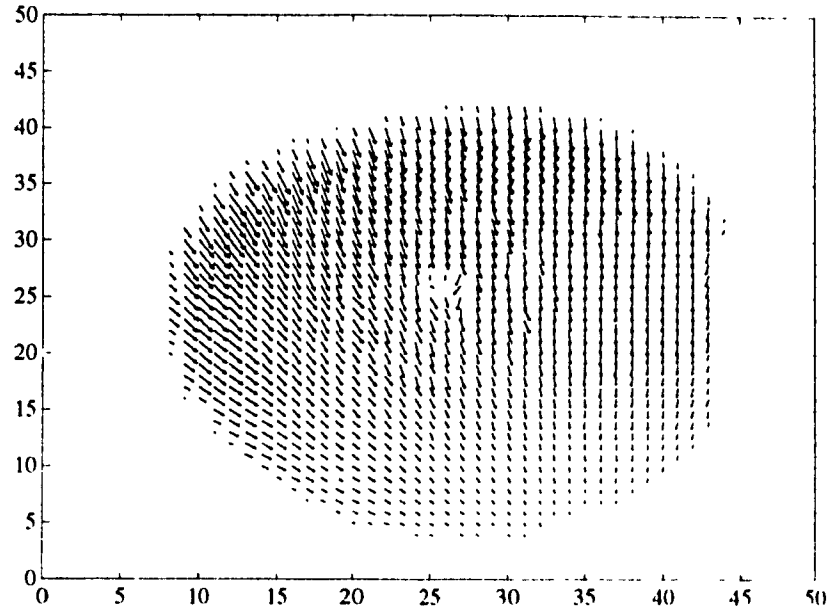


Figure 4.22: The projected 2D velocity field on u_1u_2-plane of the smoothed velocity field after 5 iterations, where $k = 16$.

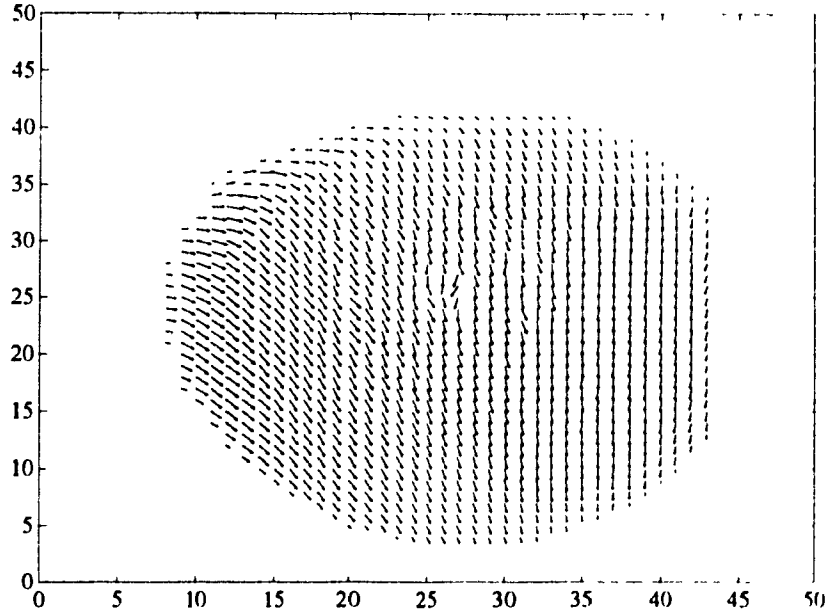


Figure 4.23: The projected 2D velocity field on u_1u_3-plane of the smoothed velocity field after 5 iterations, where $k = 16$.

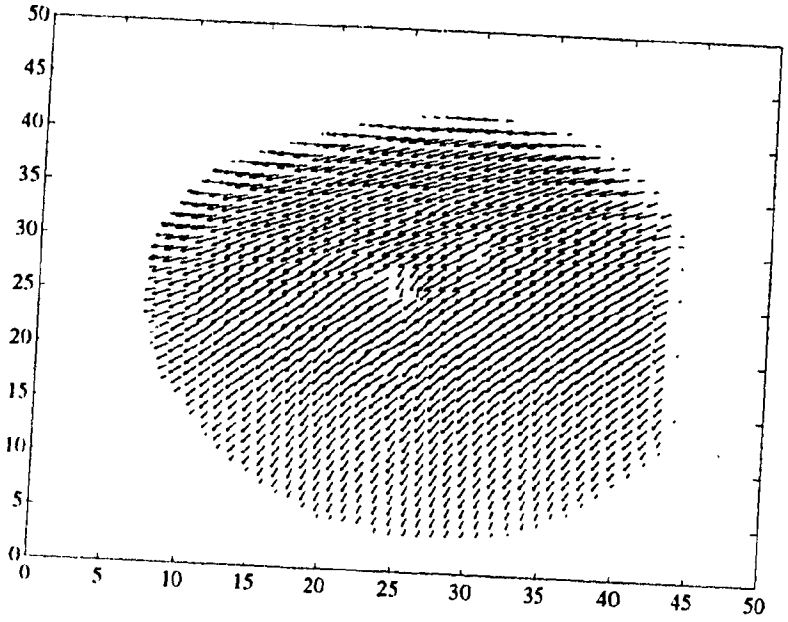


Figure 4.24: The projected 2D velocity field on u_2u_3-plane of the smoothed velocity field after 5 iterations, where $k = 16$.

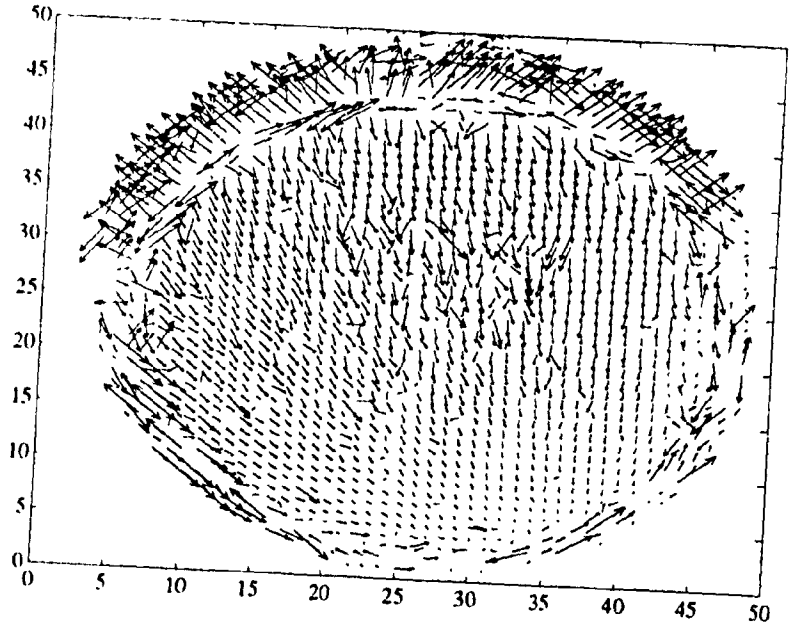


Figure 4.25: The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, where the neighborhood is 9×9.

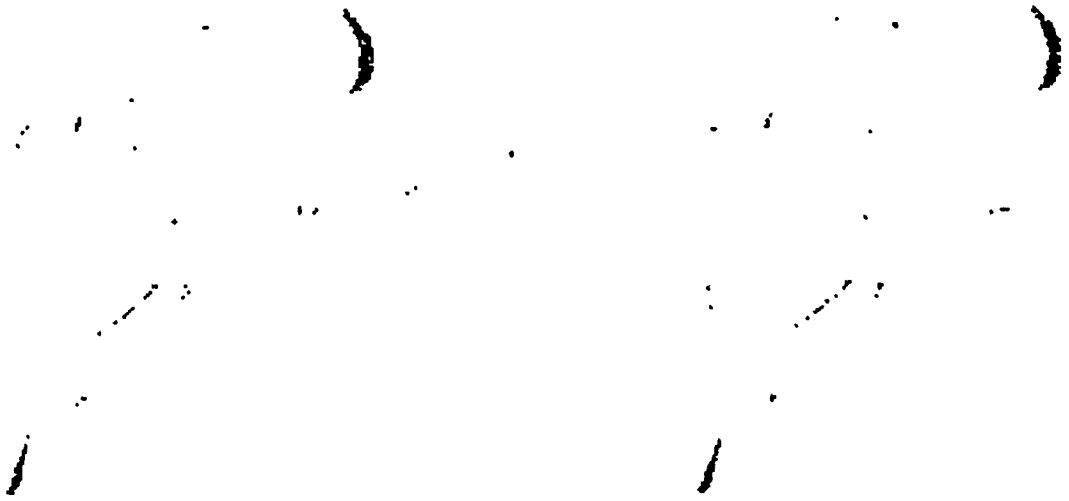


Figure 4.26: Top: the first doll image. Bottom: the second doll image.

arm and one leg of the doll. These markers were used to roughly validate the results. the positions of the markers were manually located in both images. It was found from these positions that the doll was roughly moved toward the upper left corner about 2 pixels, and was slightly rotated in anticlockwise direction.

Figure 4.27 shows the projected 2D velocity field on u_1u_2-plane of the estimated velocity field. Figure 4.28 shows the projected 2D velocity field on u_1u_2-plane of the estimated velocity field after reliability checking. There are many pixels on the doll whose velocities can not be estimated, since the doll is basically made of cylindrical surfaces. Figure 4.29 shows four binary images as we explained in last experiment. The projected 2D velocity field on u_1u_2-plane of the smoothed velocity field is shown in Figure 4.30. Figure 4.31 shows the projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method. In the figures, a velocity is set to 3 if it is exceeded for a better visualization. The doll image sequences are very noisy, therefore the initial estimated velocity field contains large number of unreliable estimates of velocities from both algorithms.

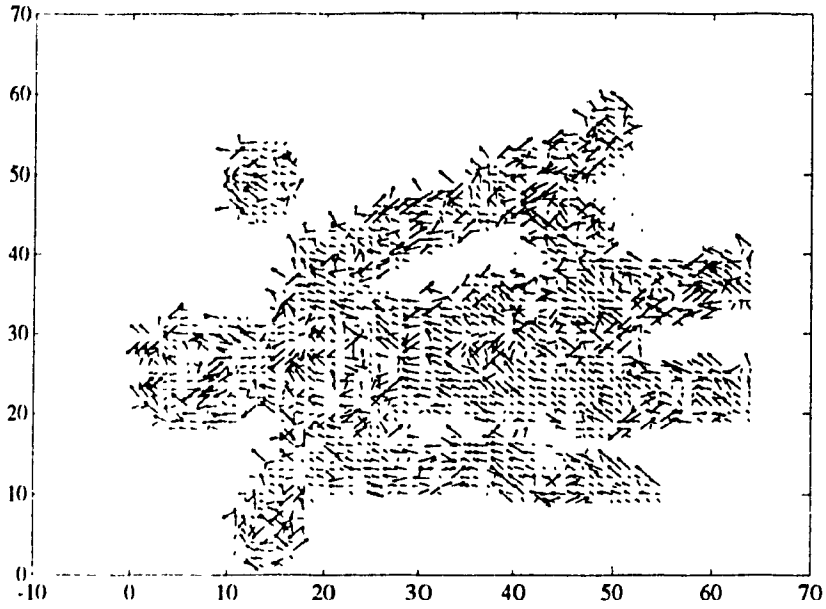


Figure 4.27: The 2D velocity field on u_1u_2-plane of the estimated velocity field, where the neighborhood is 9×9.

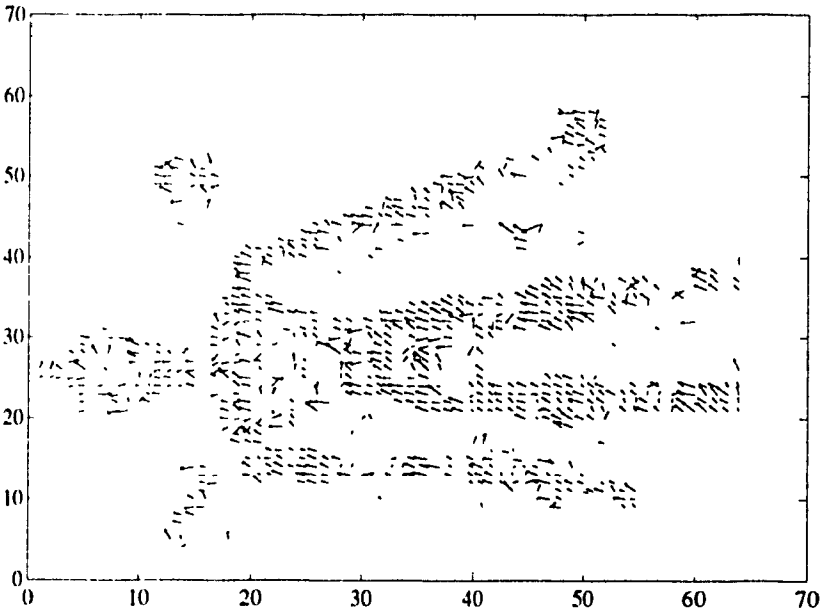


Figure 4.28: The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where $thgau = 0.0001$, $thfit = 100.0$, $thcond = 50.0$, $thvel = 3.0$.

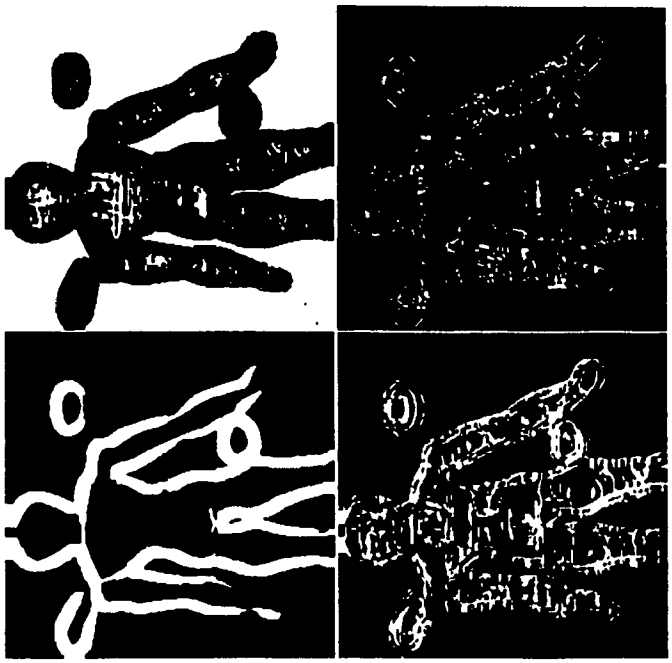


Figure 4.29: Upper left: binary image of Gaussian curvature. Upper right: binary image of condition number. Bottom left: binary image of fitting error. Bottom right: binary image of maximum allowable velocity.

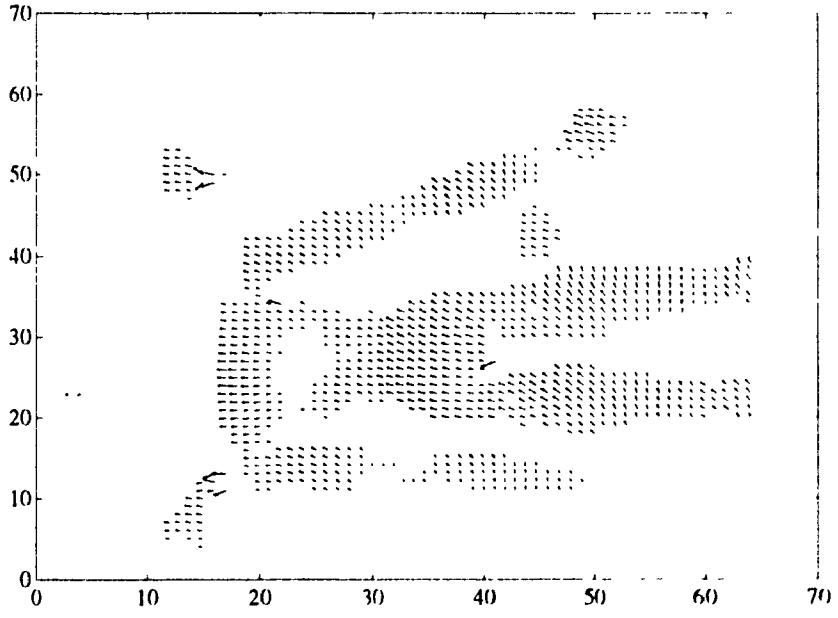


Figure 4.30: The projected 2D velocity field on u_1u_2-plane of the smoothed velocity field after 100 iterations, where $k = 16$.

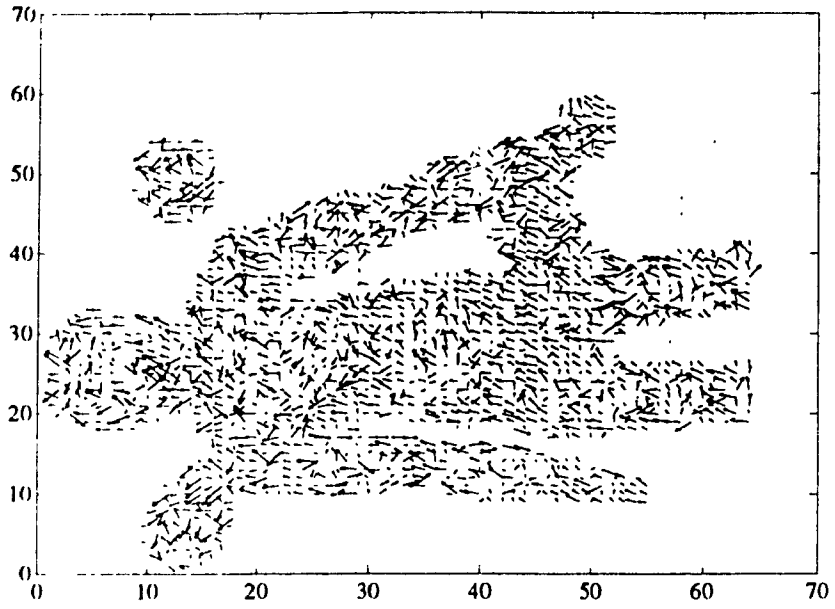


Figure 4.31: The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, where the neighborhood is 9×9.

4.8.2 A Piece of Paper

Figure 4.32 shows "a gray level encoded" range image of a crunched piece of paper. The image was also taken by NRC' video rate range finder. The image size was 256×256. Figure 4.33 is the wireframe plot of the same image. The second image is obtained by translating the first image one pixel each in the x, y and z directions. Figures 4.34 and 4.35 shows the projected 2D velocity field on u_1u_2-plane of the estimated velocity field before and after reliability checking. Figure 4.36 shows four binary images as explained in the balloon experiment. The projected 2D velocity field on u_1u_2-plane of the smoothed velocity field after 10 iterations is shown in Figure 4.37. Figure 4.38 shows the projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method. In the figures, a velocity is set to 5 if it is exceeded for a better visualization. For this scene, the LMS method gives much better results because the scene consists of approximately piecewise planar surfaces.

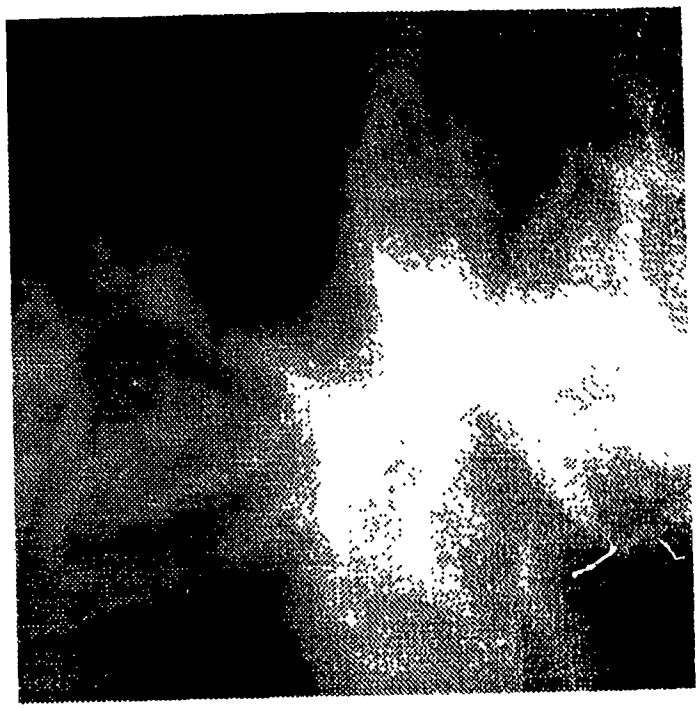


Figure 4.32: A piece of paper.

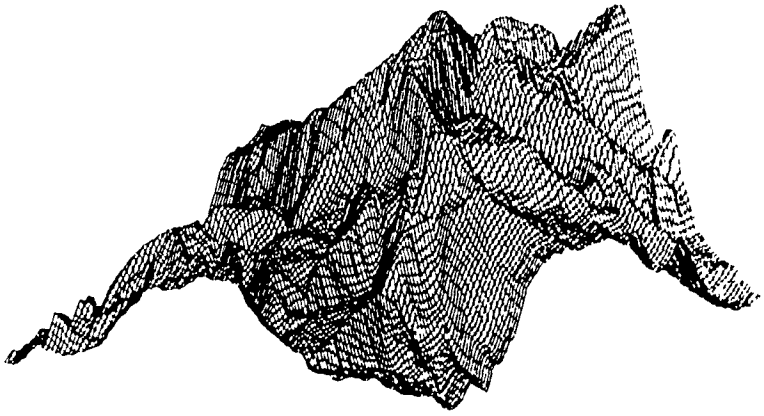


Figure 4.33: A piece of paper in wire frame plot

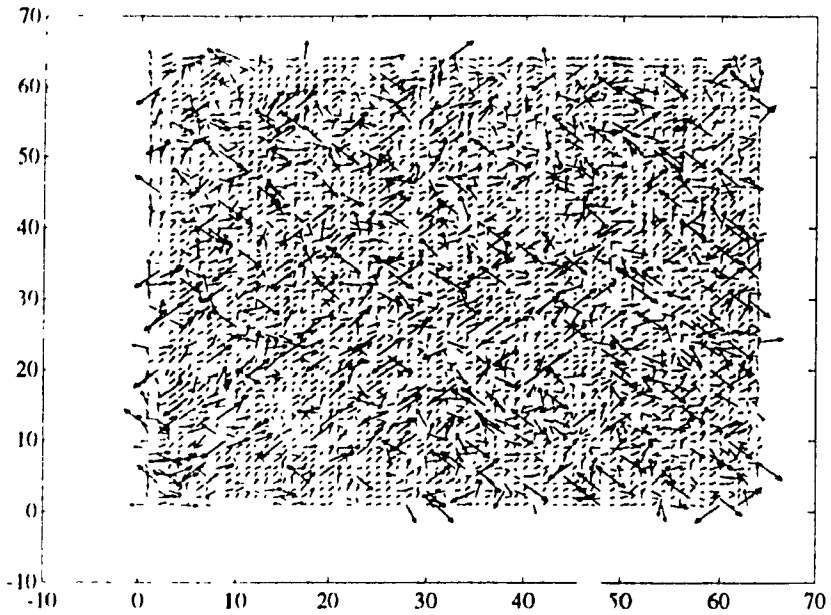


Figure 4.34: The projected 2D velocity field on u_1u_2-plane of the estimated velocity field where the neighborhood is 9×9.

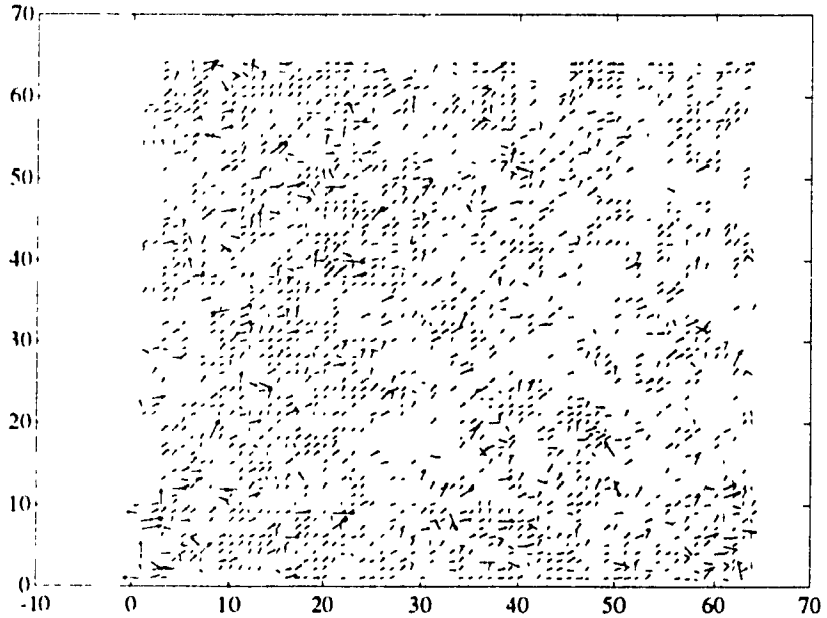


Figure 4.35: The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where $thgau = 0.001$, $thfit = 50.0$, $thcond = 20.0$, $thvel = 5.0$.

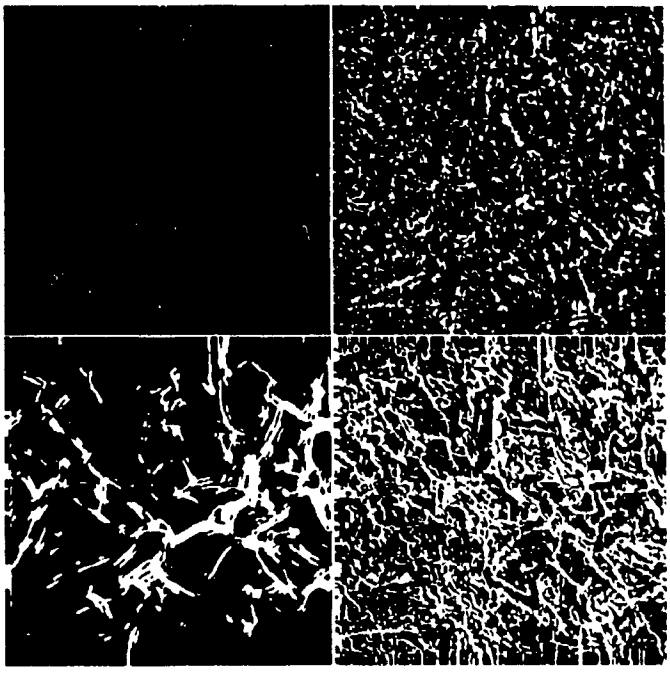


Figure 4.36: Upper left: binary image of Gaussian curvature. Upper right: binary image of condition number. Bottom left: binary image of fitting error. Bottom right: binary image of maximum allowable velocity.

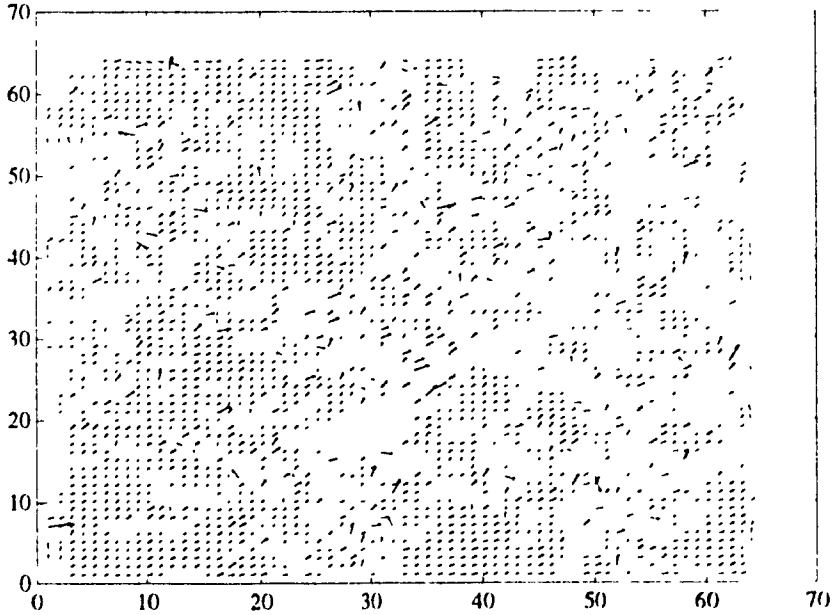


Figure 4.37: The projected 2D velocity field on u_1u_2-plane of the smoothed velocity field after 10 iterations, where $k = 16$.

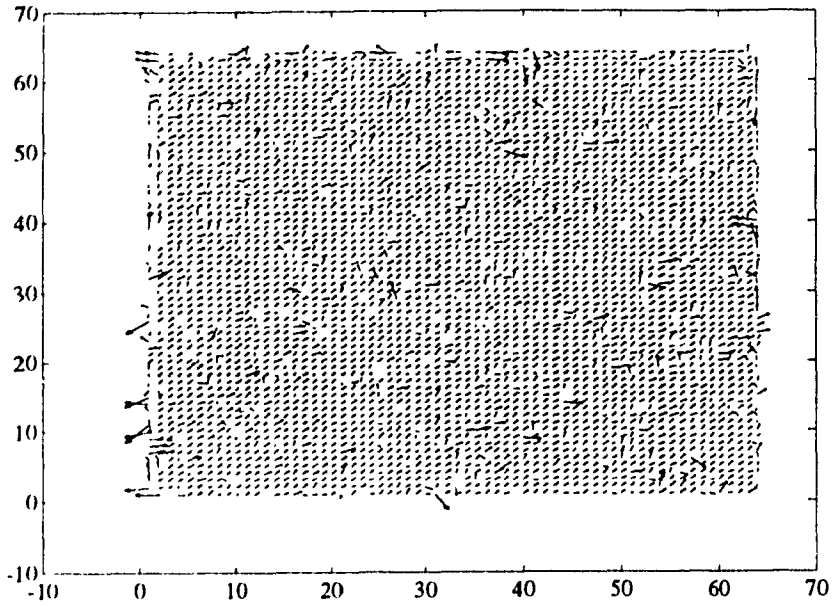


Figure 4.38: The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, where the neighborhood is 5×5.

4.8.3 Grip

In this experiment, the first image is shown in Figure 3.17. It was taken by the range finder at NRC. The second image was obtained by translating the first image one pixel in each of the x, y and z directions. Figures 4.39 and 4.40 show the projected 2D velocity fields on u_1u_2-plane of the estimated velocity field and the one after reliability checking. In the figures, a velocity is set to 5 if it is exceeded. Figure 4.41 shows four binary images as we explained before. Since most parts of the object consists of planar and conic surfaces, only velocities of parts of the object can be estimated. Figure 4.38 shows the projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method. It provides better results around the tip area.

4.8.4 Quadratic Surface

In this experiment, images were generated synthetically. An analytic quadratic surface

$$z = 0.1 + 0.02x + 0.03y + 0.01xy$$

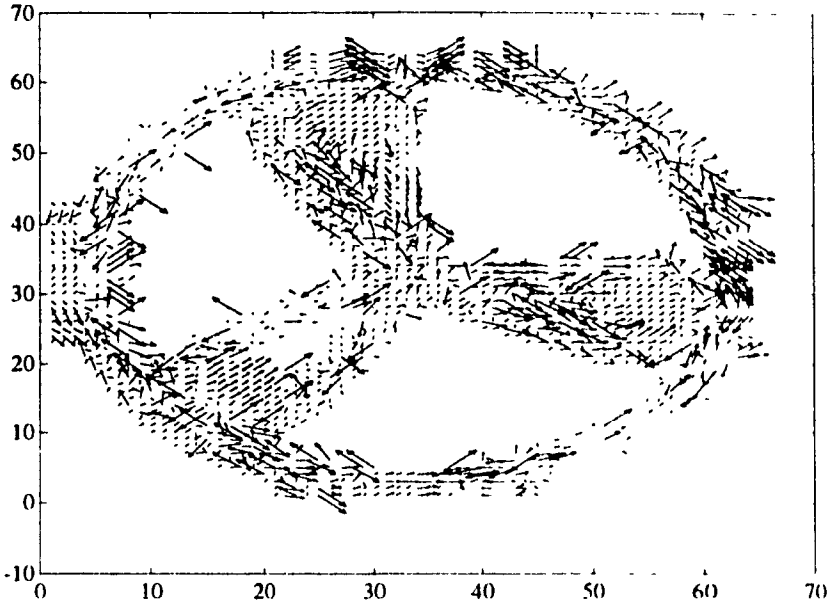


Figure 4.39: The projected 2D velocity field on u_1u_2-plane of the estimated velocity field, where the neighborhood is 9×9.

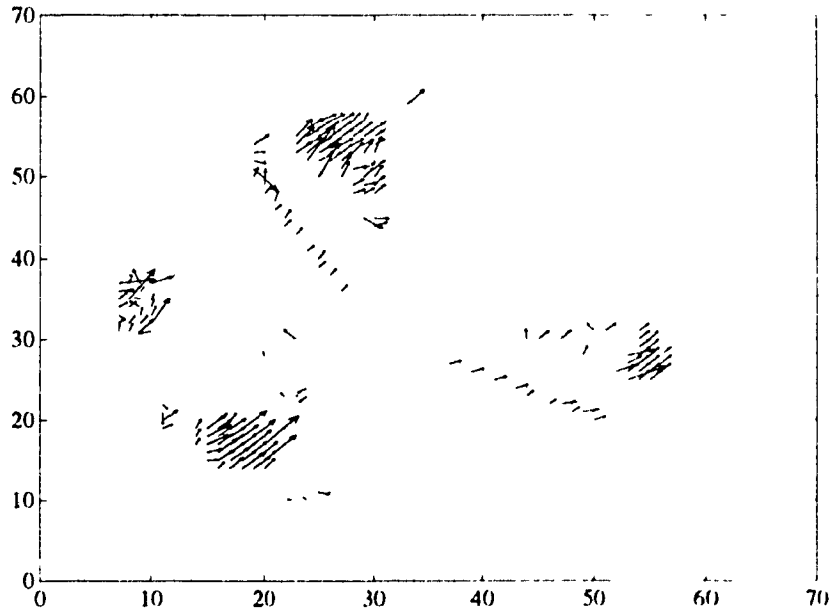


Figure 4.40: The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where $thgau = 0.0001$, $thfit = 150.0$, $thcond = 50.0$, $thvel = 5.0$.

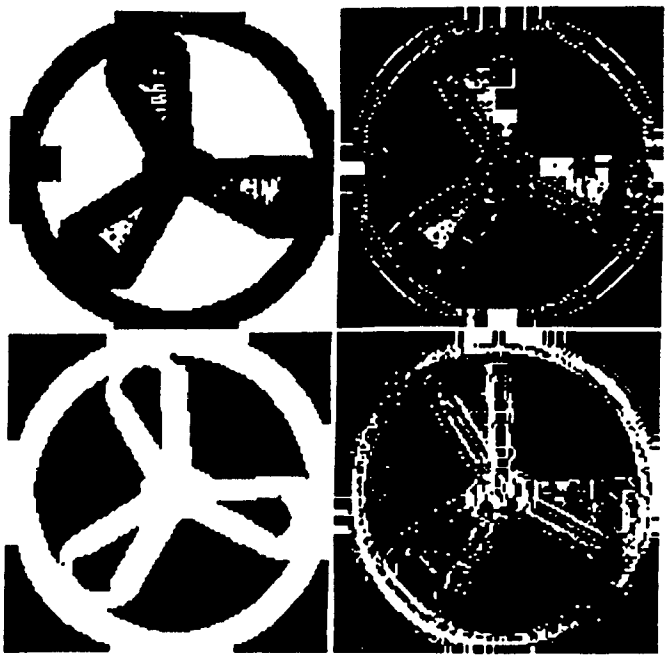


Figure 4.41: Upper left: binary image of Gaussian curvature. Upper right: binary image of condition number. Bottom left: binary image of fitting error. Bottom right: binary image of maximum allowable velocity.

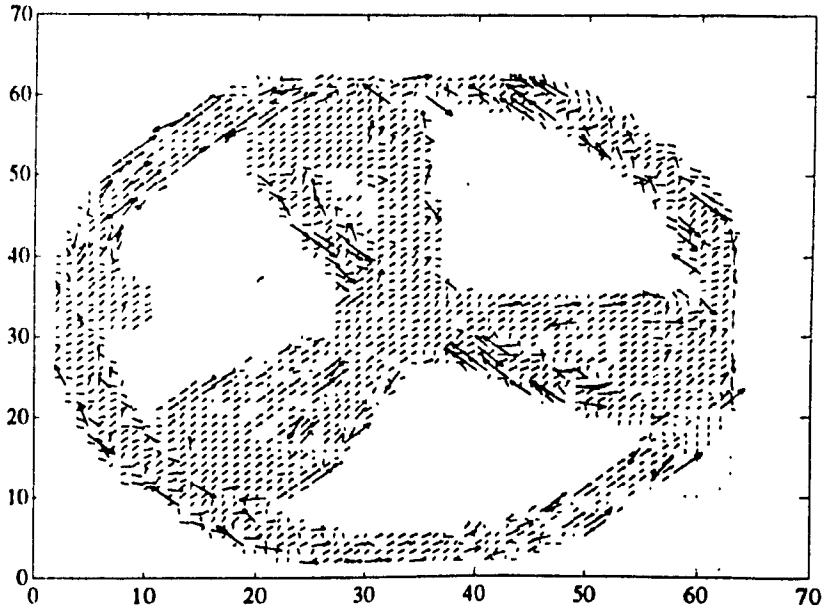


Figure 4.42: The projected 2D velocity field on u_1u_2- plane of the estimated velocity field using the LMS method, the neighborhood is 7×7.

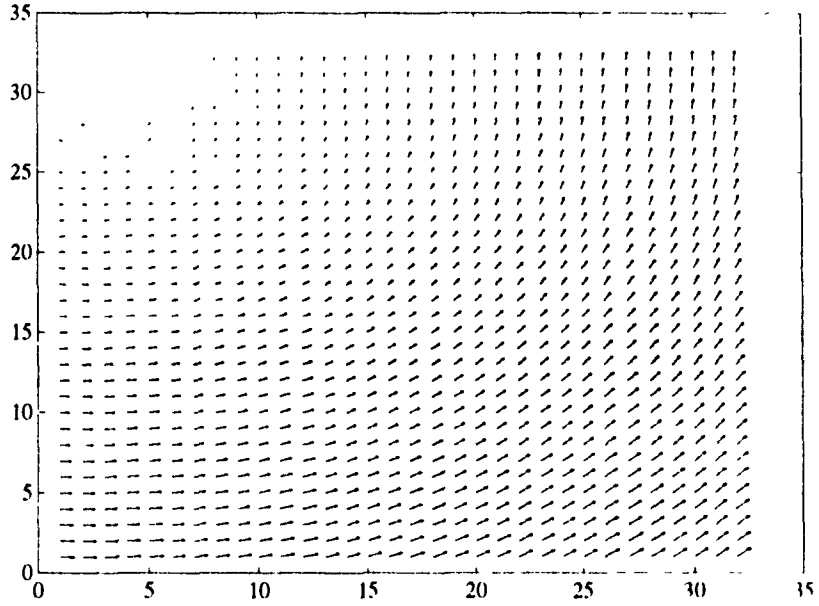


Figure 4.43: The projected 2D velocity field on u_1u_2-plane of the reliable velocity field, where the neighborhood is 5×5.

was sampled to create the first image, then the second image was obtained after rotating the surface 2 degrees around the z axis by 3D motion program. The image size was 128×128. Figure 4.43 shows the projected 2D velocity field on u_1u_2-plane of the estimated velocity field. The rotation around the origin can be clearly seen. Since the surface is a quadratic with nonzero Gaussian curvature, the whole velocity field can be reliably estimated. Figure 4.44 shows the projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method. In this scene, the LMS method gives a very similar result because the surface is changing slowly.

4.8.5 Half Sphere and Half Ellipsoid

In this experiment, images were also generated synthetically. The first image is shown in Figure 3.8. The image size was 128×128. The second image was obtained by translating the object one pixel in each of the x, y and z directions, respectively, by the 3D motion program. Figures 4.45 and 4.46 show the projected 2D velocity on u_1u_2-plane of the

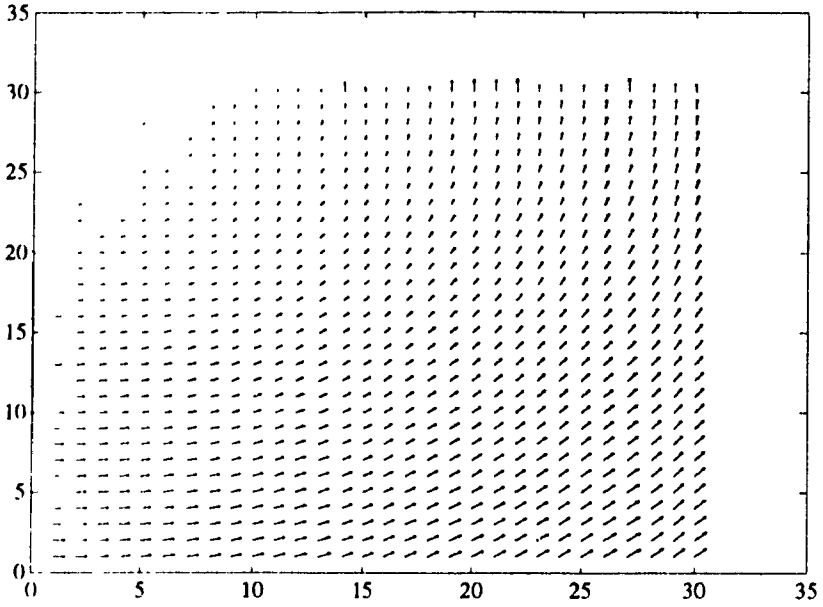


Figure 4.44: The projected 2D velocity field on u_2u_3-plane of the estimated velocity field using the LMS method, where the neighborhood is 5×5.

estimated velocity field using the proposed algorithm and the LMS method, where a velocity is set to 5 if it is exceeded. The LMS method gives better results around the edge between two different surfaces, while the algorithm using the second order derivatives provides better estimates around the boundary since the surfaces change fast there.

The experiment was repeated for rotation. The second image was obtained by rotating the surface 2 degrees around the z axis passing through the center point of the image by the 3D motion program. Figures 4.47 and 4.48 show the projected 2D velocity fields on u_1u_2-plane of the estimated velocity field using the algorithm and the LMS method. Since the surface contains a half sphere, and was rotated around the center of the sphere, velocities are unable to be estimated for the half sphere.

4.9 Conclusion

In this chapter, an new motion estimation algorithm has been proposed. This algorithm is able to uniquely determine the 3D velocity for each point except points with zero Gaussian

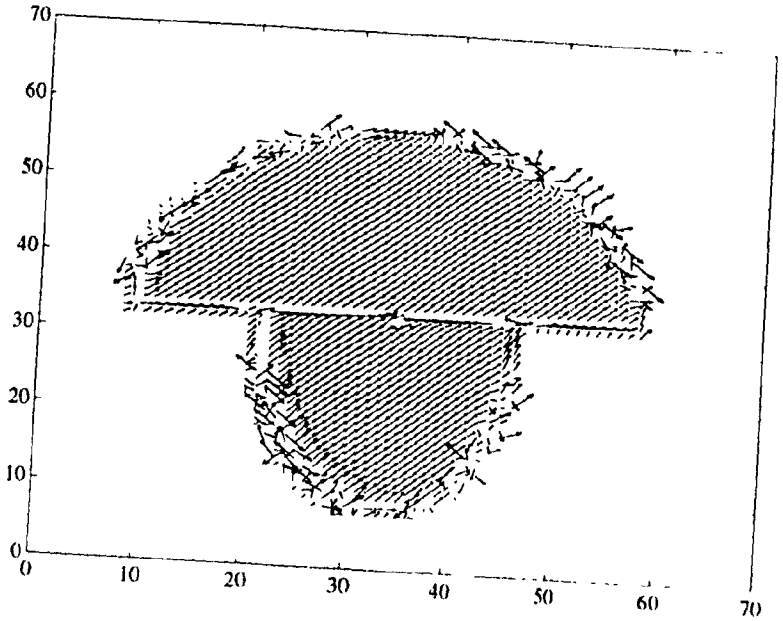


Figure 4.45: The projected 2D velocity field on u_1u_2-plane of the estimated velocity field, where the neighborhood is 5×5.

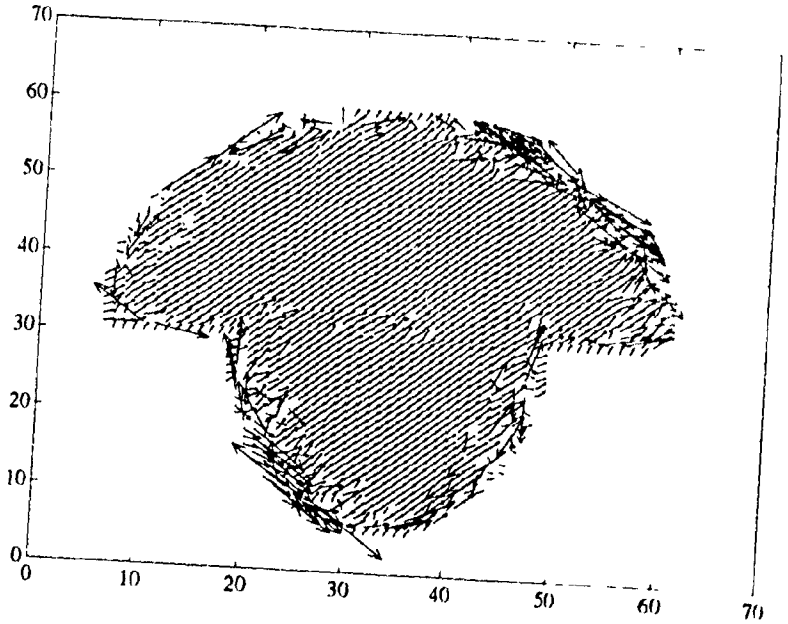


Figure 4.46: The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, where the neighborhood is 5×5.

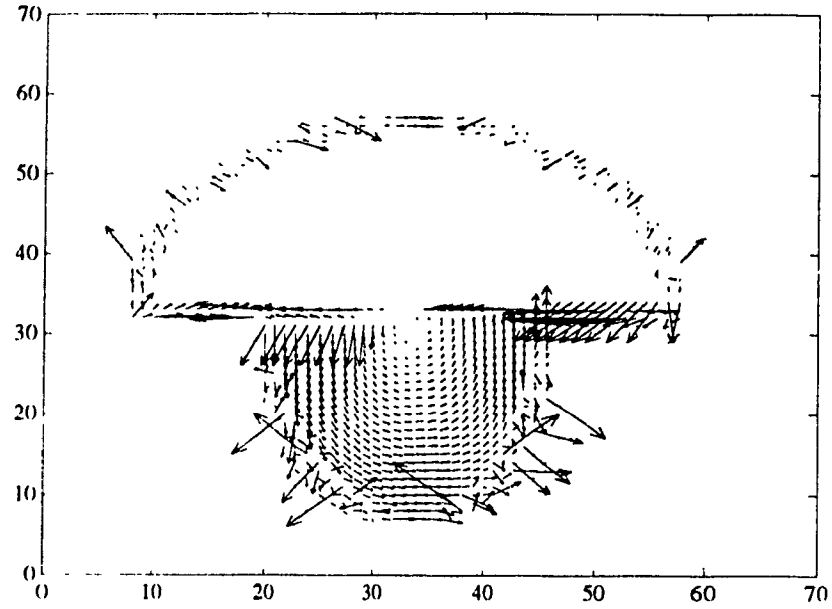


Figure 4.47: The projected 2D velocity field on u_1u_2-plane of the estimated velocity field, where the neighborhood is 5×5.

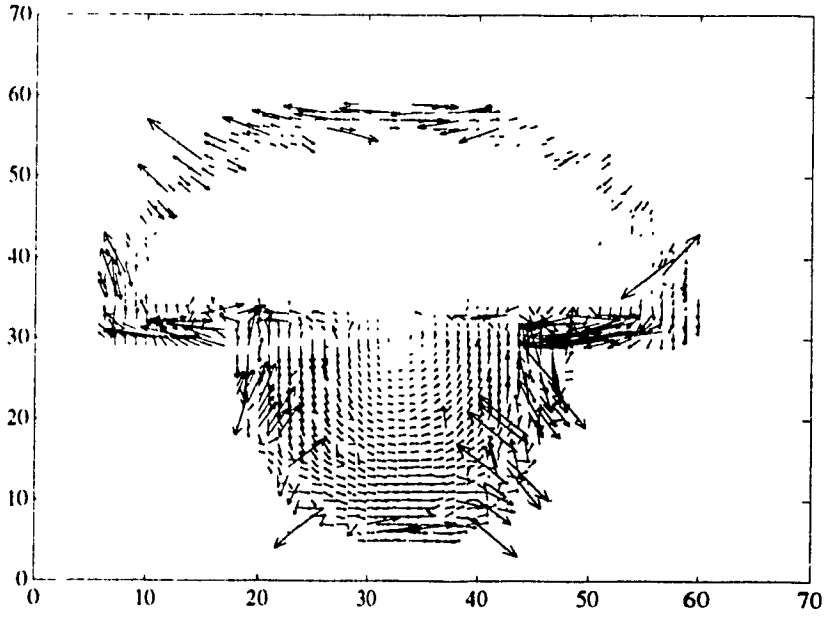


Figure 4.48: The projected 2D velocity field on u_1u_2-plane of the estimated velocity field using the LMS method, the neighborhood is 5×5.

curvatures. The algorithm can be derived from either gradient-based principle or similarity matching, which are usually considered as two totally different processes. There are some areas where 3D velocities can not be estimated locally. This problem has been coped with by introducing reliability checking and interpolation. If the measured velocity of a point is not reliable, then it is inferred from its neighboring points.

Several other possible techniques, such as least squares methods, clustering, Hough transform, have also been discussed. The least median squares method has been implemented in order to compare the performance with the proposed algorithm. The results have shown that the proposed algorithm is better than the LMS method for noisy and curved surfaces. The LMS method is better for slowly changing surfaces. However the LMS method is much more time consuming.

Chapter 5

Analyzing 3D Velocity Field by Matching Surface Features

A method to estimate 3D velocity field through token (feature) matching is proposed in this chapter. Principal curvatures are chosen to be the features since they are invariant to both rigid motion and different parametrization of a surface. The 3D velocity at point P is obtained by locating a matching point whose distribution of principal curvatures in its neighborhood is similar to that of point P. If point P is on a line edge, then its full velocity can not be determined because of aperture problem. However, its vernier velocity can be determined. Two algorithms are described to cope with this problem.

The advantage of this feature matching approach is that it can estimate velocities of both corner points as well as points on smooth curved surfaces and vernier velocities of line edge points, while the method discussed in the previous chapter is unable to deal with corner points and line edge points. For objects like polyhedra, it is very important to estimate motion on corners and line edges. The disadvantage of the feature matching method is that it is computationally intensive compared with the approach in the last chapter. It is possible to combine them, for example, one technique being used for line edges and corners, and the other technique for smooth surfaces.

5.1 Introduction

The methods discussed in the last chapter work very well when a scene contains curved smooth surfaces. However, they may fail in estimating motion of polyhedra since the motion of polyhedra is perceived from that of line edges and corners of the polyhedra, and the analytic method proposed in the last chapter does not provide good estimates near line edges and corners.

In this chapter, a method will be discussed to estimate a 3D velocity field through token (feature) matching. A set of tokens (features) is extracted from both images, points with similar features in both images are considered as matching pairs, and the displacement within the matching pair is taken as the estimate of the 3D velocity at the point. The idea behind the token matching is much more straightforward and intuitive than that of the analytic method. Naturely, one may ask the following questions:

1. What are the appropriate features?
2. How to reliably extract the chosen features?
3. What are the criteria for matching?
4. How to deal with one-to-many or one-to-zero matchings?

These questions will be discussed in the remaining part of the chapter.

Related work can be found in the calculation of optical flow [7, 21, 43, 95], and in stereopsis [42, 76].

5.1.1 Motion Invariance

Since it is desirable to compute a rather dense 3D velocity field, therefore, features should be pixel-based or small neighborhood-based. There are not many choices for pixel-based features, except if different kinds of registered images are used. For example, if both range and intensity images are used, then for each pixel, there are two candidate features: gray level and depth value. If combined with RGB images, then there exist five candidates: gray level, depth value, red value, green value and blue value. Any combinations of these

five candidates can be used as pixel features. Based on matching of features, a dense velocity field can be obtained. In this work, only one kind of image: range image, is considered, thus the only pixel feature is depth value, which is obviously not suitable for feature matching since the depth of a point changes with motion. Neighborhood-based features have to be considered.

Computing surface features for feature matching purposes is the first step towards the final goal of the motion understanding. Objects are viewed from different directions as they move in front of a camera or as a camera moves past the objects. To handle the problem of arbitrary viewing directions, viewpoint invariant surface features are needed.

As described by [13], a quantity is invariant with respect to a group of transformations if those transforms do not change its value. For example, the length of a 3D line segment does not change under the group of rotation transformations, and is therefore said to be rotationally invariant, but the length of a 3D line segment projected into a 2D image plane does change under the rotation and is not invariant. In general, opaque, rigid, physical objects do not possess explicit surface features that are visible from any viewing angle. There are almost always degenerate viewing angles in which visible object features are radically different. For example, consider an object as simple as a cylinder. A flat planar surface with a circular boundary is visible when looking down the axis of a cylinder. In contrast, a curved surface with rectangular projected boundary is visible when looking perpendicular to the axis direction. There are no explicit invariant features even in this simple area. In this example, although the same object is viewed, in fact, the totally different parts of the object are viewed, and one visible part in one viewing direction is not visible in another viewing direction. This situation will not be discussed in this thesis. By invariant features, one means that features are invariant if they are visible. In motion literature, when a part of a surface is visible at one time instant and invisible at another time instant, it is said that this part of the surface is occluded. If a part of the surface is invisible at one time instant and visible at another time instant, it is said that this part of the surface is uncovered. In this sense, the term "invariant" means "invariant if neither occluded nor uncovered" within the interval of the observation.

A nonrigid moving object may not possess invariant surface features even if all the

surfaces are visible at different time instants. The shape of the object may be totally different after a certain time. In this chapter, it is assumed that this will not happen in a short time. Therefore, invariant features are only considered under different viewing angles, or under euclidean rigid motion. In this sense, a visible-invariant surface feature is a quantitative feature of visible surfaces that does not change under the set of viewing transformations (rigid motions) that do not affect the visibility of that region.

It is equally important that surface features are invariant to changes in the parametrization of a surface. When a visible, smooth, curved surface is sampled on a rectangular image grid from two different viewpoints, the effective grid parameterizations of the surface in the two corresponding range images are different. Hence, numerical visible-invariant surface quantities must also be invariant to changes in surface parameterization.

In this thesis, maximum and minimum principal curvatures (K_1 and K_2) have been chosen as surface features and collectively referred to *principal curvatures*. The definitions of these terms are discussed in Appendix B. When a surface patch is visible, its principal curvatures are invariant not only to rigid motion, but also to changes in surface parameterization. (see Appendix A in [13]).

5.2 Computing Surface Features

Although a nice theory exists for continuously differentiable surfaces as seen from the last section, scenes in real world are normally not so smooth, and they usually contain clearly definable objects with distinguishable boundaries between the sides or facets of each object. These surfaces are usually not suitable for partial differentiation. Furthermore, range images are quantized in the x, y and z directions. A variety sources of noise are introduced during scanning, which make the computation of surface features nontrivial, even though it appears straightforward theoretically. Hence an appropriate computation method has to be developed.

A range image represents a surface with the height $F(x, y)$ above the "support" plane defined by the two coordinates (x, y). This representation is known as the *graph surface* representation and the surface is sometimes called a *Monge patch*. A 3D point on

the surface is given by $\mathbf{X}(x, y) = (x, y, F(x, y))$. Computation methods will be studied which are based only on the graphic representation.

There exist several kinds of approaches to calculate curvatures. A comparison study of various algorithms can be found in [37]. Curvature measurement of arbitrary 3D objects can be found in [97]. Some of the algorithms currently being used include:

1. The formal approach, by direct application of the formulas given in differential geometry, after conversion to difference equations.

2. Surface fitting from which the partial derivatives for the formulas can be obtained.

3. Numerical estimates which compute curvatures by numerical simulation of the derivation process in differential geometry.

The formal approach in case 1 is computationally the shortest of the above three approaches since it is basically a matter of approximating the formulas by numerical methods, given the samples $F(x, y)$ and x, y. The formulas for curvatures are quite complex, especially for the principal curvatures κ_1 and κ_2, and require the first and second order partial derivatives as a starting point. Since images always contain some noise, as well as discontinuities (such as jump edge, roof edge etc.), the derivatives can contain considerable errors, and it is rather difficult to estimate the accuracy of the results.

In case 2, analytic functions, such as orthogonal polynomials, second degree polynomial function, B-spline etc., are fitted to the range values in the "neighborhood" of the point of interest, from which the required derivatives are obtained. However, at jump edges, the fitting technique can cause problems.

Ittner and Jain [56] and Hoffman and Jain [48] produce curvature estimates at a point p by considering the orientation between it and its neighbors. The computation method proposed by Kasvand [62] essentially simulates the derivation of curvatures in differential geometry. It provides a better insight into "what is going on" and various intermediate results are available from which errors can be estimated. We use this method in our work. A brief discussion will be given here, the details can be found in [62].

Surface Normal

The numerical computation of curvatures is based on surface normals. The numeric computation of the surface normal image $\mathbf{N}(x,y)$ does not represent a major problem since $\mathbf{N}(x,y)$ is only the first order difference in $F(x,y)$. A Sobel operator is used to compute gradients resulting in the image $F_x(x,y)$ and $F_y(x,y)$. The surface normal vector $\mathbf{N}(x,y)$ is then obtained from:

$$\|\mathbf{N}\| = \sqrt{F_x^2 + F_y^2 + 1} \tag{5.1}$$

where the direction cosines $(\mathbf{N_x}, \mathbf{N_y}, \mathbf{N_z})$ of the normal N are

$$\begin{aligned} \mathbf{N_x} &= -\frac{F_x}{\|\mathbf{N}\|} \\ \mathbf{N_y} &= -\frac{F_y}{\|\mathbf{N}\|} \\ \mathbf{N_z} &= \frac{1}{\|\mathbf{N}\|} \end{aligned} \tag{5.2}$$

The gradient magnitude Z_m and the orientation θ and tilt $\hat{\theta}$ angles of N (see Figure 5.1(a)) are obtained from

$$Z_m = \sqrt{F_x^2 + F_y^2} \tag{5.3}$$

$$\theta = tg^{-1}\frac{F_y}{F_x} \tag{5.4}$$

$$\hat{\theta} = tg^{-1}\frac{1}{Z_m} \tag{5.5}$$

Principal Curvatures and Their Directions

The numerical computation of the surface curvatures $\kappa_1(x,y)$ and $\kappa_2(x,y)$, and the corresponding unit vectors $\mathbf{U_1}(x,y)$ and $\mathbf{U_2}(x,y)$ is rather complicated. The simulation for the derivation procedure is mainly outlined as follows:

1. As shown in Figure 5.1(b), the surface normal N and the direction vectors $\mathbf{U_1}$ and $\mathbf{U_2}$ for the κ_1 and κ_2 form an orthogonal triplet. The tangent vector t is in the plane formed by $\mathbf{U_1}$ and $\mathbf{U_2}$, and it is orthogonal to N. When the tangent vector t is rotated around the normal vector N, the corresponding surface curvature value at the point P in the direction of t goes through maximum and minimum values. The maximum curvature

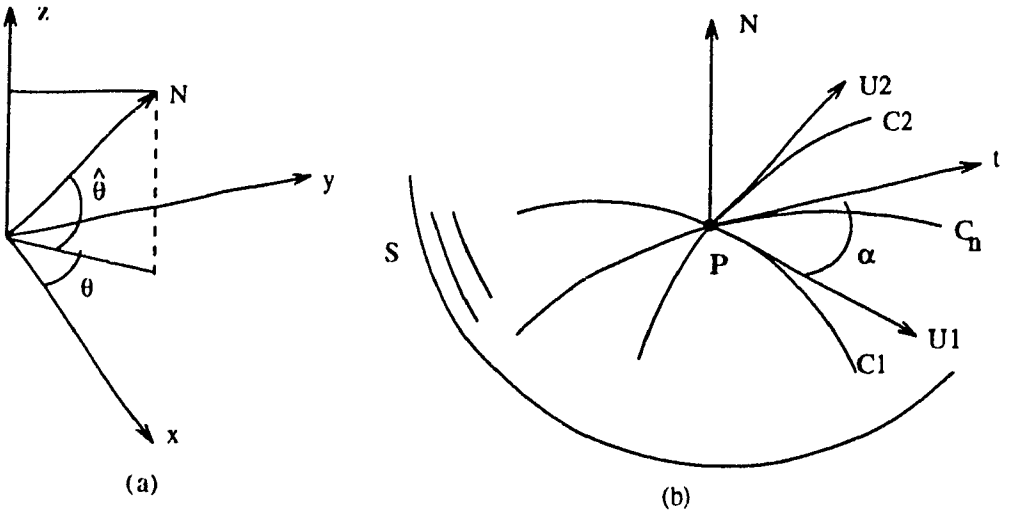


Figure 5.1: (a) The surface normal N, the orientation angle θ and the tilt angle $\hat{\theta}$. (b) The surface element in the differential geometry. The orthogonal unit vectors $\mathbf{U_1}, \mathbf{U_2}$ and N, define the orientation of the surface element, where N is the normal vector, $\mathbf{U_1}$ is the direction of maximum surface curvature, and $\mathbf{U_2}$ is the direction of minimum surface curvature.

value is κ_1 and the minimum is κ_2, and these are the quantities wanted. At any pixel in $F(x, y)$ image, the normal vector is known from previous calculations.

2. Consider a point P on the surface of $F(x, y)$, the (x, y, z) coordinates of P are x and y in the image plane and $F(x, y)$ is z. These are indicated as (x_0, y_0, z_0) in Figure 5.2. Place a disk or a circle such as C_1 with radius R at the point P and tangent to the $F(x, y)$ surface. The surface normal N is the normal of C_1. We want the values of the surface normals around the periphery of C_1 since the angle difference $d\beta$ between the surface normals at the diametrically opposite points on C_1 is proportional to the normal curvature κ_n (see Figures 5.1(b), 5.2, 5.3). Formally,

$$\kappa_n = \lim_{ds \to \infty} \left(\frac{d\beta}{ds}\right) \tag{5.6}$$

where ds in the present case is equivalent to $2R$ or the diameter of the disk C_1, but see further on. The problem thus consists of computing the angles of the normal vectors around the periphery of C_1 and finding the maximum and minimum angle differences.

3. The circle C_1 at point P projects into an ellipse C_2 on the image plane, see Figure 5.2. The general equation for an ellipse is

$$x = a\cos\alpha \tag{5.7}$$

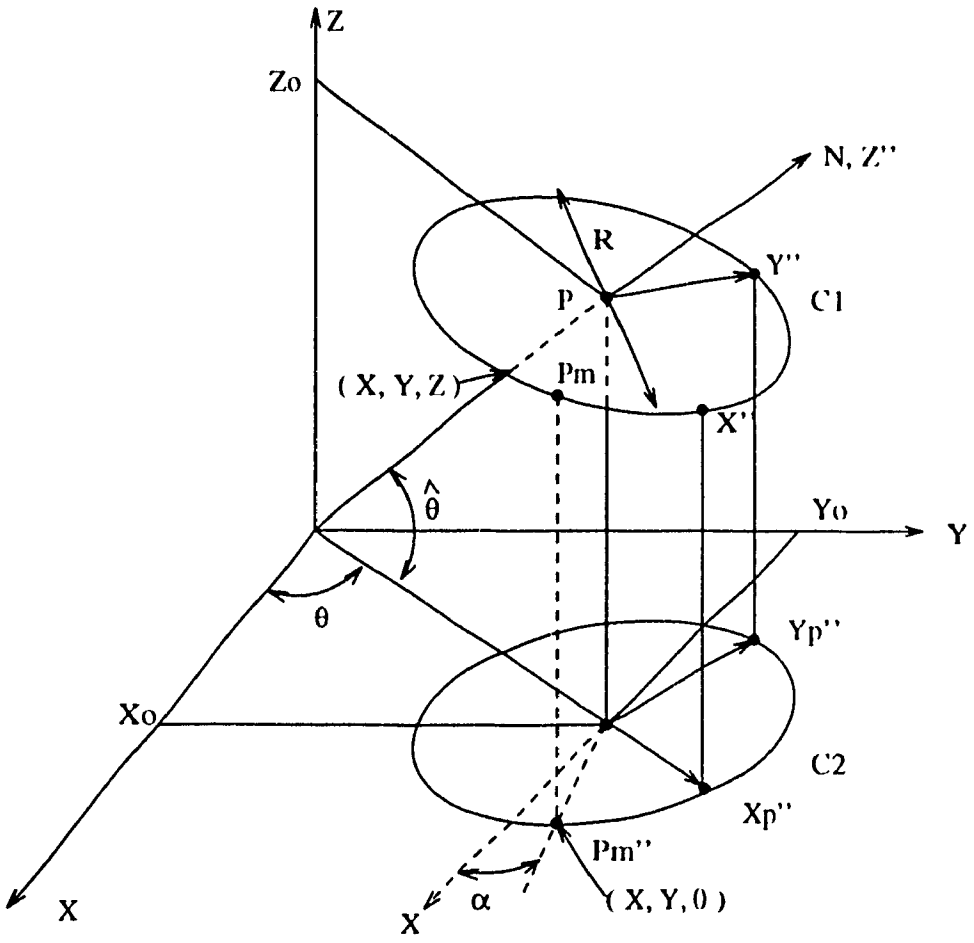


Figure 5.2: The disk C_1 and its projection C_2 in the xy space.

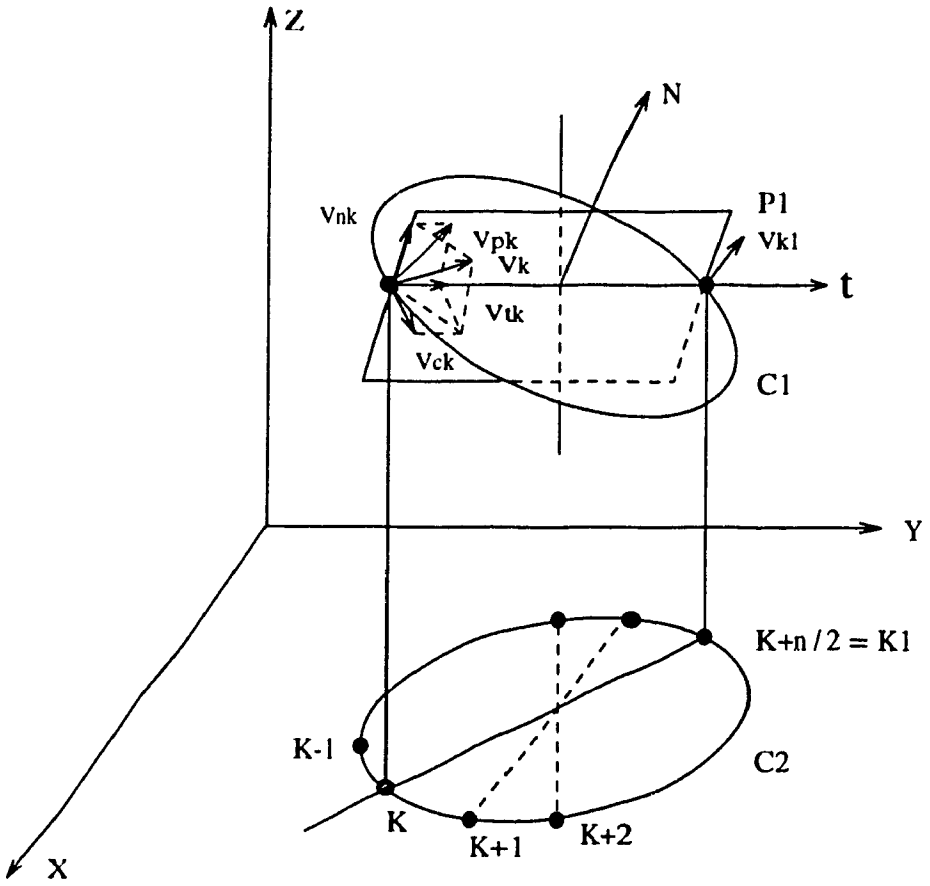


Figure 5.3: The normal vectors around the periphery of C_1.

$$y = b sin\alpha \tag{5.8}$$

where a is the major axis, b is the minor axis, and α is the angle of the radius vector with respect to the x-axis. The minor axis of the ellipse C_2 is in the direction θ and its magnitude depends on the tilt angle $\hat{\theta}$. Thus, the ellipse rotates as a function of θ and its minor diameter varies as a function of $\hat{\theta}$ ($a = R$ and $b = Rsin\theta$). A point P_m at (x, y, z) on C_1 transforms to a point P'_m at $(x, y, 0)$ on C_2. The coordinates of the point P''_m are

$$x = x_0 + Rsin\theta cos(\alpha - \hat{\theta})cos\hat{\theta} - Rsin(\alpha - \hat{\theta})sin\theta \tag{5.9}$$

$$y = y_0 + Rsin\theta cos(\alpha - \hat{\theta})sin\hat{\theta} - Rsin(\alpha - \hat{\theta})cos\theta \tag{5.10}$$

In practice it is easier to transform the point P'_m at $\alpha = 0$ ($x = x_0 + R, y = y_0, z = 0$) to the circle C_1, define a rotating coordinate system around the normal vector N and then transform the point P_m back into the xy plane. This allows the use of uniform angular displacements (α') around the periphery of C_1 (around N). The Euler transforms are, T_1 = rotation around z-axis by θ, T_2 = rotation around y'−axis by $90^\circ - \hat{\theta}$, and T_3 = rotation around z''−axis by α'. Thus, in matrix form

$$T_1 = \begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$T_2 = \begin{bmatrix} cos(90 - \hat{\theta}) & 0 & sin(90 - \hat{\theta}) \\ 0 & 1 & 0 \\ -sin(90 - \hat{\theta}) & 0 & cos(90 - \hat{\theta}) \end{bmatrix}$$

$$T_3 = \begin{bmatrix} cos\alpha' & sin\alpha' & 0 \\ -sin\alpha' & cos\alpha' & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where the signs in front of the sin terms is changed from + to − or from − to + to define the direction of rotation.

4. A point P_m on C_1 is unlikely to "hit" a (sample) pixel value at P'_m on the xy plane. Hence, the neighboring xy values of the point P'_m have to be found and the values of $F(x, y), F_x(x, y)$ and $F_y(x, y)$ interpolated from these. A suitable interpolation formula

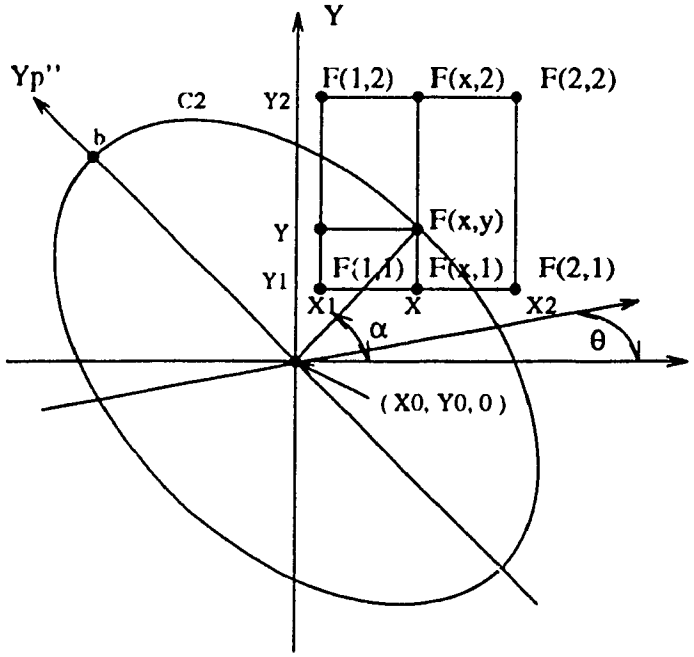


Figure 5.4: Interpolation on the xy-plane.

should be chosen. A simple 4-point formula is given below, see Figure 5.4, where $F(\cdot,\cdot)$ is any function.

$$F(x,1) = \frac{(x-x_1)[F(2,1)-F(1,1)]}{(x_2-x_1)} + F(1,1) \tag{5.11}$$

$$F(x,2) = \frac{(x-x_2)[F(2,2)-F(1,2)]}{(x_2-x_1)} + F(1,2) \tag{5.12}$$

$$F(x,y) = \frac{(y-y_1)[F(x,2)-F(x,1)]}{(y_2-y_1)} + F(x,1) \tag{5.13}$$

5. Thus, compute the values of $F(x,y), F_x(x,y)$ and $F_y(x,y)$ along the periphery of C_1 for n steps, say $n = 8$ or 12 or more and tabulate $T_f(k), T_{fx}(k), T_{fy}(k), k = 1,2,\cdots,n$. Of course, an even value n should be chosen in order to find the diametrically opposite values for k, i.e., k and $k_1 = k + \frac{n}{2}$ (module n), see Figure 5.3. Apply some smoothing if necessary. Compute the corresponding direction cosines for the surface normal $\mathbf{V_k}$ around C_1.

6. The normal vectors $\mathbf{V_k}, k = 1,2,\cdots,n$ around the periphery of C_1 are unlikely to be in the plane P_1 defined by the normal vector $\mathbf{N}$ and the tangent vector $\mathbf{t}$, see Figure 5.3, i.e., $\mathbf{V_k}$ has a component $\mathbf{V}_{pk}$ (or $\mathbf{V}_{nk}$ and $\mathbf{V_{tk}}$) in plane P_1, and a component $\mathbf{V}_{ck}$

along C_1. The vector on the diametrically opposite point on C_1 is $\mathbf{V}_{k1}$, where $k_1 = k + \frac{n}{2}$. The angle between the vectors $\mathbf{V}_{pk}$ and $\mathbf{V}_{pk1}$ is denoted by β'_k, which is normalized by dividing with the distance between the x, y, z values at k and k_1 to yield β_k

$$\beta_k = \frac{cos^{-1}(v_x w_x + v_y w_y + v_z w_z)}{d_k + d_{k1}} \tag{5.14}$$

where (v_x, v_y, v_z) and (w_x, w_y, w_z) are the direction cosines of $\mathbf{V}_{pk}$ and $\mathbf{V}_{pk1}$, respectively, and d_k and d_{k1} are the distances between the x, y, z values at k and k_1 and P. β_k is the normal curvature κ_n.

7. The principal curvatures κ_1 and κ_2 are

$$\kappa_1 = \max_{k=1}^{n/2}(\beta_k) \tag{5.15}$$

$$\kappa_2 = \min_{k=1}^{n/2}(\beta_k) \tag{5.16}$$

Let k_m and k_n denote the locations of the maximum and minimum, respectively, then κ_1 and κ_2 can be located more accurately by interpolation , for example, a second degree polynomial fitted at $\beta(k_m - 1), \beta(k_m)$ and $\beta(k_m + 1)$ for the maximum, and at $\beta(k_n - 1), \beta(k_n)$ and $\beta(k_n + 1)$ for the minimum.

8. In differential geometry, the sign of κ_n (or β_k) has no geometrical significance since it depends on the orientation of the surface normal $\mathbf{N}$ with respect to the surface S, or the choice of the direction for $\mathbf{U_1}$ and $\mathbf{U_2}$. In the case of range images, however, the surfaces are only "seen" from one side in any one image. It is important to know whether a surface is convex (say κ_n is positive) or concave (say, κ_n is negative). The signs of κ_1 and κ_2 are decided by the sign of $d_1 - d_2$ (see Figure 5.5).

9. An error measure is given by

$$e(x, y) = \sum_{k=1}^{n} \frac{\mathbf{V}_{ck}}{|\mathbf{V}|} \tag{5.17}$$

5.3 Matching Strategy

In the previous sections, it has been shown that principal curvatures are good candidates for feature matching because they are invariant under rigid motion and different parametrization, they contain substantial information about local surface shape and have

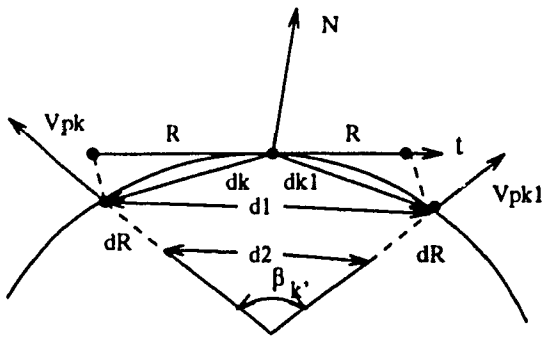


Figure 5.5: Determining the sign of the curvatures.

clear geometrical meaning. Therefore, they are chosen for the estimation of a 3D displacement field.

Let $\mathbf{u} = [\ u_1 \quad u_2 \quad u_3\ ]^T$ be the displacement vector of point $P : (x, y, z)$ at time t_1, this point is moved to point $P_1 : (x + u_1, y + u_2, z + u_3)$ at time t_2. If $t_2 - t_1 = 1$, then $\mathbf{u} = [\ u_1 \quad u_2 \quad u_3\ ]^T$ is approximately equal to the velocity vector of point P. Theoretically, the principal curvatures in the neighborhood of P and the principal curvatures in the neighborhood of point P_1 should have the same distributions. In practice, they are not exactly equal but similar because of noise in discrete images. Therefore, a correlation matching procedure can be used to find the 3D displacement field. The algorithm could be as simple as follows: choose a $m \times m$ neighborhood of point P, search around the corresponding P in the second image to find the best matching position where the error surface reaches its extreme. The error surface is defined as follows:

$$\begin{aligned} E(u_1, u_2) \quad = \quad & \sum_{x_i, y_i \in \mathcal{N}} \{\lambda_1[\kappa_1(x_i, y_i, t_1) - \kappa_1(x_i + u_1, y_i + u_2, t_2)]^2 \\ & + \lambda_2[\kappa_2(x_i, y_i, t_1) - \kappa_2(x_i + u_1, y_i + u_2, t_2)]^2\} \end{aligned} \tag{5.18}$$

where $u_1, u_2 = -l, \cdots, 0, \cdots, l$, $2l + 1$ is the size of the search area, and the maximum displacement should be smaller than l, $\mathcal{N}$ is the neighborhood of P, λ_1 and λ_2 are constants reflecting the relative importance of κ_1 and κ_2. The location (u_1, u_2) where $E(u_1, u_2)$ reachs the minimum value is considered as the estimate of the displacement vector of point P. u_3 can be simply obtained by

$$u_3 = F(x + u_1, y + u_2, t_2) - F(x, y, t_1)$$

where $F(x + u_1 + y + u_2, t_2)$ is the range image at time t_2.

Obviously, there exist many problems in this simple version of the algorithm. First, it is assumed that the error surface for each point will have an unique peak , which is not always true for points on a flat area or on edges (or on a cylinder). In the first case, the error surface will be flat with small fluctuation. In the second case, the error surface will be of a ridge shape. Simply locating extreme points in the error surface will cause problems. Even in the case of an unique peak, if the peak is not sharp enough, it is hard to locate. Second, inter-pixel displacement cannot be estimated.

Instead of using equation (5.18) for error surface, the following formula is used to calculate the correlation surface

$$\begin{aligned} C(u_1,u_2) &= \sum_{x_i,y_i \in \mathcal{N}} \{ \lambda_1 \frac{\sqrt{|\kappa_1(x_i,y_i,t_1)\kappa_1(x_i+u_1,y_i+u_2,t_2)|}}{|\kappa_1(x_i,y_i,t_1)-\kappa_1(x_i+u_1,y_i+u_2,t_2)|+\sigma} \\ &+ \lambda_2 \frac{\sqrt{|\kappa_2(x_i,y_i,t_1)\kappa_2(x_i+u_1,y_i+u_2,t_2)|}}{|\kappa_2(x_i,y_i,t_1)-\kappa_2(x_i+u_1,y_i+u_2,t_2)|+\sigma} \} \end{aligned} \tag{5.19}$$

where σ is a constant to prevent $C(u_1,u_2)$ becoming infinite. In this formula, the correlation will be small if the principal curvatures are small even in its corresponding point because this corresponds to a flat area. If principal curvatures are different, then the correlation value is small, if the principal curvatures are large and similar, then the correlation value is large. Compared with equation (5.18), this formula reduces the effect of small principal curvatures because the accuracy of estimated small curvatures is lower than that of larger curvatures.

The error measure in computing κ_1 and κ_2 is used to prevent incorrectly estimated principal curvatures affecting the correlation surface. If the error measures for κ_1 and κ_2 are large, then these κ_1 and κ_2 will not be used for correlation.

The correlation surface is interpolated by a second degree polynomial function around 3×3 neighborhood of the peak. The (u_1,u_2) where this second degree polynomial function reaches maximum gives the estimate of displacement vector at point P. Thus, inter-pixel displacement can be estimated.

The correlation surfaces with nonunique peaks are further processed by the procedure discussed in the next section.

5.4 Coping with Nonunique Peaks

As discussed before, the correlation surface may be flat, ridge shape, or have multiple peaks. Figure 5.7 shows the correlation surfaces computed from a synthetic image (see Figure 5.6), where the template size is 5×5 and the search size is 9×9. The image size is 64×64. The object in the image is a polyhedron which consists of flat surfaces and line edges between them. The object is translated toward upper right corners by two pixels. For each pixel, a 9×9 correlation surface is obtained. A zero value boundary is inserted between two correlation surfaces for a better view. The darker the pixel is, the smaller the correlation value is. It is clear that the correlation surfaces for corner points have single bright points, which means that the correlation surfaces have unique peaks (Figure 5.8(a) shows the correlation surface at corner pixel $(15, 15)$), therefore, the displacement vectors of those pixels can be determined uniquely. For pixels near line edges, their correlation surfaces have a line of bright spots, i.e., the correlation surfaces have ridge shapes (Figure 5.8(b) shows the correlation surface at edge pixel $(32, 17)$), the displacement of a point on a ridge is ambiguous, its component in the perpendicular direction of the ridge is unambiguous. This phenomenon is the well-known aperture problem. For a pixel on a plane, its correlation surface is flat, there are no enough information to locally determine the displacement vector of the pixel. Since there always exist edges in most of real scenes, the algorithm has to be able to cope with these problems.

There are several ways to deal with the problem. Obviously, one can first locate corners, line edges, planes or cylinders in images, which have been studied by lots of people (see [64, 61] etc.). Once this information is obtained, appropriate methods can be used to deal with different cases.

The other way is to directly process correlation surfaces. Correlation surfaces can be classified into peak, flat, or ridge etc., then appropriate methods are utilized to deal with different cases.

The latter method has been chosen since some reliability measures can be obtained at the same time.

Two techniques to classify the correlation surfaces have been developed. They will

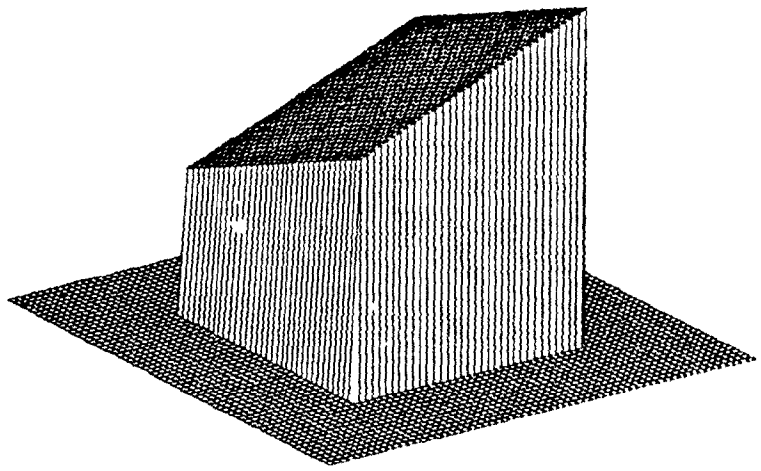


Figure 5.6: A synthetic polyhedron image.

be described in the following sections.

5.4.1 Algorithm 1

The principle of the first algorithm is as follows:

1. Threshold a correlation surface into a binary image, where '1' for peak area and '0' for non-peak area. Let C_p denote the correlation surface for pixel P and $C_p(i,j), i,j = -l,\cdots,0,\cdots,l$ be correlation values, then calculate average correlation value $\bar{C}_p$

$$\bar{C}_p = \sum_{i=-l}^{l}\sum_{j=-l}^{l}\frac{C_p(i,j)}{l^2}$$

Convert the correlation surface into a binary image C_{pb} according to

$$C_{pb}(i,j) = \begin{cases} 1 & if\ \ (C_p(i,j) - \bar{C}_p) > thavg \\ 0 & otherwise \end{cases}$$

where $thavg$ is a predefined threshold, it is set to $thavg = 1.5$ for all experiments.

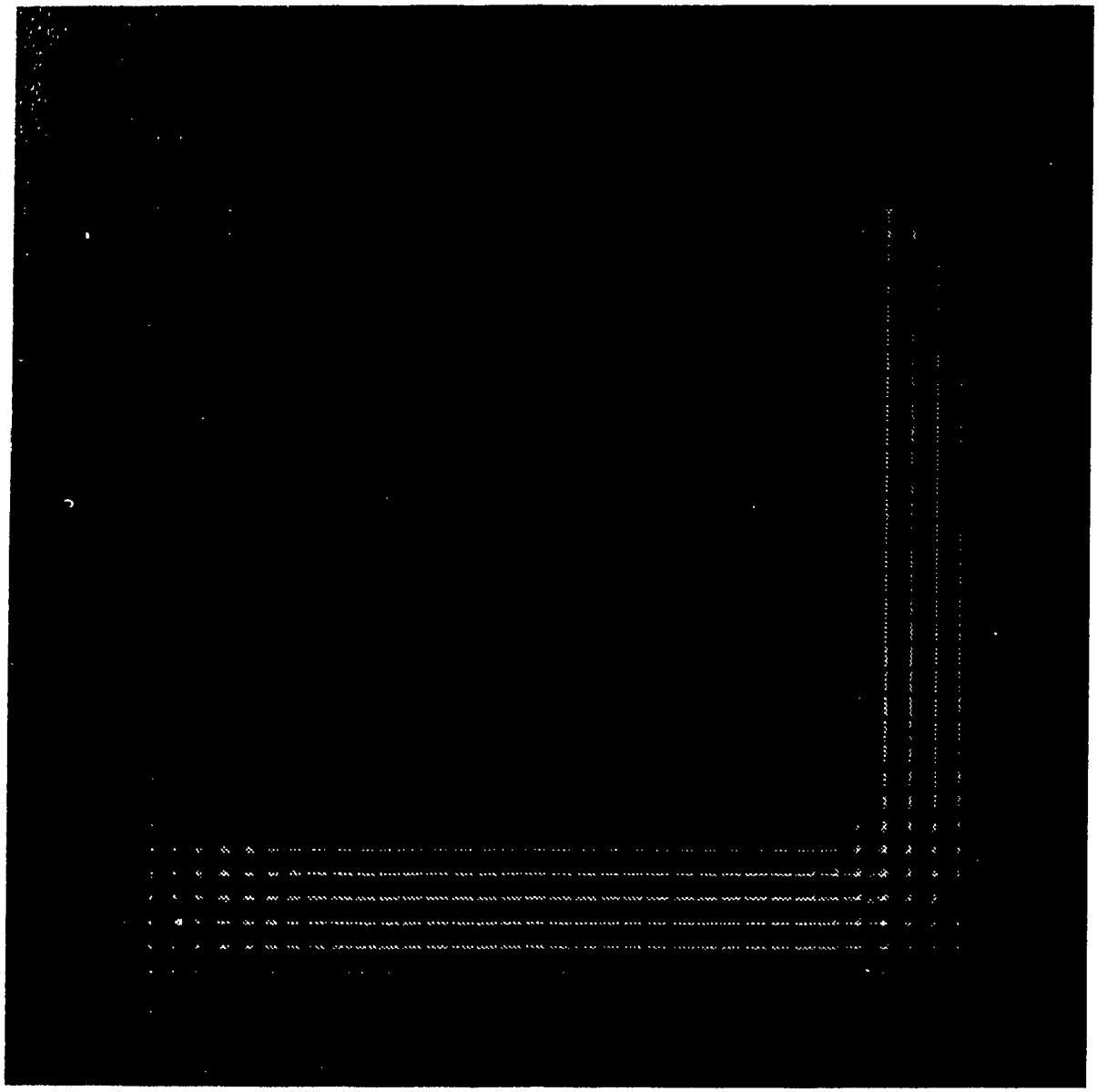
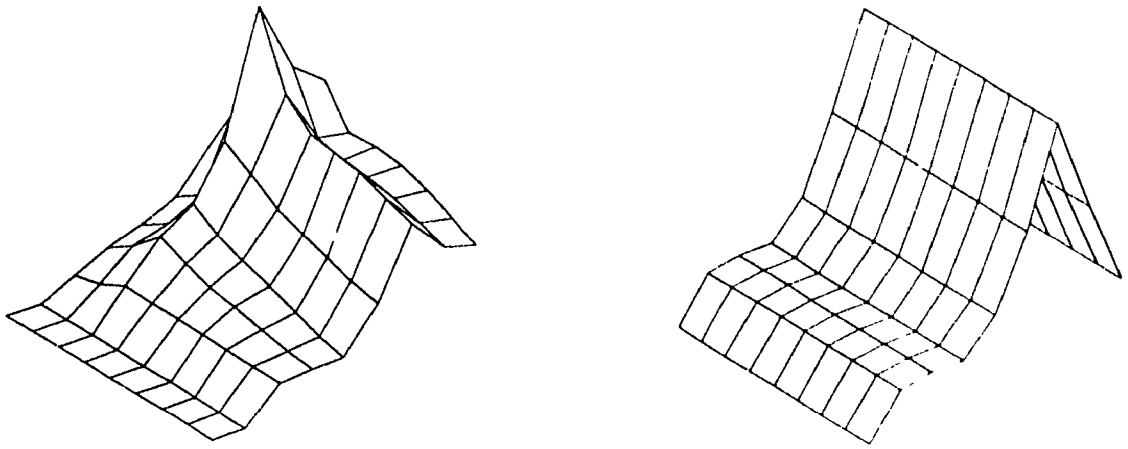


Figure 5.7: Correlation surfaces for the polyhedron image.



(a) (b)

Figure 5.8: (a) The correlation surface at corner pixel (15, 15). (b) The correlation surface at line edge pixel (32, 17).

2. Thin the resultant binary image C_{pb} to yield C_{pbt} using the algorithm described in [96] (see Figure 5.9).

3. Label each connected '1' segment in the image C_{pbt} and store the resultant image into C_{pbtl}

4. Classify the correlation surface into one of the five classes: *flat peak*, *multi-peaks*, *line peak*, *uni-peak*, *multi-line peak*, by checking the image C_{pbtl} according to the following rules

   (a) If there is no connected '1' segment (the maximum label number is zero in image C_{pbtl}), then no peak exists in the correlation surface, return *flat peak*

   (b) If only one connected '1' segment exists (the maximum label number is one in image C_{pbtl}) and if this segment forms a line, then return *line peak*, otherwise return *uni-peak*

   (c) If there exists more than one connected '1' segment (the maximum label number is larger than 1 in image C_{pbtl}), then more than one peak exist in the correlation surface, if these '1' segments form a line, then return *multi-line peak*, otherwise

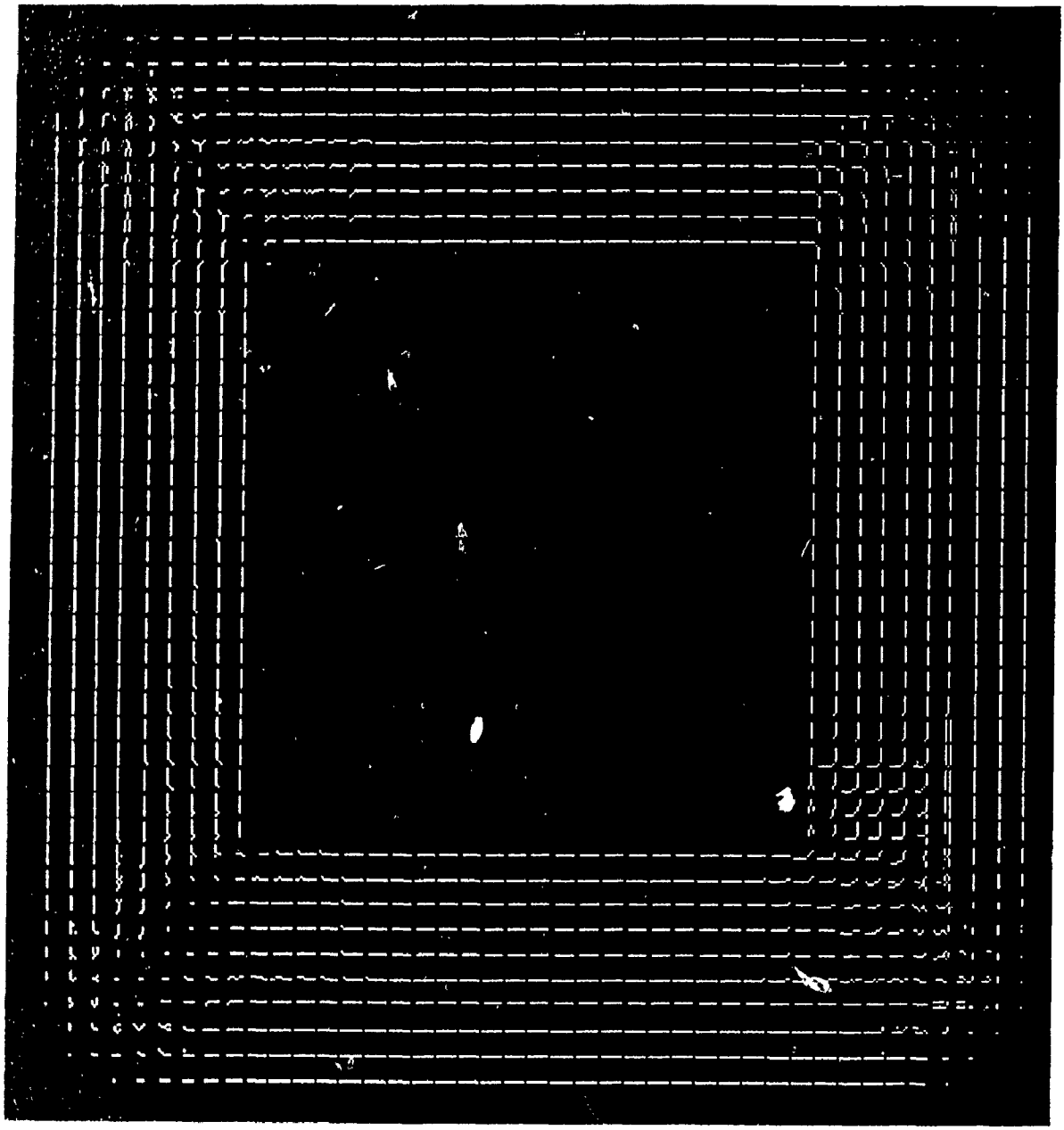


Figure 5.9: Thinning binary image of correlation surfaces for the polyhedron.

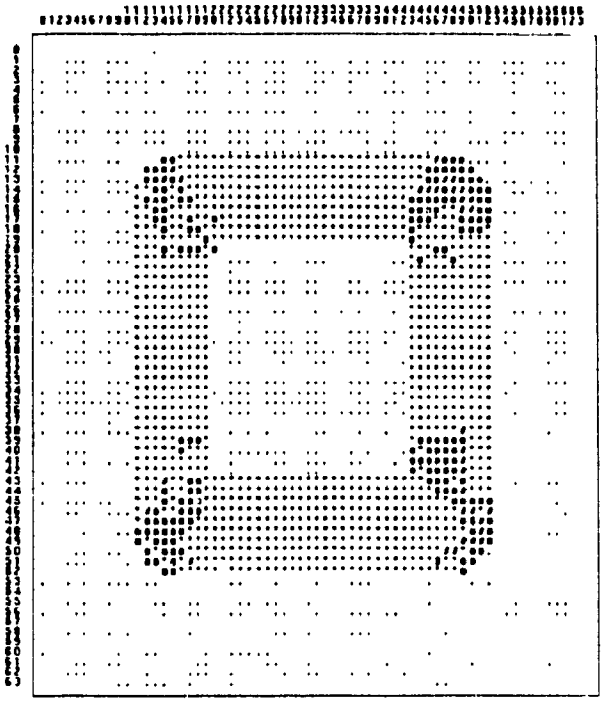


Figure 5.10: Classification of correlation surfaces for the polyhedron using the first algorithm.

return *multi-peak*

Figure 5.10 shows the classified image, where '.' denotes *flat peak*, '*' for *line peak*, 'o' for *uni-peak* and '#' for *multi-line peak*.

5.4.2 Algorithm 2

The second technique classifies a correlation surface into one of the following categories: *flat peak*, where the correlation surface is flat, *multi-peak* where more than two peaks exist in the correlation surface, *line peak*, where the correlation surface has a ridge shape, *uni-peak*, where there exists one sharp peak in the correlation surface, *boundary peak*, where peak is near the boundary, *unknown peak*, where the correlation surface is irregular.

1. Obtain peak areas by thresholding: let C_p denote the correlation surface for pixel P and $C_p(i,j), i,j = -l, \cdots, 0, \cdots l$ be correlation values, calculate the average correlation value $\bar{C}_p$

$$\bar{C}_p = \sum_{i=-l}^{l} \sum_{j=-l}^{l} \frac{C_p(i,j)}{l^2}$$

where l is the search size. If $(C_p(i,j) - \bar{C}_p) > thavg$, then (i,j) belongs to peak area, where $thavg$ is a predefined threshold, it is set to $thavg = 1.5$ for all experiments.

2. If the number of points in the peak areas is zero, then return *flat peak*, otherwise continue.

3. Locate peaks in the peak area, if more than two peaks exist, then return *multi-peak*, otherwise continue.

4. If the peak is on the boundary of correlation surface, return *boundary peak*, otherwise continue.

5. If the peak is ambiguous, for example, the correlation surface has a ridge shape, then calculate the center of gravity of the peak area as the peak point. Let points in the peak area be $(x_1, y_1, F(x_1, y_1)), (x_2, y_2, F(x_2, y_2)), \cdots, (x_n, y_n, F(x_n, y_n))$, the center of gravity (x_g, y_g) is defined as

$$x_g = \frac{\sum_{i=1}^{n} x_i F(x_i, y_i)}{\sum_{i=1}^{n} F(x_i, y_i)}$$
$$y_g = \frac{\sum_{i=1}^{n} y_i F(x_i, y_i)}{\sum_{i=1}^{n} F(x_i, y_i)}$$

6. Take a 3×3 neighborhood around the peak point, fit a second degree surface into the local surface in this neighborhood, calculate principal curvatures κ_1, κ_2 and their directions from the fitted surface

7. Classify the local surface based on principal curvatures. If $k1 > thk1$ and $k2 > thk2$, then return *uni-peak*, If $k1 \leq thk1$ and $k2 \leq thk2$, then return *flat peak*, If $k1 > thk1$ and $k2 \leq thk2$, then return *line peak*, otherwise, return *unknown peak*. $thk1$ and $thk2$ are two thresholds. They are set to $thk1 = 0.05, thk2 = 0.05$ for all experiments. Figure 5.11 shows the classification results for the polyhedra of Figure 5.6.

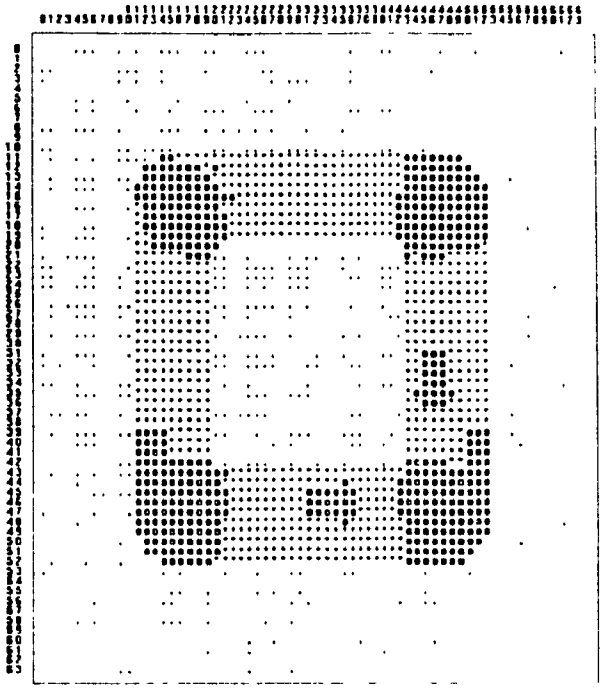


Figure 5.11: Classification of correlation surfaces for the polyhedron using the second algorithm.

Once the classification of correlation surfaces is obtained using one of these two algorithms , the displacement field is estimated by the following algorithm:

1. Classify a correlation surface by one of the above techniques.

2. If the peak is *uni-peak*, then the peak point of the correlation surface gives an unique estimate of the velocity. In order to obtain inter-pixel movement, a surface is fitted around the peak and velocity can be obtained from the peak of the fitted surface.

3. If the peak is *line peak*, then only the normal velocity can be estimated, which is equal to the distance between the center of the correlation surface and the peak line, the line passing through the center of gravity of the peak area in the principal direction k2 (normal direction). The following steps are taken to obtain this normal velocity.

   (a) Find the equation of the peak line. Let θ be the angle of the normal direction with respect to x (see Figure 5.12), then the normal vector is $(\cos\theta, \sin\theta)$, the

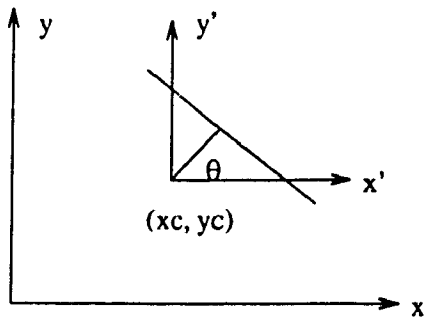


Figure 5.12: Peak line.

vector $(x - x_g, y - y_g)$ is perpendicular to the normal vector, therefore, their dot productor is zero, the line equation can be obtained from

$$(x - x_q)\cos\theta + (y - y_g)\sin\theta = 0$$

$$x\cos\theta + (y - x_q)\sin\theta\cos\theta - y_g\sin\theta = 0$$

Line parameters are

$$a = \cos\theta$$

$$b = \sin\theta$$

$$c = -x_q cos\theta - y_q \sin\theta$$

(b) Calculate the distance d of the line to the origin

$$d = -c$$

$$v_\perp = d$$

where $v_\perp$ is the normal velocity.

(c) Assuming that the velocity along the line is zero, then the x, y velocities are

$$u_1 = v_\perp \cos\theta$$

$$u_2 = v_\perp \sin\theta$$

where $\theta = \tan^{-1}(b/a)$.

4. For all other peaks, no estimates of velocities can be obtained, set them to zeros.

Figures (5.15) and (5.16) show the estimated full velocity field and the normal velocity field based on the classification of the correlation surfaces using the first algorithm. Figures

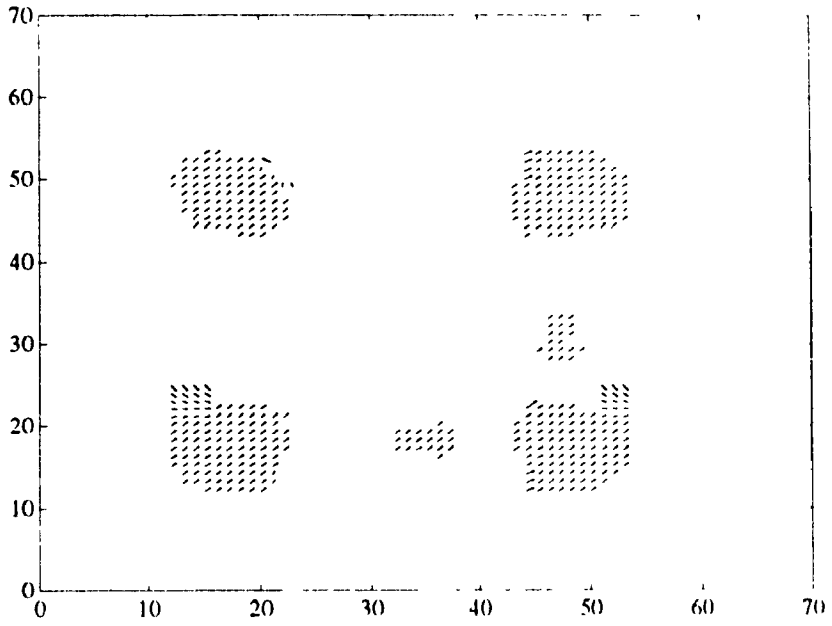


Figure 5.13: Full velocity field by the first algorithm.

(5.13) and (5.14) show the estimated full velocity field and the normal velocity field based on the classification of the correlation surfaces using the second algorithm. It is clear that velocities can be estimated uniquely at the corner pixels, and only normal velocities can be estimated at line edge points. However, the classification of the correlation surfaces using the first algorithm is very sensitive to the threshold "thavg". In addition, the principal curvatures at the peak point of a correlation surface provided by the second algorithm can be used as reliability measure of the estimated velocity. If both principal curvatures are large, then the peak is very sharp, which means that the estimated velocity is more reliable. Therefore, it is better to use the second algorithm to classify correlation surfaces. The remaining experiments in this chapter will use the second algorithm.

5.5 Experimental Results

More experiments have been done on both real and synthetic image sequences. In this section, part of the results will be presented. Most image sequences used are the same as

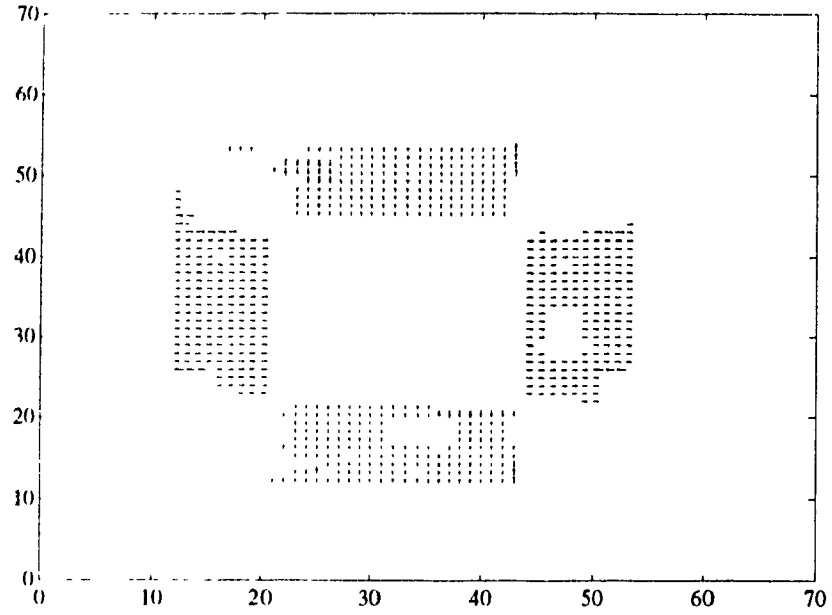


Figure 5.14: Normal velocity field by the first algorithm.

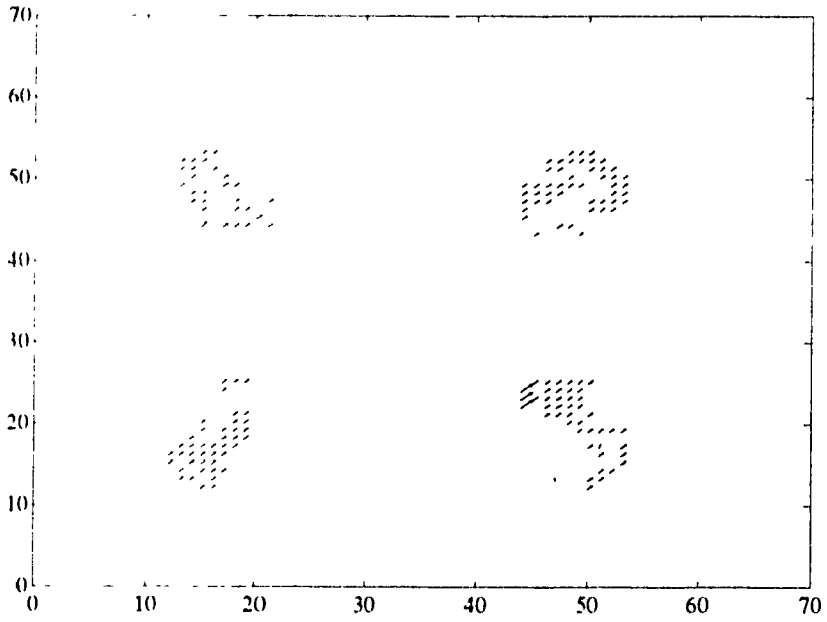


Figure 5.15: Full velocity field by the second algorithm.

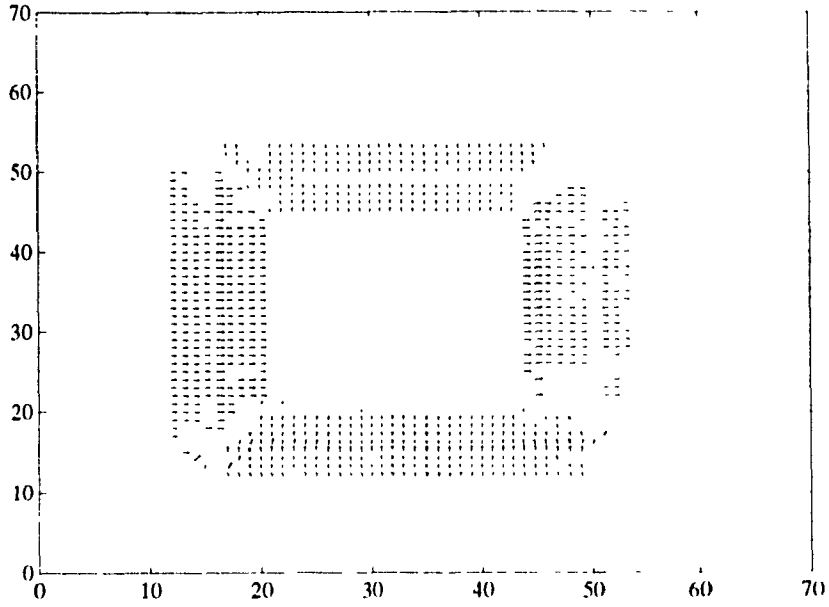


Figure 5.16: Normal velocity field by the second algorithm.

the ones used in the previous chapter.

5.5.1 A Piece of Paper

The description of the paper image sequence can be found in the previous chapter. 9×9 was used for the search size, and 5×5 for the template size. Figures 5.17 and 5.18 show the calculated principal curvatures. Since the object in the scene contains many plane facets, principal curvatures are very small in those areas, and the estimates of the velocities are unable to be obtained in these areas. Figure 5.19 shows the classification of peaks of correlation surfaces, where a white pixel for an *uni-peak*, a black for a *flat peak* and a gray pixel for a *line peak*. The estimated full velocity field is shown in Figure 5.20. It is well estimated for uni-peak pixels.

5.5.2 Grip

The description of Grip image sequence has been given in the previous chapter. 9×9 was used for the search size, and 5×5 for the template size. The estimated full velocity field

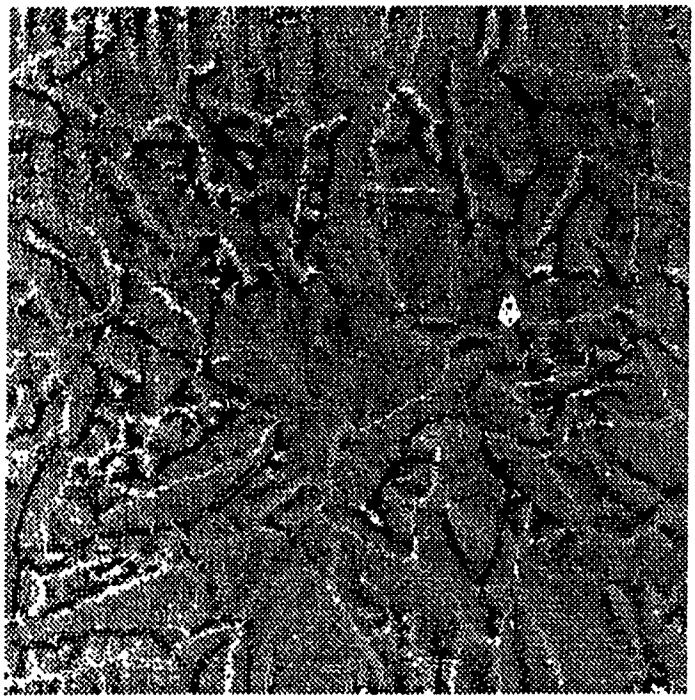


Figure 5.17: Maximum curvature image of paper sequence.

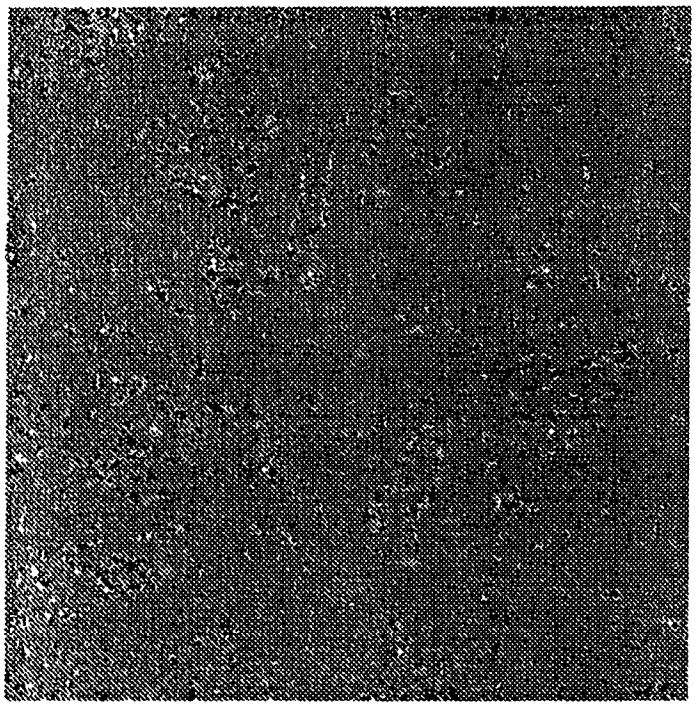


Figure 5.18: Minimum curvature image of paper sequence.

Figure 5.19: Classification of peaks of correlation surfaces of paper sequence.

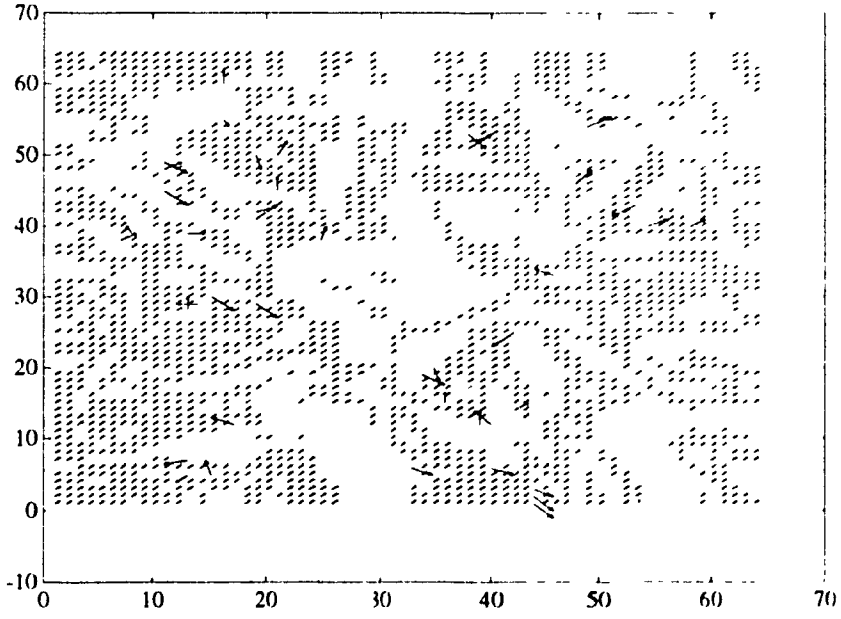


Figure 5.20: Full velocity field of paper sequence.

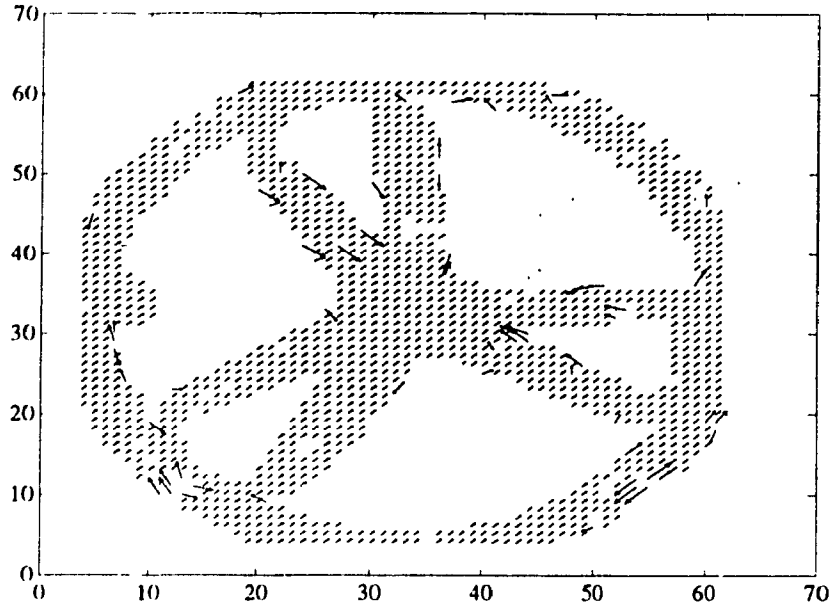


Figure 5.21: Estimated full velocity field of grip image sequence.

is shown in Figure 5.21. The object in the scene is made of planes, conic surfaces and cylindrical surfaces. Therefore, full velocities can not be estimated for these surfaces. Full velocities are well estimated around the boundaries of these surfaces.

5.5.3 Doll

The description of image sequence can be found in the previous chapter. 11×11 was used for the search size, and 5×5 for the template size. The estimated full velocity field is shown in figure 5.22.

5.6 Further Improvement

In this section, some possible improvements will be discussed for the algorithm discussed above.

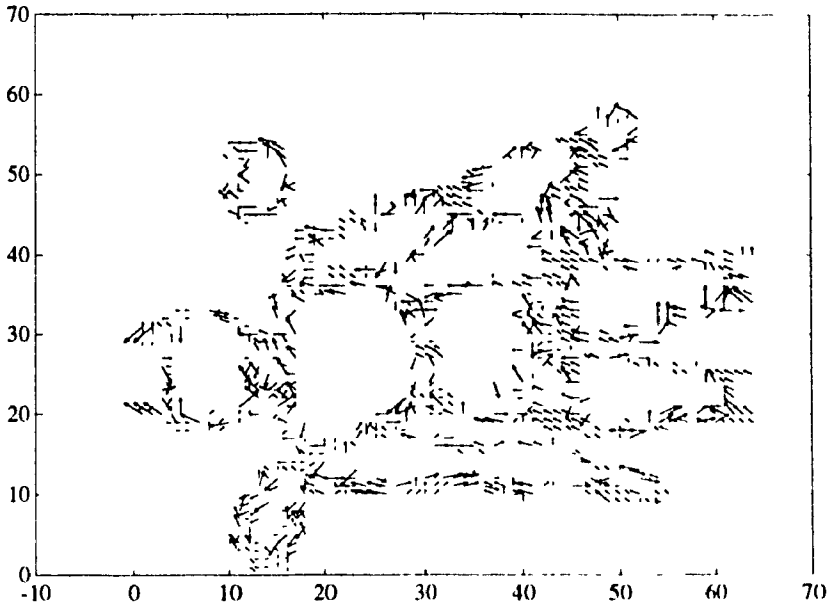


Figure 5.22: Estimated full velocity field of doll image sequence.

5.6.1 Using the Principal Directions

In the previous discussions, only principal curvatures are used to estimate displacement vector fields. It is obvious that a correlation surface of a point will be flat if the point is on a spherical surface, because principal curvatures are equal for a spherical surface. In this case, other features have to be used for matching. One possible way to avoid this problem is to use the angle differences between triplets $\mathbf{U_1}(x,y,t), \mathbf{U_2}(x,y,t), \mathbf{N}(x,y,t)$ and $\mathbf{U_1}(x+1,y+1,t), \mathbf{U_2}(x+1,y+1,t), \mathbf{N}(x+1,y+1,t)$, expressed by three Euler angles $\phi(x,y,t), \psi(x,y,t)$ and $\theta(x,y,t)$. These angle images are also intrinsic properties of a rigid object. These angles can be added to the formula of the correlation function.

Another way is to increase the size of a template such that the distributions of κ_1 an κ_2 in the template appear sufficiently changed. However, in the above matching process, it is assumed that the displacements of all points in the template are near constant. If the size of the template is too large, this assumption would be violated. Therefore, it is better to use the angle information when a scene contains spherical surfaces ($\kappa_1 = \kappa_2$).

5.6.2 Decoupling Rotation and Translation

In the case of rigid motion, the displacement vector results from a translational vector and a rotational vector. These two components in fact may be determined separately.

If point P at time t_1 is translated to point P_1 at time t_2, then the maximum curvature direction, the minimum curvature direction and the normal direction at point P should be the same as those directions at P_1, respectively. Any direction changes of $\mathbf{U_1}, \mathbf{U_2}$ and $\mathbf{N}$ are caused by a rigid rotation. This idea is used to decouple the rotation and translation components. Let (u_1, u_2, u_3) be the displacement vector between P and P_1, then the Euler angles between their corresponding orthogonal triplets at P and P_1 represent the rotation component from P to P_1, since $P_1 : (x + u_1, y + u_2)$ may not be on the sampling grid, an appropriate vector interpolation procedure is needed to find $\mathbf{U_1}, \mathbf{U_2}$ and $\mathbf{N}$ at point P_1 from the nearest neighbors of P_1. The interpola ion has to keep the orthogonal properties of $\mathbf{U_1}$, $\mathbf{U_2}$ and $\mathbf{N}$. From Euler angles, the rotation matrix $\mathbf{R_p}$ can be obtained. Translation vector is obtained by

$$\mathbf{t_P} = (\mathbf{I} - \mathbf{R_p})\mathbf{u} \tag{5.20}$$

where $\mathbf{t_p}$ is the translation vector, $\mathbf{u}$ is the displacement vector and $\mathbf{I}$ is a unit matrix. $\mathbf{t_p}$ and $\mathbf{R_p}$ are computed at point P. If the translation and rotation is expressed with respect to the origin, then

$$\begin{aligned} \mathbf{R_0} &= \mathbf{R_P} \\ \mathbf{t_0} &= (\mathbf{I} - \mathbf{R_p})\mathbf{P} + \mathbf{u} \end{aligned}$$

where $\mathbf{P} = [\ x \quad y \quad z\ ]^T$.

5.6.3 Increasing Speed

The disadvantage of the feature matching approach is that it is time consuming, especially when the motion is large such that a large search size has to be used. One of the many possible techniques to increasing the speed is to use multiple resolutions [7, 29, 30]. At lower resolution, motion is small, therefore, only a small area need to be searched. At higher resolution, matching is found around potential points provided by the lower level.

The algorithm discussed in the previous chapter is very fast, however, it will fail around the areas where surfaces are discontinuous. Therefore, it may be combined with the feature matching algorithm, used only in the areas which other algorithm cannot handle.

5.7 Conclusion

In this chapter, an approach has been proposed to estimate a 3D velocity field through token (feature) matching. Principal curvatures are chosen to be the features since they are invariant to both rigid motion and different parametrization of a surface. The 3D velocity at point P is obtained by locating a matching point whose distribution of principal curvatures in its neighborhood is similar to that of point P. If point P is on a line edge, then its full velocity cannot be determined because of the aperture problem, however, its vernier velocity can be determined. Two algorithms have been described to cope with this problem. The advantage of the feature matching approach is that it can estimate velocities of both corner points as well as points on smooth curved surfaces, and vernier velocities of line edge points. The method discussed in the previous chapter is unable to deal with corner points and line edge points. For objects like the polyhedra, it is very important to estimate the motion on corners and line edges. The disadvantage of the feature matching method is that it is very time consuming compared with the approach in the previous chapter. To increase processing speed, it is possible to combine them so that one technique can be used for line edges and corners, and the other technique for smooth surfaces.

Chapter 6

General Motion

In the previous chapters, the estimation of motion of a single rigid object and low level processing for more complicated motions have been discussed. In this chapter, a framework will be proposed which allows us to apply previous knowledge to analyze more general and higher level motion.

6.1 Representation of the High Level Process

Any arbitrary relative movement between objects in a scene and an observer can be expressed by a motion field where a 3D instantaneous velocity vector has been assigned to each point in the image at each time instance.

Let $\mathbf{u}(x,y,t) = [\ u_1(x,y,t)\quad u_2(x,y,t)\quad u_3(x,y,t)\ ]^T$ denote a motion field, that is, a function of both space and time. The motion field describes the local properties of motion, which is the goal pursued by a low level process but does not directly provide the relation of movements between neighboring points.

In the high level process, one has to interpret the motion field in order to provide more compact information about motion. If, for example, objects in a scene are rigid, then six rigid motion parameters for each object uniquely describe motion of the objects. Usually, the output of the high level process depends on the requirements of a specific application, or the nature of the objects, and therefore, there is no general representation for the high level process in this sense. In this section, the representation of the high level

process will be discussed assuming that there is no prior knowledge about the applications or the objects. A question can be posed: what is the appropriate representation?

Most work done so far use the model of uniform rigid body motion or special kinds of nonrigid motions [69, 24, 22, 104]. Subbarao [98] provided a model for general non-rigid motion of a small surface patch. In his model, the motion field is expressed locally as a polynomial function of position and time. This is a good model when no prior knowledge about the objects or motion is available. In this paper, the following representation is adopted: an image is segmented into several connected regions in which points tend to move coherently, and for each region, motion is represented by a polynomial function. This resembles the "common fate" law in Gestalist theory [120].

Let the segmentation at time t_0 be $S = (S^1, S^2, \cdots, S^m)$, where S^i is the i^{th} region in which all points undergo similar motion and m is the number of regions. Then the motion field in the i^{th} region can be expressed as

$$\mathbf{u}^i(x, y, t) = \mathbf{a}_0^i + \mathbf{a}_1^i x + \mathbf{a}_2^i y + \mathbf{a}_3^i z + \mathbf{a}_4^i t + \cdots \tag{6.1}$$

where $\mathbf{a}_j^i = [\ a_{1j}^i \quad a_{2j}^i \quad a_{3j}^i\ ]^T$. The $\mathbf{a}_j^i$'s are called *region motion parameters* or simply *region parameters*. Based on this model, the task for motion analysis is to find S^i and $\mathbf{a}_j^i$, $i = 1, \cdots, m$, given the input $z = F(x, y, t_0), \cdots, z = F(x, y, t_n)$, where n is the number of images used in motion analysis. The range of j depends on the degree of the polynomial which reflects the nature of motion.

In the case of the uniform motion of rigid bodies, the motion field in the i^{th} region is

$$\mathbf{u}^i(x, y, t) = \mathbf{v}^i + \omega^i \times \mathbf{r} \tag{6.2}$$

where $\mathbf{v}^i = [\ v_1^i \quad v_2^i \quad v_3^i\ ]^T$ is translational velocity, $\omega^i = [\ \omega_1^i \quad \omega_2^i \quad \omega_3^i\ ]^T$ is rotational velocity and $\mathbf{r} = [\ x \quad y \quad z\ ]^T$, $a_{01}^i = a_{02}^i = a_{03}^i = v_1^i$, $a_{11}^i = a_{22}^i = a_{33}^i$, $a_{23}^i = -a_{32}^i \quad \omega_1^i$, $_3 = a_{31}^i = \omega_2^i$, $a_{12}^i = a_{21}^i = \omega_3^i$. The task for motion analysis is to segment a scene into regions, each of which is undergoing a rigid motion, and estimate six rigid motion parameters for each region.

For the uniform motion of deformable objects, the m tion field in the i^{th} region can

be expressed by

$$\mathbf{u}^i(x, y, t) = \mathbf{a}_0^i + \mathbf{a}_1^i x + \mathbf{a}_2^i y + \mathbf{a}_3^i z \tag{6.3}$$

where the first order partial derivatives form a velocity tensor. Rigid translation, rotation and deformation parameters, such as stretching and shear, can be uniquely determined from $\mathbf{a}_j^i$ [98]. In the next sections, an approach will be presented which analyzes motion based on the above model.

There are several ways to achieve the goal. One of the possible ways is the following: estimate the motion field using any one of the methods discussed in the previous chapters, then extract region parameters locally from estimated motion field. After that, cluster these local results to form segmentation, and finally compute the region parameters. In the next sections, it will be shown that the first two steps may be combined to directly extract point parameters from each local area, then continue the process from these results.

6.2 Overview of the Approach

Figure 6.1 shows a block diagram of the proposed approach. It consists of six main modules.

1. Preprocessing

   Before any processing is performed on the image, it is necessary to filter out noise since the second derivatives of a surface need to be calculated, and they are sensitive to noise. A median filter may be a good choice since it does not corrupt boundaries.

2. Local measurement of point motion parameters

   Assuming that an arbitrary surface can be approximated by a second degree polynomial function in a very small but finite neighborhood, motion parameters are estimated in that neighborhood and the estimated values are assigned to the center point of the neighborhood, and are called *point motion parameters* (or *point parameters*). All point parameters form the point motion parameter field. At the same time, each point is assigned a weight reflecting the reliability of point motion parameters.

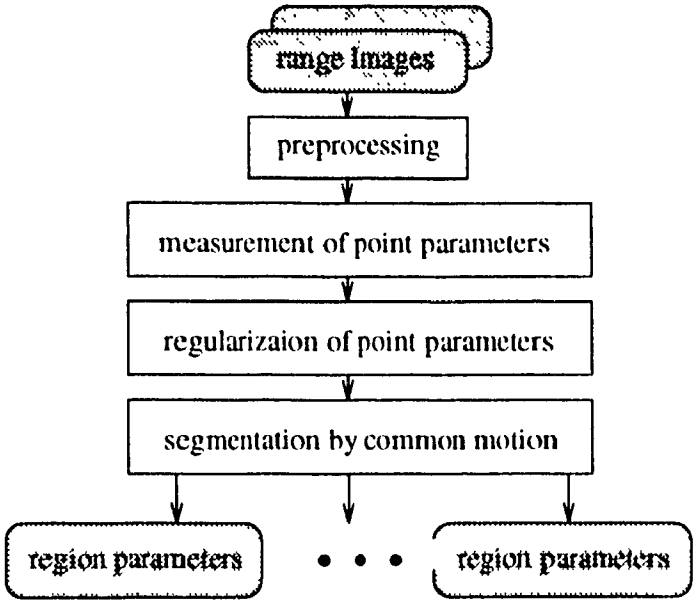


Figure 6.1: Block diagram of the approach.

3. Regularization of point motion parameters by motion coherence theory

   The estimated point motion parameter field is usually noisy and not very accurate. There are some points whose parameters cannot be estimated, or only some of the parameters can be estimated. The motion coherence theory [119, 120] is used to regularize (smooth) this parameter field.

4. Segmentation based on common motion

   If points belong to the same moving object, then their parameters should be similar. This idea is used to group points together to segment the image into a set of connected regions.

5. Estimation of region motion parameters

   Motion parameters of a region could simply take the average of point parameters in the same region. It is not a recommended method because the local measurements of point parameters are relatively noisy. Instead, a direct method is used to recover region motion parameters globally.

In the following sections, a detailed description of each module is presented.

6.3 Local Measurement of Point Motion Parameters

A method will be presented to estimate point parameters starting from a very simple but important case of locally pure translation or dominant translation, and then it is extended to general motion.

6.3.1 Uniform Locally Pure Translation or Dominant Translation

Let S_n be a small but finite neighborhood of a point P : [x_0 y_0 z_0]. If the surface in S_n undergoes only translation or translation with a relatively small rotation with respect to translation, then the former case is called locally pure translation and the latter locally dominant translation. In these two cases, 3D instantaneous velocities are either the same or nearly the same in S_n so that they can be treated as equal. This 3D velocity vector is the point parameter to be estimated. Any of the methods discussed in the last two chapters can be used in this case.

6.3.2 Uniform Locally Rigid Motion

In the previous section, it is assumed that the velocities are constant in a small neighborhood. In fact, the method discussed in chapter 4 can be easily extended to any locally rigid motion. Consider a local surface S_0 around the point $P : \mathbf{r_0} = [\ x_0 \quad y_0 \quad z_0\ ]^T$ at time t_0. Let XYZ be a local coordinate system with the point $\mathbf{r_0}$ as the origin and the X, Y and Z axes parallel to the x, y and z axes (see Figure 4.7). In this case, point parameters of P are expressed by six rigid motion parameters $\mathbf{v}$ and ω. 3D instantaneous velocity $\mathbf{u}$ of point P is related to these rigid parameters by

$$\begin{aligned} u_1 &= \omega_2(Z + z_0) - \omega_3(Y + y_0) + v_1 \\ u_2 &= \omega_3(X + x_0) - \omega_1(Z + z_0) + v_2 \\ u_3 &= \omega_1(Y + y_0) - \omega_2(X + x_0) + v_3 \end{aligned} \tag{6.4}$$

Assuming that the local surface S_0 can be approximated by the second order polynomial surface, then, since S_0 is undergoing a rotation, $Z = F(X, Y, t)$ can not be approximated by a second order polynomial function, instead it can be approximated by the following

function (see Appendix E) in the local coordinate system

$$\begin{aligned} Z &= F_x X + F_y Y + F_t t + F_{xy} XY + F_{xt} Xt + F_{yt} Yt + 1/2 F_{xx} X^2 + 1/2 F_{yy} Y^2 \\ &\quad + 1/2 F_{tt} t^2 + F_{xyt} XYt + 1/2 F_{xxt} X^2 t + 1/2 F_{yyt} Y^2 t \end{aligned} \tag{6.5}$$

Substituting this value of Z into equation (6.5), then substituting the resultant equation into the 3D velocity constraint equation (3.2) produces, using symbolic computation program (Maple), we get

$$a_0 + a_1 X + a_2 Y + a_3 XY + a_4 X^2 + a_5 Y^2 + a_6 X^2 Y + a_7 Y^2 X + a_8 X^3 + a_9 Y^3 = 0 \tag{6.6}$$

where

$$\begin{aligned} a_0 &= (-F_y z_0 - y_0)\omega_x + (F_x z_0 + x_0)\omega_y + (F_y x_0 - F_x y_0)\omega_z + F_x v_1 + F_y v_2 - v_3 + F_t \\ a_1 &= (-F_{xy} z_0 - F_y F_x)\omega_x + (1 + F_x^2 + F_{xx} z_0)\omega_y + (F_y + F_{xy} x_0 - F_{xx} y_0)\omega_z \\ &\quad + F_{xx} v_1 + F_{xy} v_2 + F_{xt} \\ a_2 &= (-1 - F_y^2 - f_{yy} z_0)\omega_x + (F_{xy} z_0 + F_y F_x)\omega_y + (-F_{xy} y_0 + F_{yy} x_0 - F_x)\omega_z \\ &\quad + F_{xy} v_1 + F_{yy} v_2 + F_{yt} \\ a_3 &= (-2F_y F_{xy} - F_{yy} F_x)\omega_x + (2F_{xy} F_x + F_y F_{xx})\omega_y + (F_{yy} - F_{xx})\omega_z + F_{xyt} \\ a_4 &= (-1/2 F_y F_{xx} - F_{xy} F_x)\omega_x + 3/2 F_x F_{xx}\omega_y + F_{xy}\omega_z + 1/2 F_{xxt} \\ a_5 &= -3/2 F_y F_{yy}\omega_x + (1/2 F_{yy} F_x + F_y F_{xy})\omega_y - F_{xy}\omega_z + 1/2 F_{yyt} \\ a_6 &= (-F_{xy}^2 - 1/2 F_{yy} F_{xx})\omega_x + 3/2 F_{xy} F_{xx}\omega_y \\ a_7 &= -3/2 F_{xy} F_{yy}\omega_x + (F_{xy}^2 + 1/2 F_{yy} F_{xx})\omega_y \\ a_8 &= -1/2 F_{xy} F_{xx}\omega_x + 1/2 F_{xx}^2\omega_y \\ a_9 &= -1/2 F_{yy}^2\omega_x + 1/2 F_{xy} F_{yy}\omega_y \end{aligned}$$

Equation (6.6) is valid for all neighboring points of the interested point P, as long as the number of points of this neighborhood is larger than 10, then $a_i = 0, i = 0, \cdots, 9$. This is a set of 10 linear equations with six unknowns for which it is easy to find the least squares estimates of six rigid motion parameters for the point (x_0, y_0, z_0). In the same way, motion parameters can be locally estimated for all other points. It is an interesting problem to see if there are some points for which motion cannot be locally recovered.

6.3.3 Arbitrary Motion: Nonrigid and Nonuniform

So far the problem how to measure point parameter field of a locally rigid body has been discussed. Actually, the method can be generalized to any arbitrary time-varying 3D scenes, which include nonrigid and nonuniform motions [1]. For arbitrary motion, 3D velocities of a point can be expressed by

$$\begin{aligned} u_1 &= a_{10} + a_{11}(X + x_0) + a_{12}(Y + y_0) + a_{13}(Z + z_0) + a_{14}(t + t_0) + O_2(X, Y, Z, t) \\ u_2 &= a_{20} + a_{21}(X + x_0) + a_{22}(Y + y_0) + a_{23}(Z + z_0) + a_{24}(t + t_0) + O_2(X, Y, Z, t) \\ u_3 &= a_{30} + a_{31}(X + x_0) + a_{32}(Y + y_0) + a_{33}(Z + z_0) + a_{34}(t + t_0) + O_2(X, Y, Z, t) \end{aligned}$$

where $O_2(X, Y, Z, t)$ denotes the second and higher order terms of X, Y, Z and t. As discussed before, the existence of coefficient a_{ij} depends on the nature of the motion. For example, for uniform motion of a deformable surface, the first four terms of each velocity component exist. For nonuniform motion of a deformable surface, the coefficients of the term $(t - t_0)$ are included. In these two cases, instead of considering two frames, multiple frames have to be considered. After a very similar computation procedure as before, the following equation is obtained,

$$b_0 + b_1 x + b_2 y + b_3 t + \cdots + b_{17} x^3 + b_{18} y^3 + b_{19} t^3 = 0 \tag{6.7}$$

where $b_i, i = 0, \cdots, 19$ are linear functions of a_{ij}.

$$\begin{aligned} b_0 &= F_x a_{10} + F_x x_0 a_{11} + F_x y_0 a_{12} + F_x z_0 a_{13} + F_y a_{20} + F_y x_0 a_{21} \\ &\quad + F_y y_0 a_{22} + F_y z_0 a_{23} - a_{30} - x_0 a_{31} - y_0 a_{32} - z_0 a_{33} + F_t \\ b_1 &= F_{xx} a_{10} + (F_{xx} x_0 + F_x) a_{11} + F_{xx} y_0 a_{12} + (F_{xx} z_0 + F_x^2) a_{13} + F_{xy} a_{20} \\ &\quad + (F_y + F_{xy} x_0) a_{21} + F_{xy} y_0 a_{22} + (F_{xy} z_0 + F_y F_x) a_{23} - a_{31} - F_x a_{33} + F_{xt} \\ b_2 &= F_{xy} a_{10} + F_{xy} x_0 a_{11} + (F_{xy} y_0 + F_x) a_{12} + (F_{xy} z_0 + F_y F_x) a_{13} + F_{yy} a_{20} \\ &\quad + F_{yy} y_0 a_{21} + (F_{yy} y_0 + F_y) a_{22} + (F_{yy} z_0 + F_y^2) a_{23} - a_{32} - F_y a_{33} + F_{yt} \\ b_3 &= 3/2 F_{xt} F_{xx} a_{13} + (1/2 F_{yt} F_{xx} + F_{xy} F_{xt}) a_{23} + F_{tt} \\ b_4 &= F_{xy} a_{11} + F_{xx} a_{12} + (2 F_{xy} F_x + F_y F_{xx}) a_{13} \\ &\quad + F_{yy} a_{21} + F_{xy} a_{22} + (2 F_{xy} F_y + F_x F_{yy}) a_{23} - F_{xy} a_{33} \end{aligned}$$

[1] Nonuniform motion means that motion parameters change with time

$$
\begin{aligned}
b_5 &= F_{xt}a_{11} + (2F_{xt} + F_{xx}F_t)a_{13} + F_{yt}a_{23} \\
&\quad +(F_{xt}F_y + F_{xy}F_t + F_xF_{yt})a_{23} - F_{yt}a_{33} \\
b_6 &= F_{xt}a_{12} + (F_{xt}F_y + F_{xy}F_t + F_xF_{yt})a_{13} + F_{yt}a_{23} \\
&\quad +(F_{yy}F_t + 2F_yF_{yt})a_{23} - F_{yt}a_{33} \\
b_7 &= F_{xt}a_{10} + F_{xt}x_0a_{11} + F_{xt}y_0a_{12} + (F_xF_t + F_{xt}z_0)a_{13} + F_{yt}a_{20} \\
&\quad +F_{yt}x_0a_{21} + F_{yt}y_0a_{22} + (F_{yt}z_0 + F_yF_t)a_{23} - F_ta_{33} \\
b_8 &= F_{xy}a_{12} + (1/2F_xF_{yy} + F_{xy}F_y)a_{13} + F_{yy}a_{22} + 3/2F_yF_{yy}a_{23} - 1/2F_{yy}a_{33} \\
b_9 &= (F_{xt}F_t + 1/2F_xF_{tt})a_{13} + (1/2F_yF_{tt} + F_{yt}F_t)a_{23} - 1/2F_{tt}a_{33} \\
b_{10} &= (F_{yt}F_{xx} + 2F_{xy}F_{xt})a_{13} + (F_{xt}F_{yy} + 2F_{xy}F_{yt})a_{23} \\
b_{11} &= (F_{xy}^2 + 1/2F_{xx}F_{yy})a_{13} + 3/2F_{xy}F_{yy}a_{23} \\
b_{12} &= 3/2F_{xy}F_{xx}a_{13} + (F_{xy}^2 + 1/2F_{xx}F_{yy})a_{23} \\
b_{13} &= 3/2F_{xt}F_{xx}a_{13} + (1/2F_{yt}F_{xx} + F_{xy}F_{xt})a_{23} \\
b_{14} &= (1/2F_{xt}F_{yy} + F_{xy}F_{yt})a_{13} + 3/2F_{yt}F_{yy}a_{23} \\
b_{15} &= (1/2F_{xx}F_{tt} + F_{xt}^2)a_{13} + (F_{yt}F_{xt} + 1/2F_{xy}F_{tt})a_{23} \\
b_{16} &= (F_{yt}F_{xt} + 1/2F_{xy}F_{tt})a_{13} + (F_{yt}^2 + 1/2F_{yy}F_{tt})a_{23} \\
b_{17} &= 1/2F_{xx}^2a_{13} + 1/2F_{xy}F_{xx}a_{23} \\
b_{18} &= 1/2F_{xy}F_{yy}a_{13} + 1/2F_{yy}a_{23} \\
b_{19} &= 1/2F_{xt}F_{tt}a_{13} + 1/2F_{yt}F_{tt}a_{23}
\end{aligned}
$$

For the same reason as before, b_i must equal zero, and so a set of 20 linear equations with 12 unknowns are formed for which it is easy to find the least square solution. Here F_{tt} has to be calculated, thus at least three frames are needed. For nonuniform motion of a deformable surface, the coefficients of the term $(t - t_0)$ are included. After a similar procedure, a set of 20 linear equations with 15 unknown will be obtained. For any other more complicated nonrigid motion which includes coefficients of the second or higher order terms, the second order approximation of z is not enough to estimate the point motion parameters, but these parameters can be estimated in a very small region by a similar procedure.

6.4 Regularization of Point Motion Parameters

So far, point motion parameters are being estimated at all points except where they cannot be locally estimated. For example, for local translation, point parameters cannot be estimated at points with zero Gaussian curvature. However, our segmentation by common motion required a dense point motion parameter field. Also, the local measurements of point motion parameters are not accurate near boundaries. Smoothing and interpolating procedure (regularization) is needed for these points. Recently, Yuille and Grzywacz [119, 120] proposed a motion coherence theory which performs smoothing and interpolation. Their results are extended to our problem by introducing a weight for each measured value. Since x and y components of the velocity field in motion coherence theory do not interact, each component can be treated separately. Let a_i be one component of point motion parameters measured at $\mathbf{r_i} = [\ x_i \quad y_i\ ]^T$ The motion coherence theory suggests the construction of a smoothed component $a(x, y)$ such that the following function is minimized

$$E(\mathbf{r}, a_i) = \sum_i w_i(\mathbf{r} - a_i)^2 + \lambda \int \sum_{m=0}^{\infty} c_m (D^m a)^2 \tag{6.8}$$

where $D^{2m}a = \Delta^{2m}a$, $D^{2m+1}a = \Delta(\Delta^{2m}a)$, $\lambda \geq 0$ and $c_m \geq 0$ are constant. The index i runs over all points with estimated motion parameters and w_i is a normalized weight which represents reliability of the measured value. The weight is given during the measurement of point parameters. The solution of the above equation has the form

$$a(\mathbf{r}) = \sum_i \frac{\beta_i}{2\pi\sigma^2} \left(\frac{-||\mathbf{r} - \mathbf{r}_i||^2}{2\sigma} \right) \tag{6.9}$$

where the β_i are solutions of

$$\sum_j (\lambda\delta_{i,j} + w_i G_{i,j})\beta_j = w_i a_i$$

where

$$G_{i,j} = \frac{1}{2\pi\sigma} exp\left(\frac{-||\mathbf{r_i} - \mathbf{r_j}||^2}{2\sigma} \right)$$

The error function E contains an approximation-error term and a smoothness term. Parameter λ determines the importance of smoothness while parameter σ determines the

desired range of influence of this smoothing and interpolating process. This smoothing process can also be extended to include partially estimated parameters, for example, the normal velocity.

6.5 Segmentation by Motion Coherence

An image is segmented into regions which have similar point motion parameters. The method is an extension of Kasvand's 2D clustering method [65]. First, construct a histogram $H(\mathbf{a})$, where $\mathbf{a}$ is point parameter vector, smooth $H(\mathbf{a})$, and carry out dynamic thresholding to obtain cluster centers. Give "identity" to the isolated centers by region labeling and "spread" the labels out since dynamic thresholding only preserves the peaks and ridges in $H(\mathbf{a})$. Map the cluster labels back into the image space. This produces the labeled image where each label corresponds to a region moving coherently. Label relaxation may be needed if measured point motion parameters are too noisy. Connected cluster labels in image space are given new unique labels. Finally, a segmentation $S = (S^1, S^2, \cdots, S^m)$ is obtained in the image space.

6.6 Estimation of Region Motion Parameters

If a scene consists of locally rigid objects, then the estimation of region motion parameters is the same as the estimation of a single rigid object motion, since each region can be treated as a single rigid object. The method discussed in chapter 3 can be easily applied here. Furthermore, the method can be extended to general motion, the only difference is the number of motion parameters.

6.7 Experiments

To test the feasibility of the method, a preliminary experiment on the synthetic images shown in Figure 6.2 has been performed. The image size is 128× 128. Two overlapping objects are moving in the opposite directions. The left object is moving toward upper right and the right object is moving toward bottom left. The motion parameters for the left object are $\mathbf{T} = [\ 1 \quad 1 \quad 1\ ]^T/pixel, \theta = 1^0, \mathbf{n} = [\ 0 \quad 0 \quad 1\ ]^T$ and $\mathbf{T} = [\ -1 \quad -1 \quad -1\ ]^T, \theta = -1^0, \mathbf{n} = [\ 0 \quad 0 \quad 1\ ]^T$ for the right object, where θ is rotation angle, $\mathbf{T}$ is translation vector and $\mathbf{n}$ is unit vector of rotation axis. The rotation axes are passing through point $[\ 48 \quad 48 \quad 0\ ]^T$ for the left object and point $[\ 75 \quad 75 \quad 0\ ]^T$ for the right object. The noise of range data taken by the range finder is proportional to the distance between the sensor and the point being measured. Therefore, uniform noise whose range is 10 percent of the considered z to each point were added to the images. Figure 6.3 shows the estimated velocities using the method discussed in Chapter 3 without reliability checking and smoothing. Absolute velocities larger than the threshold (5) are set to the threshold for display and images are reduced to 64 × 64. In order to show the effect of noise, no preprocessing is used. Better results can be expected if a filter is applied first. Figure 6.4 shows the smoothed velocity field. Weights are shown in Figure 6.5. Segmentation is given in Figure 6.6, where each symbol represents a region in which points are moving coherently.

6.8 Conclusion

In this chapter, a framework has been described for recovering general motion from range image sequences. It consists of four main steps: local estimation of point motion parameters, regularization of point parameters, segmentation based on common motion, and region motion parameter estimation. Motion for both rigid and nonrigid objects are obtained from solving a set of linear equations. The framework is proposed to illustrate the potential extensions of the work done so far and for further future works. For each step in the framework, there exist numerous ways to reach the goal. Here, only some possibilities are pointed out. Further study needs to be done to find the best algorithm.

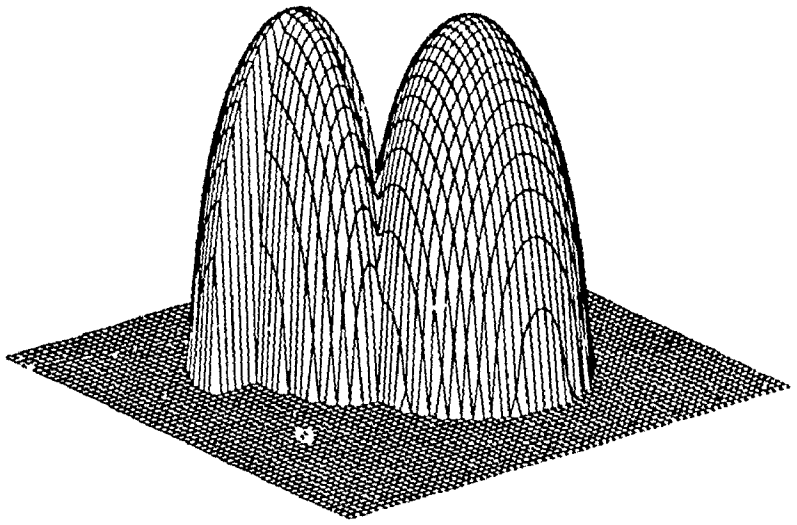


Figure 6.2: Synthetic image: half sphere - half ellipse.

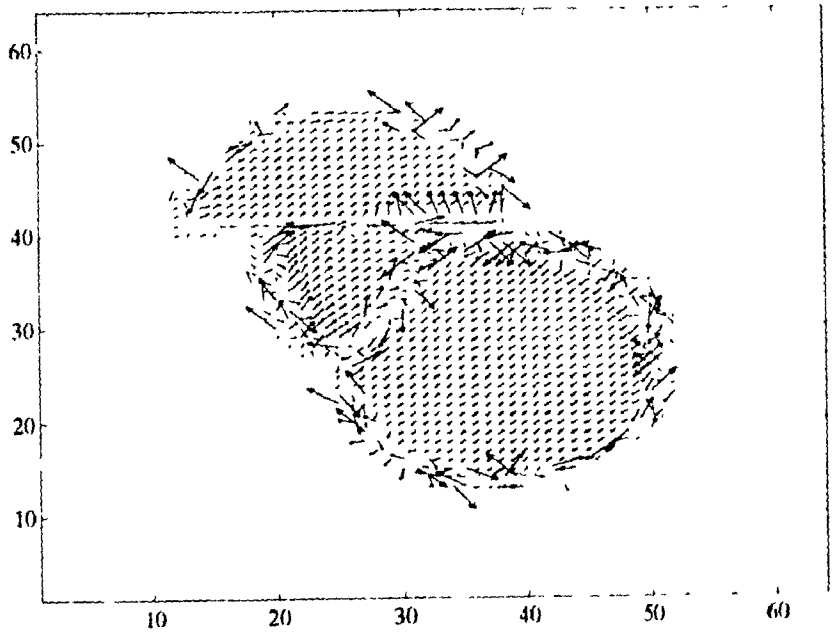


Figure 6.3: Estimated point parameters of synthetic image: half sphere - half ellipse.

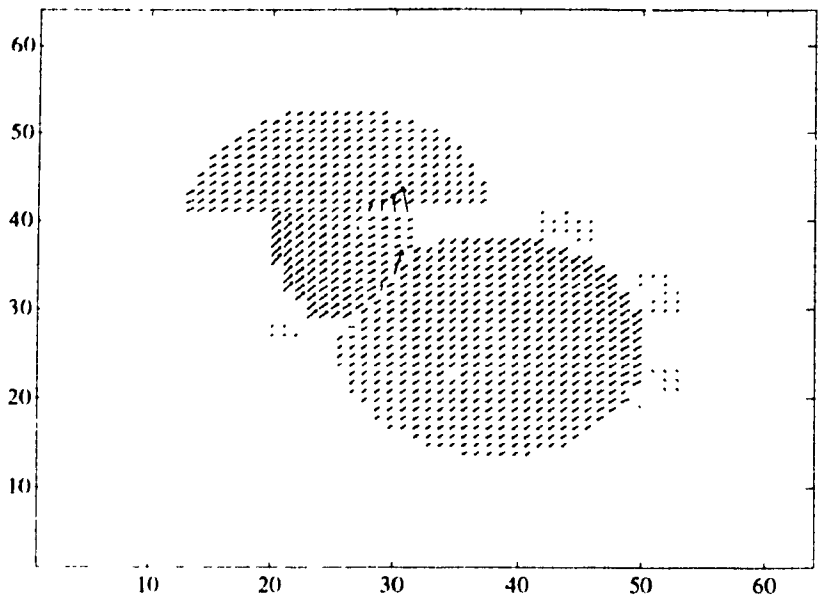


Figure 6.4: Smoothed point parameters of synthetic image: half sphere - half ellipse.

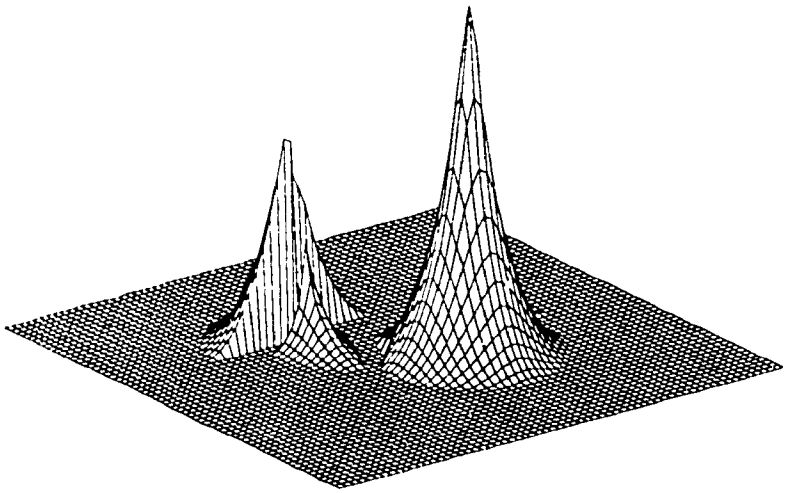


Figure 6.5: Weight image for smoothing.

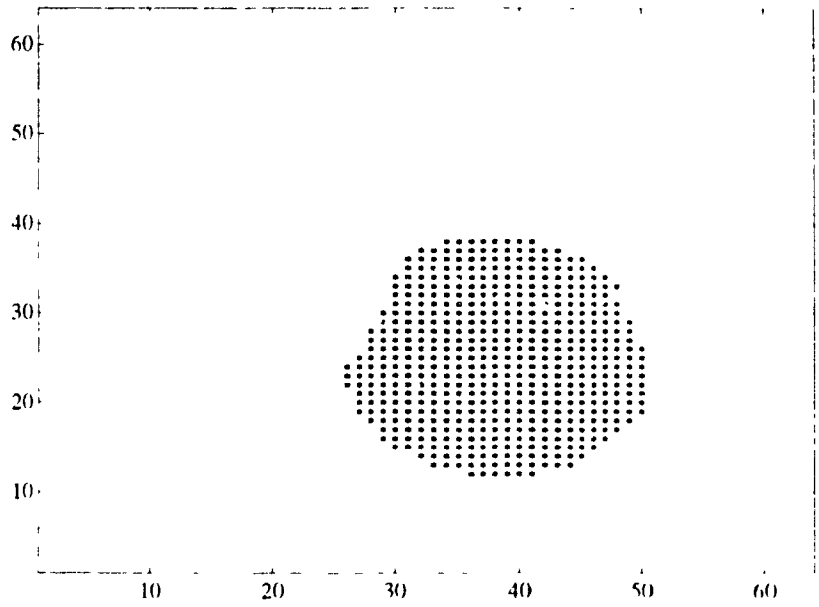


Figure 6.6: Segmented synthetic image using motion coherence: half sphere half ellipse

Chapter 7

Conclusion

Our final chapter summaries the work done so far, and provides a brief introduction to future research.

The work presented in this thesis can be divided into two separate parts. Chapter 2 describes the long term process, and chapter 3 to 6 discuss the short term process.

7.1 The Long Term Process

In the long term process, it is assumed that motion is recovered in two stages: the obtaining of correspondence of features and the estimation of motion parameter from matching features. Chapter 2 proposes an approach to establish the best match of point features between successive frames using a Hopfield neural network. The "best" is in the sense that the number of found matching pairs which satisfy the physical constraints should be as large as possible. Two physical constraints have been considered, that is, there exists no one-to-many matching and points are the features of rigid objects. These constraints are used to model the Hopfield neural network such that the stable state of the network corresponds to the matching solution which is being sought. Once the matching is established, a δ-bound matching test is used to discard mismatched features during parameter estimation. Therefore, the estimated motion is less sensitive to noisy features and occlusions.

The proposed approach has been tested on noisy synthetic data. Very promising

results have been obtained. No tests on real data have been done since it involves developing algorithms to extract features, which is beyond the scope of this thesis. However, feature extraction may be bypassed using the method proposed in chapter 3 if all the 3D coordinates in both images are taken as point features, provided by the range scanner directly. In this case, the number of features is huge. It takes too long for a conventional matching algorithm to process them. However, it should not dramatically increase the processing time for a neural network which essentially performs parallel processing. This idea has not been tested due to the lack of sufficient facilities. It should be verified in the future.

The proposed method is not restricted to point features. It has been shown that the idea can be easily extended to line, plane and surface matching. The method can also be used to estimate motions of multiple rigid objects. In this case, nothing needs to be changed at the matching stage. The algorithm to estimate rigid motion parameters need to be modified. One possible way is to check all found matched features by the δ-bound test. The features which pass the test are considered to belong to one object, and the remaining matched features belong to other objects. The features which belong to the second object can be found by repeating the δ-bound test on these remaining matched features. The same procedure can be repeated until no more matched features can fit a rigid object. These ideas should be further explored, and experiments on real data should be done.

7.2 The Short Term Process

The methods for the long term process can certainly be used for the short term process. However, in the latter case, other techniques which do not require correspondence can also be used to solve the problem. Furthermore, for a nonrigid object, its motion may not be recovered simply by matching features. It is necessary to study short term motion in its own right.

In this thesis, the case of a single rigid moving object is first studied in chapter 3. It is the simplest type of motion, yet it is often encountered. For example, when a mobile

robot moves in a static environment, the scene can be considered as a single rigid object.

A simple, straightforward, and very powerful direct method has been developed for this purpose. The method utilizes a similar idea which is used for optical flow to derive a local 3D velocity constraint equation. Based on this constraint, six rigid motion parameters of a single rigid moving object can be directly obtained by solving a set of linear equations using least squares techniques. It has been found that a weighted least squares method provides the best performance after comparing the performance of several least squares methods against noise.

Sufficient and necessary conditions for the uniqueness of the motion perception have been discussed. The uniqueness of the motion perception depends on the structure of the 3D object in a scene. It has been proven that if and only if the surface in a scene is a plane, or a cylindrical surface, or a sphere, or a surface of revolution with its axis of revolution passing through the origin, then their rigid motion cannot be uniquely determined.

The behavior of the algorithm with noisy data has been analyzed in two steps. First the error sources have been discussed, then sensitivity of the algorithm to these errors has been analyzed. It has been shown that the major error, gradient measurement error, consists of two components: random error from random noise in measured depth and systematic error introduced by sampling surfaces discretely in time and space. The random error is determined by the range scanner. The systematic error depends on the spatial resolution of the camera, time interval between two successive frames, the second of partial derivatives of the surface, and motion itself. The error will be reduced if the spatial resolution of the camera and the sampling rate in time are increased. The slower the motion and the surface change in space, the smaller the systematic error is.

The sensitivity of the estimated motion parameters to noise also depends on the condition number of the linear system. It has also been proven that if normal vectors of a surface tends to be orthogonal to each other, then the system will be least sensitive to noise if the motion is translation, and for small objects or objects far away from the observer, the estimate of rotation will be very sensitive to noise in data.

The algorithm is tested on a real "shut" image sequence, the estimation error for the rotation angle is less than 2%.

The analysis of the single rigid motion problem is only a small branch of motion understanding. There are many applications which require analysis of more complicated motion, such as nonrigid motion, multiple objects in motion, etc. In these cases, Typically, low level processing locally extracts motion information. In high level processing, the final description of motion is derived.

In this thesis, attention has been paid to the low level processing since the high level processing is usually application dependent, and this thesis aims to explore more general principles behind understanding motion. A 3D velocity field has been chosen to be the output of the low level stage.

Chapter 4 first reviews and compares several plausible techniques: least squares method, robust regression methods, clustering approaches, for the estimation of the local motion. The major drawback of these techniques is that they are only suitable to surfaces which can be locally approximated by planes. To overcome this problem, a new algorithm is proposed. This algorithm is based on the assumptions that a local surface can be approximated by a second degree polynomial function, and 3D velocities are constant in a very small neighborhood. The algorithm uniquely determines the 3D velocity of each point by using the first and second order spatial and temporal partial derivatives, except at parabolic points where Gaussian curvatures are equal to zero.

For each calculated velocity, a measure of the reliability is calculated according to the following criteria: the value of Gaussian curvature, the coefficient matrix of the linear system which determines the 3D velocity, and how well a local surface is fitted by a second degree polynomial function.

For those parabolic points, their velocities are interpolated from the reliable velocities of their neighbors. For range images, it is reasonable to assume that the discontinuities of a velocity field only occur where depth is discontinuous. Therefore, the velocity field can be smoothed within areas where there are no discontinuities. Based on this idea, the interpolation problem is solved by smoothing the estimated reliable velocity field within motion boundaries.

The advantage of this algorithm is that it is very fast and easy to implement in hardware or software. The problem with this algorithm is that only velocities of smooth

surfaces can be estimated. For edge points, their velocities have to be inferred from their neighbors. However, for an object like polyhedron, its motion is perceived by the motion of its edges and corners. The proposed algorithm fails in this case.

Chapter 5 describes another algorithm to estimate 3D velocity field through token (feature) matching. Principal curvatures features are chosen since they are invariant to both rigid motion and different parametrization of a surface. The basic idea of the algorithm is that the 3D velocity at a point P can be obtained by locating a matching point whose distribution of principal curvatures in its neighborhood is similar to that of point P. The advantage of the feature matching approach is that it can estimate velocities of both corner points as well as points on smooth curved surfaces, and vernier velocities of line edge points.

If point P is on a line edge, then its full velocity cannot be determined because of the aperture problem, however, its vernier velocity can be determined. Two techniques have been developed to cope with this problem.

The disadvantage of the feature matching method is that it is computationally intensive compared with the approach of the proceeding chapter. To increase processing speed, it is possible to combine them, for example, into one algorithm in which one technique is used for line edges and corners, and the other technique is used for smooth surfaces.

These two algorithms to estimate 3D velocity are tested on real and synthetic images. For most image sequences, they provides reasonable results.

This feature matching method can also determine the rotation of a local surface by detecting the rotation of the principal curvature directions and normal vector. Therefore, translation and rotation of a local surface can be estimated separately. This ideas should be further explored. Besides, multiple resolution images can be used to speed up the correlation. This should be also studied in the future.

Once a 3D velocity field is obtained, higher level description of motion can be derived. Some ideas toward this end are presented in chapter 6. However, only preliminary experiments on the synthetic images were performed to test the feasibility of the approach. In the future, work should be done to verify and improve suggested ideas.

Bibliography

[1] N. N. Abdelmalek. Linear l1 approximation for a discrete point and l1 solutions of overdetermined linear equations. *J. Assoc. Comput. Mach.*, 18:41–47, 1971.

[2] Gilad Adiv. Inherent ambiguities in recovering 3-D motion and structure from a noisy flow field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5):477–489, May 1989.

[3] J. K. Aggarwal, L. S. Davis, and W. N. Martin. Correspondence processes in dynamic scene analysis. *Proceedings of the IEEE*, 69(5):562–572, May 1981.

[4] J. K. Aggarwal and M. J. Magee. Determining motion parameters using intensity guided range sensing. *Pattern Recognition*, 19(2):169–180, February 1986.

[5] J. K. Aggarwal and N. Nandhakumar. On the computation of motion from sequences of images – a review. *Proceedings of the IEEE*, 76(8):917–935, August 1988.

[6] John (Yiannis) Aloimonos and Isidore Rigoutsos. Determining the 3-D motion of a rigid surface patch without correspondence, under perspective projection: I. planar surfaces. II. curved surfaces. In *Proceedings of American Association for Artificial Intelligence National Conference on Artificial Intelligence*, pages 681–688, Philadelphia, August 1986.

[7] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[8] D. F. Andrews. A robust method for multiple linear regression. *Technometrics*, 16:523–531, 1974.

[9] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698 700, September 1987.

[10] H. S. Baird. *Model-based image matching using location.* The MIT Press, 1985.

[11] S. T. Barnard and W. B. Thompson. Disparity analysis of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(4):333 340, July 1980.

[12] J. A. Beraldin, M. Rioux, F. Blais, L. Cournoyer, and J. Domey. Registered intensity and range imaging at 10 mega-samples per second. *Optical Engineering*, 31(1):88 94, January 1992.

[13] P. J. Besl. *Surfaces in Range Image Understanding.* Springer-Verlag, New York, 1988.

[14] Paul Besl and Ramesh Jain. Range image understanding. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 430–449, San Francisco, California, June 1985. IEEE Computer Society, IEEE Computer Society Press.

[15] Paul J. Besl and Ramesh C. Jain. Three-dimensional object recognition. *Computing surveys*, 17(1):75–145, March 1985.

[16] S. Blostein and T. Huang. Estimating 2-d motion from range data. In *Proceedings 1st Conference on Artifitial Intelligence Application*, pages 246 250, Denver, Colorado, December 1984.

[17] Robert C. Bolles and Martin A. Fischler. A RANSAC-based approach to model fitting and its application to finding cylinders in range data. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 637 643, UBC, Vacouver, B.C. Canada, August 24-28 1981.

[18] O. J. Braddick. A short-range process in apparent motion. *Vision Research*, 14:519 527, 1974.

[19] O. J. Braddick. Low-level and high-level processes in apparent motion. *Phil Trans Roy Soc Lond B*, 290:137–151, 1980.

[20] J. P. Brady, N. Naudhakumar, and J. K. Aggarwal. Recent progress in the recognition of objects from range data. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 85–92, Rome,Italy, November 1988. IEEE Computer Society, IEEE Computer Society Press.

[21] P. J. Burt, C. Yen, and X. Xu. Local correlation measures for motion analysis: A comparative study. In *IEEE Conference Pattern Recognition and Image Processing*, pages 269-274, 1982.

[22] Subhasis Chaudhuri and Shankar Chatterjee. Estimation of motion parameters for a deformable object from range data. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 291–295, San Diego, California, June 1989. IEEE Computer Society, IEEE Computer Society Press.

[23] Subhasis Chaudhuri and Shankar Chatterjee. Performance analysis of total least squares methods in three-dimensional motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(5), October 1991.

[24] Su-Shing Chen and Michael Penna. Shape and motion of nonrigid bodies. *Computer Vision, Graphics and Image Processing*, 36:175–207, 1986.

[25] Roland T. Chin and Charles R. Dyer. Model-based recognition in robot vision. *Computing surveys*, 18(1):67-108, March 1986.

[26] Manfredo P. do Carmo. *Differential geometry of curves and surfaces.* Prentice-Hall Inc., New Jersey, 1976.

[27] J. Domey, M. Rioux, and F. Blais. 3-D sensing for robot vision. In *Proceedings of Workshop on Range Image Understanding, Vision Interface'89*, pages 1–35, London, Ontario, June 1989.

[28] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis.* John Wile & Sons, New York, 1973.

[29] Wilfried Enkelmann. Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences. In *IEEE Workshop on Motion: Representation and Analysis*, pages 81–87, Kiawah Island Resort, May 1986. IEEE Computer Society, IEEE Computer Society Press.

[30] Wilfried Enkelmann. Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences. *Computer Vision, Graphics and Image Processing*, 43:150–177, 1988.

[31] J. Q. Fang and T. S. Huang. Some experiments on estimating the 3-d motion parameters of a rigid body from two consecutive image frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(5):545 554, September 1984.

[32] O. D. Faugeras. A 3-D recognition and positioning algorithm using geometrical matching between primitive surfaces. In *Proceedings of International Joint Conference Artificial Intelligence*, pages 996–1002, Karlshure. West Germany, Aug. 1983.

[33] O. D. Faugeras, F. Lustman, and G. Toscani. Motion and structure from point and line matches. In *1st international Conference Computer Vision*, pages 25 34, London, England, June 1987.

[34] C. L. Fennema and W. B. Thompson. Velocity determination in scenes containing several moving objects. *Computer Graphics and Image Processing*, 9:301 315, April 1979.

[35] M. A. Fischler and R. C. Bolles. Randon sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24:381–395, 1981.

[36] David J. Fleet and Allan D. Jepson. Hierarchical construction of orientation and velocity selective filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(3):315-325, March 1989.

[37] Patrick J. Flynn and Anil K. Jain. On reliable curvature estimation. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*,

pages 110 116, San Diego, California, June 1989. IEEE Computer Society, IEEE Computer Society Press.

[38] Abraham Goetz. *Introduction to differential geometry.* Addison Wesley Publishing Company, Don Milles, Ontario, 1968.

[39] G. H. Golub and C. F. Van Loan. An analysis of the total least square problem. *SIAM J. Numer. Anal.*, 17:883-893, 1980.

[40] G. H. Golub and C. F. Van Loan. *Matrix Computation.* The Johns Hopkins University Press, Baltimore, Maryland, 1989.

[41] W. E. L. Grimson. *From images to surfaces: a computational study of the human early visual system.* MIT Press, Cambridge, Mass., 1981.

[42] W. E. L. Grimson. Computational experiments with feature based stereo algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(1):17-34, 1985.

[43] U. L. Haass and T. A. Brubaker. Estimation of cloud motion from satellite pictures. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pages 422-425, Denver, Colorado, 1980.

[44] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel. *Robust Statistics: An Approach Based on Influence Functions.* Wiley, New York, 1986.

[45] David J. Heeger. Optical flow using spatiotemporal filters. *International Journal of Computer Vision*, 1(4):279-302, 1988.

[46] Joachim Heel. Direct dynamic motion vision. In *Proceddings 1990 IEEE International Conference on Robotics and Automation*, pages 1142-1147, Cincinnati, Ohio, May 1990. IEEE Robotics and Automation, IEEE Computer Society Press.

[47] E. C. Hildreth. *The Measurement of Visual Motion.* The MIT Press, Cambridge, Massachusetts, 1983.

[48] R. L. Hoffman and A. K. Jain. Segmentation and classification of range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):608 620, September 1987.

[49] J. J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceeding National Academic Society USA*, 81(5):3088-3092, May 1984.

[50] J. J. Hopfield and D. W. Tank. "neural" computation of decisions in optimization problems. *Biological Cybernetics*, 52:141-152, 1985.

[51] Berthold K. P. Horn. *Robot Vision*. The MIT Press, Cambridge, MA, 1986.

[52] Berthold K. P. Horn. Closed-form solution of absolute orientation using quaternions. *J. Opt. Soc. Am.*, 4:629-642, 1987.

[53] Berthold K. P. Horn and John G. Harris. Rigid body motion from range image sequences. *Computer Vision, Graphics and Image Processing: Image Understanding*, 53(1):1-13, January 1991.

[54] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185-203, 1981.

[55] P. J. Huber. *Robust Statistics*. Wiley, New York, 1981.

[56] D. J. Ittner and A. K. Jain. 3-d surface discrimination from local curvature measures. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 119-123. IEEE Computer Society, IEEE Computer Society Press, May 1985.

[57] L. A. Jaeckel. Estimating regression coefficients by minimizing the dispersion of residuals. *Ann. Math. Stat.*, 80:1449 - 1458, 1972. from Mee7:91.

[58] R. A. Jarvis. A laser time-of-flight range scanner for robotic vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(5):505 511, September 1983.

[59] R. A. Jarvis. A perspective on range finding techniques for computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):122–139, March 1983.

[60] Behzad Kamgar-Parsi, Behrooz Kamgar-Parsi, and Nathan S. Netanyahu. A non-parametric method for fitting a straight line to a noisy image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(9):998–1001, September 1989.

[61] T. Kanade. *Three-Dimensional Machine Vision.* Kluwer Academic Publishers, Boston, MA, 1987.

[62] Tony Kasvand. Surface curvatures in 3d range images. In *Proceedings of International Conference on Pattern Recognition*, pages 842–845, Paris, France,, October 1986. IEEE Computer Society, IEEE Computer Society Press.

[63] Tony Kasvand. The k1k2 space in range image analysis. In *Proceedings of International Conference on Pattern Recognition*, pages 923–926, Rome, Italy, November 1988. IEEE Computer Society, IEEE Computer Society Press.

[64] Tony Kasvand. A strategy for processing 3d range images. In *Tutorial for Vision Interface '88*, Edmondon, Alberta, June 1988.

[65] Tony Kasvand. A strategy for processing 3d range images. In *IAPR Workshop on CV - Special Hardware and Industrial Application*, Tokyo, October 1988.

[66] J. K. Kearney, W. B. Thompson, and D. L. Boley. Optical flow estimation: An error analysis of gradient-based methods with local optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(2):229–244, March 1987.

[67] N. Kehtarnavaz and S. Mohan. A framework for estimation of motion parameters from range images. *Computer Vision, Graphics and Image Processing*, 45:88–105, 1989.

[68] Y. C. Kim. Determining object motion in a sequence of stereo images. *IEEE Journal Robotics Automation*. 3(6):599–614, December 1987.

[69] J. J. Koenderink and A. J. Van Doorn. Depth and shape from differential perspective in the presence of bending deformations. *Journal of Optical Society American Association*, 3(2):242-249, 1986.

[70] E. Kreyszig. *Introduction to differential geometry and riemannian geometry.* University of Toronto Press, 1968.

[71] J. O. Limb and J. A. Murphy. Estimating the velocity of moving images in television signals. *Computer Graphics and Image Processing*, 4:311-327, 1975.

[72] Zse-Cherng Lin, Thomas S. Huang, and Steven D. Blostein. Motion estimation from 3-d point sets with and without correspondences. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 194-201, Miami Beach, FL, June 1986. IEEE Computer Society, IEEE Computer Society Press.

[73] Zse-Cherng Lin, Hua Lee, and Thomas S. Huang. Finding 3-d point correspondences in motion estimation. In *Proceedings of International Conference on Pattern Recognition*, pages 303–305, Paris, France, October 1986.

[74] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133-135, 1981.

[75] H. C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. *Proceedings of the Royal Society of London B*, 208:385-397, 1980.

[76] D. Marr and T. Poggio. A computational theory of human stereo vision. *Proceedings of the Royal Socity of London B*, 204:301–308, 1979.

[77] D. Marr and S. Ullman. Directional selectivity and its use in early visual processing. *Proceedings of the Royal Socity of London B*, 211:151-180, 1981.

[78] W. N. Martin and J. K. Aggarwal. *Motion Understanding: Robot and Human Vision.* Kluwer Academic Publishers, Boston, 1988.

[79] Peter Meer, Doron Mintz, Azriel Rosenfeld, and Dong Yoon Kim. Robust regression methods for computer vision: A review. *International Journal of Computer Vision*, 6(1):59–70, April 1991.

[80] Hans-Hellmut Nagel. Displacement vectors derived from second-order intensity variations in image sequences. *Computer Vision, Graphics and Image Processing*, 21:85–117, 1983.

[81] Hans-Hellmut Nagel. Image sequences—ten (octal) years—from phenomenology towards a theoretical foundation. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(3):459–483, 1988.

[82] Shahriar Negahdaripour. Ambiguities of a motion field. In *Proceedings of the IEEE First International Conference on Computer Vision*, pages 607–612, London, England, June 1987. IEEE Computer Society, IEEE Computer Society Press.

[83] Shahriar Negahdaripour. A direct method for locating the focus of expansion. *Computer Vision, Graphics and Image Processing*, 46:303–326, 1989.

[84] Shahriar Negahdaripour and Berthold K. P. Horn. Direct passive navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):168–176, January 1987.

[85] R. J. Popplestoe, C. M. Brown, A. P. Ambler, and G. F. Crawford. Forming models of plane-and-cylinder faceted bodies from light stripes. In *Proceedings of 4th Joint Conference on Artifitial Intelligence*, pages 664–668, 1975.

[86] S. Ranada and A. Rosenfeld. Pattern matching by relaxation. *Point Pattern Recognition*, 12(3):269-275, 1980.

[87] M. Rioux. Laser range finder based on synchronized scanners. *Applied Optics*, 23(21):3837–3844, 1984.

[88] Peter J. Rousseeuw and Annick M. Leroy. *Robust Regression and Outlier Detection*. John Wiley & Sons, Inc., New York, 1987.

[89] Bikash Sabata and J. K. Aggarwal. Estimation of motion from a pair of range images: A review. *CVGIP: Image Understanding*, 54(3):309-321, November 1991.

[90] Brian G. Schunck. *Motion Segmentation and Estimation.* PhD thesis, MIT, Depatment of Electrical Engineering and Computer Science, 1983.

[91] Brian G. Schunck. The motion constraint equation for optical flow. In *Proceedings of International Conference on Pattern Recognition*, pages 20-22, Montreal, Canada, 1984.

[92] Brian G. Schunck. Image flow: Fundamentals and future research. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 560–571, San Francisco, June 1985. IEEE Computer Society, IEEE Computer Society Press.

[93] Brian G. Schunck. Image flow segmentation and estimation by constraint line clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(10):1010 1027, October 1989.

[94] S. K. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):56–73, January 1987.

[95] P. Spoer. Displacement estimation for objects on moving background. In T.S.Huang, editor, *Image sequence Processing and Dynamic scene analysis*, pages 424 436. Springer-Verlag, Berlin, 1983.

[96] F. W. M. Stentiford and R. G. Mortimer. Some new heuristics for thinning binary handprinted characters for ocr. *IEEE Transactions on Systems, Man, and Cybernetics*, 13(1), January 1983.

[97] E. M. Stokely and S. Y. Wu. Surface parametrization and curvature measurement of arbitrary 3-d objects: Five practical methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8), Aug. 1992.

[98] Muralidhara Subbarao. Interpretation of image flow: A spatio-temporal approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(3):266–278, March 1989.

[99] William B. Thompson and Stephen T. Barnard. Lower-level estimation and interpretation of visual motion. *Computer*, 14(8):20–28, August 1981.

[100] Roger Y. Tsai and Thomas S. Huang. Estimating 3-d motion parameters of a rigid planar patch, i: singular value decomposition. *IEEE Trans. Acoust. Speech Signal Processing*, 29:1147–1152, Dec. 1982.

[101] Roger Y. Tsai and Thomas S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(1):13–27, January 1984.

[102] Roger Y. Tsai, Thomas S. Huang, and W. L. Zhu. Estimating three-dimensional motion parameters of a rigid planar patch, ii: singular value decomposition. *IEEE Trans. Acoust. Speech Signal Processing*, 30:525–534, Aug. 1982.

[103] S. Ullman. *The Interpretation of Visual Motion.* MIT Press, Cambridge, MA, USA, 1979.

[104] S. Ullman. Maximizing rigidity: the incremental recovery of 3d structure from rigid and nonrigid motion. *Perception*, 13:255–274, 1984.

[105] S. Umeyama and T. Kasvand. A new alogorithm for point and plane pattern matching. ERB-1009 28455, National Research Council Canada, Ottawa,Ontario, November 1987.

[106] Shinji Umeyama. An eigendecomposition approach to weighted graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695-703, September 1988.

[107] J. F. Vega-Riveros and K. Jabbour. Review of motion analysis techniques. *IEE Proceedings*, 136, Pt. I(6):397-404, December 1989.

[108] A. Verri and T. Poggio. Against quantitative optical flow. In *Proceedings of the IEEE First International Conference on Computer Vision*, pages 171–180, London, England, June 1987. IEEE Computer Society, IEEE Computer Society Press.

[109] Alessandro Verri and Tomaso Poggio. Motion field and optical flow: Qualitative properties. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5):490–498, May 1989.

[110] Allen M. Waxman. An image flow paradiam. In Martin A. Fischler and Oscar Firschein, editors, *Readings in Computer Vision*, chapter Recovering Scene Geometry, pages 145–168. Morgan Kaufmann, 95 First Street, Los Altos, California 94022, 1987.

[111] Allen M. Waxman and Shimon Ullman. Surface structure and three-dimensional motion from image flow kinematics. *The International Journal of Robotics Research*, 4(3):72–94, 1985.

[112] Allen M. Waxman and Kwangyoen Wohn. Contour evolution, neighborhood deformation, and global image flow: Planar surfaces in motion. *The International Journal of Robotics Research*, 4(3):95–108, 1985.

[113] J. A. Webb and J. K. Aggarwal. Structure and motion of rigid and jointed objects. *Artifitial Intelligence*, 19:107–130, 1982.

[114] Edwin Bidwell Wilson. *Vector Analysis*. Dover Publication, Inc., New York, 1960.

[115] A. K. C. Wong. An algorithm for constellation matching. In *Proceedings of the 8th International Conference of Pattern Recognition*, pages 546–554, 1986.

[116] Masanobu Yamamoto. A general aperture problem for direct estimation of 3-d motion parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5):528–541, May 1989.

[117] Masanobu Yamamoto, J. Angelo Beraldin, Marc Rioux, and Jacques Domey. Direct estimation of deformable motion parameters from range image sequences. In

Proceedings of the IEEE Third International Conference on Computer Vision, pages 460-463, International House Osaka, Osaka, Japan, December 1991. IEEE Computer Society Press, IEEE Computer Society Press.

[118] B. L. Yen and T. S. Huang. Determining 3-D motion and structure of a rigid body using straight line correspondences. In T.S.Huang, editor, *Image sequence Processing and Dynamic scene analysis*, pages 365–394. Springer-Verlag, New York, NY, 1983.

[119] Alan L. Yuille and Norberto M. Grzywacz. A computational theory for the perception of coherent visual motion. *Nature*, 333(5):71–74, May 1988.

[120] Alan L. Yuille and Norberto M. Grzywacz. The motion coherence theory. In *Second International Conference on Computer Vision*, pages 344–353, Tampa, FL, USA, December 1989. The Computer Society of the IEEE, IEEE Computer Society Press.

[121] P. Y. Zhu, T. Kasvand, and A. Krzyzak. Motion estimation based on point correspondence using neural network. In *International Joint Conference on Neural Network*, pages 869–874, San Diago, CA, June 1990.

[122] P. Y. Zhu, T. Kasvand, and A. Krzyzak. Recovering motion from range image sequences. In *SPIE Proceeding: Intelligent Robotic Systems*, pages 1611–1623. Boston, November 1991.

[123] P. Y. Zhu, A. Krzyzak, and T. Kasvand. The local measurement and global interpretation of 3d motion field generated by several moving objects from range image sequence. In *Canadian Conference on Electrical and Computer Engineering*, pages 20.1.1-20.1.4, Quebec, Quebec, September 1991.

[124] P. Y. Zhu, A. Krzyzak, and T. Kasvand. Range image segmentation based on coherent motion. In *The 14th Symposium on information theory and its application*, pages 563–566. Ibusuki, Japan, December 1991.

[125] X. Zhuang, T. S. Huang, and R. M. Haralick. Two-view analysis: A unified algorithm. *J. Opt. Soc. Amer.*, 3(9):1492–1500, September 1986.

Appendix A

Linear Algebra

This appendix includes some mathematic background used in the thesis. For details see [40, 114, 38]

A.1 Singular Value Decomposition

Theorem A.1 *Let*

$$\mathbf{A} = \sum_{i=1}^{r=rank(\mathbf{A})} \sigma_i u_i v_i^T$$

where $\mathbf{U} = [\mathbf{u_1}, \cdots, \mathbf{u_m}], \mathbf{V} = [\mathbf{v_1}, \cdots, \mathbf{v_n}]$ *be the SVD of* $\mathbf{A} \in R^{m \times n} (m \geq n)$. *If* $\mathbf{b} \in R^m$ *then*

$$\mathbf{x}_{LS} = \sum_{i=1}^{r} (\mathbf{u_i}^T \mathbf{b} \sigma_i) \mathbf{v_i} \tag{A.1}$$

$$p_{LS} = \sum_{i=r+1}^{m} (\mathbf{u_i}^T \mathbf{b})^2 \tag{A.2}$$

Let

$$\mathbf{A}^+ = \mathbf{V} \Sigma^+ \mathbf{U}^T$$

where

$$\Sigma^+ = diag(\sigma_1^{-1}, \cdots, \sigma^{-1}, 0, \cdots, 0) \in R^{n \times m}$$

then

$$\mathbf{x}_{LS} = \mathbf{A}^{+}\mathbf{b} \qquad p_{LS} = \|(I - AA^{+})b\|^2$$

where $\mathbf{A}^{+}$ *is referred to as the pseudo-inverse of* $\mathbf{A}$, $\mathbf{x}_{LS}$ *is the least squares solution and* p_{LS} *is the projection of* $\mathbf{b}$ *on the range space of* $\mathbf{A}$.

A.2 Inverse of a 3×3 Matrix

Theorem A.2 *If* $\mathbf{B} \in \mathcal{R}^{3\times3}$ *and* $\mathbf{B}$ *is nonsingular, then there exists a unique inverse matrix* $\mathbf{B}^{-1}$

$$\mathbf{B}^{-1} = [\ \mathbf{c}_1 \quad \mathbf{c}_2 \quad \mathbf{c}_3\ ] = [\ \frac{\mathbf{b}_2\times\mathbf{b}_3}{(\mathbf{b}_2\times\mathbf{b}_3)\bullet\mathbf{b}_1} \quad \frac{\mathbf{b}_3\times\mathbf{b}_1}{(\mathbf{b}_3\times\mathbf{b}_1)\bullet\mathbf{b}_2} \quad \frac{\mathbf{b}_1\times\mathbf{b}_2}{(\mathbf{b}_1\times\mathbf{b}_2)\bullet\mathbf{b}_3}\ ] \tag{A.3}$$

where $\mathbf{c}_i$ *is* i^{th} *column of* $\mathbf{B}^{-1}$ *and* $\mathbf{b}_i$ *is the* i^{th} *row of* $\mathbf{B}$.

Proof: It can be directly proved by calculating $\mathbf{B}\mathbf{B}^{-1} = \mathbf{I}$.

Appendix B

Differential Geometry Review

Classical differential geometry is the study of local properties of curves and surfaces. It is used to motivate and justify the choice of surface features. For this reason, the basic concepts and terms of differential geometry are briefly reviewed here. For more information, the reader should consult one of the many books on the subject [26, 70].

B.1 Space Curves

Some local properties of curves appear while studying surfaces, therefore, this section will be used for a brief treatment of curves.

General space curves are represented parametrically as a function of an interval of the real value. A curve C is written as follows

$$C = \{\mathbf{X}(s) : (x(s), y(s), z(s)) \in \mathcal{R}^3, a < s < b\} \qquad (B.1)$$

where s is parameter. It is assumed that curves are parameterized by arc length in order to have a better geometric explanation. Only smooth curves are considered with components of $\mathbf{X}(s)$ having continuous second order derivatives.

The first derivative $\mathbf{X}'(s) = \frac{d\mathbf{X}(s)}{ds}$ of C at point s is called the *tangent vector* of the curve C at s. When s is the arc length, the tangent vector $\mathbf{X}'(s)$ is a unit vector. The norm $\|\mathbf{X}''(s)\| = \|\frac{d\mathbf{X}'(s)}{ds}\|$ of the second order derivative measures the rate of change of the angle which neighboring tangents make with the tangent at s. It is called the *curvature*

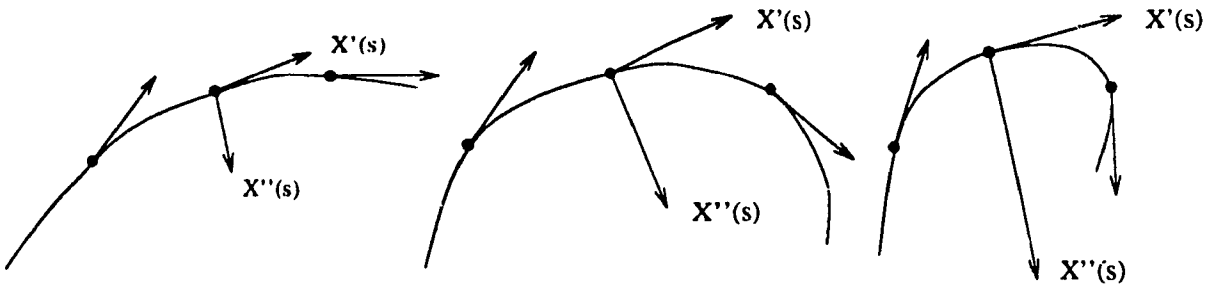


Figure B.1: Characteristics of a space curve, where $\mathbf{X}'(s) = \frac{d\mathbf{X}(s)}{ds}$ and $\mathbf{X}''(s) = \frac{d\mathbf{X}'(s)}{ds}$.

of C at s, denoted by $\kappa(s) = \|\mathbf{X}''(s)\|$. $\kappa(s)$ is a measure of how rapidly the curve "pulls away" from the tangent line at s, in a neighborhood of s (see Figure B.1).

A unit vector $\mathbf{n}(s)$ in the direction $\mathbf{X}''(s)$ is well defined by the equation $\mathbf{X}''(s) = \kappa(s)\mathbf{n}(s)$. Moreover, $\mathbf{X}''(s)$ is normal to $\mathbf{X}'(s)$, thus $\mathbf{n}(s)$ is normal to $\mathbf{X}'(s)$ and is called the *normal vector*.

Let $\mathbf{t}(s) = \mathbf{X}'(s)$ denote the unit tangent vector of C at s, then $\mathbf{t}(s)$ and $\mathbf{n}(s)$ determine a plane called the *osculating plane*. The unit vector $\mathbf{b}(s) = \mathbf{t}(s) \times \mathbf{n}(s)$ is normal to the osculating plane, and is therefore called the *binormal vector* of C at s. Since $\mathbf{b}(s)$ is a unit vector, the length $\|\mathbf{b}'(s)\|$ measures the rate of change of the neighboring osculating planes with the osculating plane at s; thus, $\mathbf{b}'(s)$ measures how rapidly the curve "pulls away" from the the osculating plane at s in a neighborhood of s (see Figure B.2). $\mathbf{b}'(s)$ is parallel to $\mathbf{n}(s)$, and may be written as

$$\mathbf{b}'(s) = \tau(s)\mathbf{n}(s) \tag{B.2}$$

for some function $\tau(s)$. $\tau(s)$ is called the *torsion* of C at s.

Three orthogonal unit vectors $\mathbf{t}(s), \mathbf{n}(s)$ and $\mathbf{b}(s)$ are referred to as the *Frenet trihedron* at s. Their derivatives $\mathbf{t}'(s), \mathbf{n}'(s)$ and $\mathbf{b}'(s)$, when expressed in the basis $\{\mathbf{t}, \mathbf{n}, \mathbf{b}\}$, yield the famous *Frenet formulas*, which can be written as a matrix of ordinary differential equations

$$\begin{bmatrix} \mathbf{t}'(s) \\ \mathbf{n}'(s) \\ \mathbf{b}'(s) \end{bmatrix} = \begin{bmatrix} 0 & \kappa(s) & 0 \\ -\kappa(s) & 0 & -\tau(s) \\ 0 & \tau(s) & 0 \end{bmatrix} \begin{bmatrix} \mathbf{t}(s) \\ \mathbf{n}(s) \\ \mathbf{b}(s) \end{bmatrix} \tag{B.3}$$

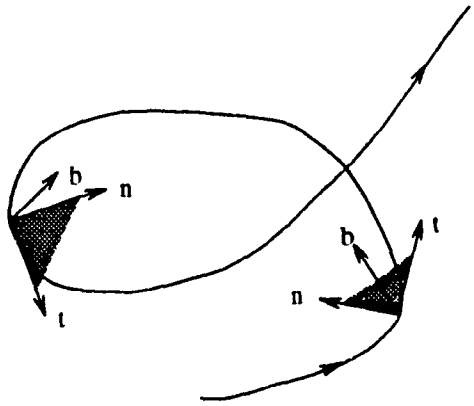


Figure B.2: Osculating planes determined by vectors **t** and **n**.

The well known fact that a smooth continuous curve C in the Euclidean 3D space is fully specified by the Frenet formulas comes from the following fundamental theorem of the local theory of curves.

Theorem B.3 *Given differentiable function $\kappa(s) > 0$ and $\tau(s) > 0$, there exists a smooth parametrized curve C such that s is the arc length, $\kappa(s)$ is the curvature, and $\tau(s)$ is the torsion of C. Moreover, any other curve $\bar{C}$, satisfying the same conditions, differs from C only by a rigid motion.*

The above theorem is not restricted to arc length. For details see [26].

Curvature, torsion and the parameter s uniquely determine the shape of the curve. They are invariant to rigid motions and have one-to-one relationship with curve shapes. These characteristics are the ideal candidates for feature matching. Surface characteristics with similar properties are discussed in the next section.

B.2 Surface

The parametric form of a general surface with respect to a known coordinate system may be written as follows:

$$S = \left\{ \mathbf{X} \in \mathcal{R}^3 : \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{bmatrix}, (u,v) \in D \subseteq \mathcal{R}^2 \right\} \tag{B.4}$$

This general parametric representation is referred to as $\mathbf{X}(u, v)$ where the x-component of $\mathbf{X}$ is $x(u, v)$, the y-component is $y(u, v)$, and the z-component is $z(u, v)$. Range image surface functions are represented by the graph surface form, where $x = x(u, v) = u, y = y(u, v) = v$ and $z = z(x, y)$. A surface can also be represented by an implicit form $F(x, y, z) = 0$, but this kind of surface representation is not considered here. Only regular parametric surfaces are considered. The definition of a regular surface can be found in [26]. Roughly speaking, it means that all three parametric functions $x(u, v), y(u, v)$ and $z(u, v)$ are smooth, and possesses continuous second order partial derivatives.

There are two basic mathematical entities that are considered in the differential geometry of surfaces. In classical differential geometry, they are known as the first and the second fundamental forms of a surface. Modern mathematics uses differential forms and favors an equivalent formulation of these quantifies in terms of the metric tensor and the Weingarten mapping (the "shape" operator). Complete knowledge of either of these forms at every surface point uniquely quantities general surface shape. In general, the modern approach is preferred, especially by mathematicians, because of its simpler formulas to work with once all the necessary terminology is established. The classical approach is favored in engineering since more people are familiar with partial derivatives than differential forms. In this thesis, 3D surface is introduced from the concepts of 3D curve because some ideas will be used to compute geometric identities later on, and it also allows a better geometric explanation.

For each point $p \in S$, there exists a set of curves which lie in S and pass through p, the set of tangent vectors to the curves constitutes a plane, which is called *tangent plane*, denoted $T_p(S)$.

The first fundamental form $I_p(S)$ of the surface S is defined as a quadratic form on $T_p(S)$

$$I_p(S) = \mathbf{w} \bullet \mathbf{w} \tag{B.5}$$

where w is a vector on the tangent plane $T_p(S)$. If the first fundamental form is expressed in the basis $\{\mathbf{X_u}, \mathbf{X_v}\}$ associated to a parametrization $\mathbf{X}(u, v)$

$$I_p(S) = (\mathbf{X_u}u' + \mathbf{X_v}v') \bullet (\mathbf{X_u}u' + \mathbf{X_v}v')$$

$$= E(u')^2 + 2Fu'v' + G(v')^2 \tag{B.6}$$

where

$$E = \mathbf{X_u} \bullet \mathbf{X_u} \qquad F = \mathbf{X_u} \bullet \mathbf{X_v} \qquad E = \mathbf{X_v} \bullet \mathbf{X_v}$$

and $\mathbf{X_u} = \frac{\partial \mathbf{X}}{\partial u}, \mathbf{X_v} = \frac{\partial \mathbf{X}}{\partial v}$ are refered to as the *u-tangent vector* and *v-tangent vector*, respectively.

The most important factor about the first fundamental form is that it allows us to make measurements on the surface (lengths of curves, angles of tangents, areas of regions) without referring back to the ambient space $\mathcal{R}^3$ where the surface lies.

The unit normal vector at each point of $\mathbf{X}(u, v)$ is given by

$$\mathbf{N} = \frac{\mathbf{X_u} \times \mathbf{X_v}}{\|\mathbf{X_u} \times \mathbf{X_v}\|} \tag{B.7}$$

It is also the normal vector of the tangent plane.

The second fundamental form of S at p is defined as

$$II_p(S) = -d\mathbf{N_p} \bullet \mathbf{w} \tag{B.8}$$

where $\mathbf{w}$ is a vector on the tangent plane $T_p(S)$.

Let C be a curve in S passing through p, κ the curvature of C at p, and θ is the angle between $\mathbf{n}$ and N, where $\mathbf{n}$ is the normal vector to C and N is the normal vector to S at p. The number $\kappa_n = \kappa cos(\theta)$ is then called the normal curvature of $C \in S$ at p (see Figure B.3(a)). If the curve is parametrized by the arc length, then

$$II_p(S) = \kappa_n \tag{B.9}$$

that is, the value of the second fundamental form for a unit vector $\mathbf{w} \in T_p(S)$ is equal to the normal curvature of a curve passing through p and tangent to $\mathbf{w}$. Given a unit vector $\mathbf{w} \in T_p(S)$, the intersection of S with the plane containing $\mathbf{w}$ and N is called the *normal section* of S at p along $\mathbf{w}$ (see Figure B.3(b)). The maximum normal curvature κ_1 and minimum normal curvature κ_2 are called the *principal curvatures* at p, the corresponding directions are called the *principal directions*.

The knowledge of the principal curvatures at p allows us to compute easily the normal curvature along given direction of $T_p(S)$. Let $\mathbf{w} \in T_p(S)$ with $\|\mathbf{w}\| = 1$, θ be

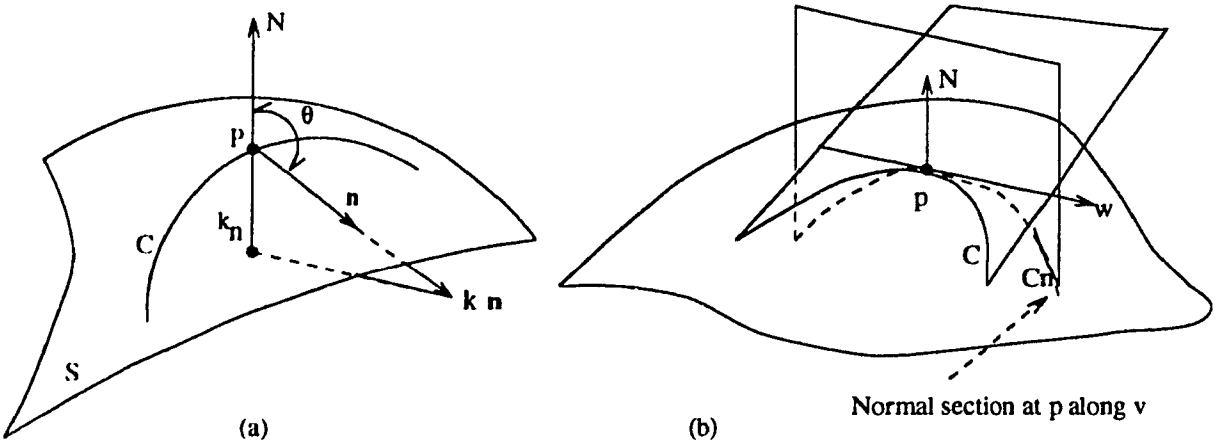


Figure B.3: (a) Normal curvature. (b) Normal section.

the angle between w and the maximum principle curvature direction, then the normal curvature κ_n along w is given by

$$\kappa_n = II_p(S) = \kappa_1 cos^2\theta + \kappa_2 sin^2\theta \tag{B.10}$$

This expression is known as the *Euler formula*, actually, it is the expression of the second fundamental form in the basis of the principal directions.

The expression of the second fundamental form in the basis $\{\mathbf{X_u}, \mathbf{X_v}\}$ is given by

$$II_p(S) = e(u')^2 + 2fu'v' + g(v')^2 \tag{B.11}$$

where

$$e = \mathbf{X_{uu}} \bullet \mathbf{N} \qquad f = \mathbf{X_{uv}} \bullet \mathbf{N} \qquad g = \mathbf{X_{vv}} \bullet \mathbf{N}$$

and $\mathbf{X_{uu}} = \frac{\partial^2 \mathbf{X}}{\partial u^2}, \mathbf{X_{uv}} = \frac{\partial^2 \mathbf{X}}{\partial u \partial v}, \mathbf{X_{vv}} = \frac{\partial^2 \mathbf{X}}{\partial v^2}$.

Similar to the Frenet formulas, the *Gauss-Weingarten equations* for 3D surfaces express the derivatives of the vectors $\mathbf{X_u}, \mathbf{X_v}, \mathbf{N}$ in the basis $\{\mathbf{X_u}, \mathbf{X_v}, \mathbf{N}\}$ by

$$\begin{bmatrix} \mathbf{X}_{uu} \\ \mathbf{X}_{uv} \\ \mathbf{X}_{vu} \\ \mathbf{X}_{vv} \\ \mathbf{N}_u \\ \mathbf{N}_v \end{bmatrix} = \begin{bmatrix} \Gamma^1_{11}\mathbf{X_u} + \Gamma^2_{11}\mathbf{X_v} + e\mathbf{N} \\ \Gamma^1_{12}\mathbf{X_u} + \Gamma^2_{12}\mathbf{X_v} + f\mathbf{N} \\ \Gamma^1_{21}\mathbf{X_u} + \Gamma^2_{21}\mathbf{X_v} + f\mathbf{N} \\ \Gamma^1_{22}\mathbf{X_u} + \Gamma^2_{22}\mathbf{X_v} + g\mathbf{N} \\ a_{11}\mathbf{X_u} + a_{12}\mathbf{X_v} \\ a_{21}\mathbf{X_u} + a_{22}\mathbf{X_v} \end{bmatrix} \tag{B.12}$$

where $\Gamma^k_{ij}, i,j,k = 1,2$ are called the *Christoffel symbols* of S. They are determined in terms of the coefficients of the first fundamental form E, F, G and their derivatives

$$\Gamma^1_{11}E + \Gamma^2_{11}F = \mathbf{X}_{uu} \bullet \mathbf{X}_\mathbf{u} = \frac{1}{2}E_u \tag{B.13}$$

$$\Gamma^1_{11}F + \Gamma^2_{11}G = \mathbf{X}_{uu} \bullet \mathbf{X}_\mathbf{v} = F_u - \frac{1}{2}E_v \tag{B.14}$$

$$\Gamma^1_{12}E + \Gamma^2_{12}F = \mathbf{X}_{uv} \bullet \mathbf{X}_\mathbf{u} = \frac{1}{2}E_v \tag{B.15}$$

$$\Gamma^1_{12}F + \Gamma^2_{12}G = \mathbf{X}_{uv} \bullet \mathbf{X}_\mathbf{v} = \frac{1}{2}G_u \tag{B.16}$$

$$\Gamma^1_{22}E + \Gamma^2_{22}F = \mathbf{X}_{vv} \bullet \mathbf{X}_\mathbf{u} = F_v + \frac{1}{2}G_u \tag{B.17}$$

$$\Gamma^1_{22}F + \Gamma^2_{22}G = \mathbf{X}_{vv} \bullet \mathbf{X}_\mathbf{v} = \frac{1}{2}G_v \tag{B.18}$$

where $a_{ij}, i,j = 1,2$ are obtained by

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = -\begin{bmatrix} e & f \\ f & g \end{bmatrix} \begin{bmatrix} E & F \\ F & G \end{bmatrix}^{-1} \tag{B.19}$$

This relation together with

$$\begin{cases} \mathbf{N}_\mathbf{u} = a_{11}\mathbf{X}_\mathbf{u} + a_{12}\mathbf{X}_\mathbf{v} \\ \mathbf{N}_\mathbf{v} = a_{21}\mathbf{X}_\mathbf{u} + a_{22}\mathbf{X}_\mathbf{v} \end{cases} \tag{B.20}$$

are known as the *equation of Weingarten.*

The determinant K of matrix (a_{ij})

$$K = det(a_{ij}) = \frac{eg - f^2}{EG - F^2} \tag{B.21}$$

is called *Gaussian curvature.* The negative of half of the trace of matrix (a_{ij}) is called the *mean curvature H*

$$H = -\frac{a_{11} + a_{22}}{2} = \frac{1}{2}\frac{eG - 2fF + gE}{EG - F^2} \tag{B.22}$$

It is easy to prove that

$$K = \kappa_1\kappa_2 \tag{B.23}$$

$$H = \frac{\kappa_1 + \kappa_2}{2} \tag{B.24}$$

and the principal curvatures κ_1, κ_2 are the roots of the equation

$$\kappa^2 - 2H\kappa + K = 0 \tag{B.25}$$

The fundamental theorem for 3D surfaces is

Theorem B.4 *Let E, F, G, e, f, g be differentiable functions, defined in an open set $\mathbf{v} \in \mathcal{R}^2$ with $E > 0$ and $G > 0$. Assume that the given functions satisfy formally the Gauss equation (B.26) and Mainardi-Codazzi equations (B.27) and that $EG - F^2 > 0$, then there exists an unique surface patch defined in the neighborhood of q such that E, F, G and e, f, g are the coefficients of the first and second fundamental forms, respectively. Furthermore, if two surfaces S and $\bar{S}$ possess coefficients of fundamental forms E, F, G, e, f, g and $\bar{E}, \bar{F}, \bar{G}, \bar{e}, \bar{f}, \bar{g}$ respectively such that the following relations hold*

$$E = \bar{E} \qquad F = \bar{F} \qquad G = \bar{G}$$

$$e = \bar{e} \qquad f = \bar{f} \qquad g = \bar{g}$$

then there exists an approriate rigid motion such that S and $\bar{S}$ coincide exactly implying they have the same shape.

Where the Gauss formula is

$$\begin{aligned} eg - f^2 \;=\; & F((\Gamma^2_{22})_u - (\Gamma^2_{12})_v + \Gamma^1_{22}\Gamma^2_{11} - \Gamma^1_{12}\Gamma^2_{12}) \\ & E(\Gamma^1_{22}\Gamma^1_{11} + \Gamma^2_{22}\Gamma^1_{12} - \Gamma^1_{12}\Gamma^1_{12} - \Gamma^2_{12}\Gamma^1_{22}) \end{aligned} \tag{B.26}$$

and the Mainardi-Codarri equations are

$$\begin{cases} e_v - f_u = e\Gamma^1_{12} + f(\Gamma^2_{12} - \Gamma^1_{11}) - g\Gamma^2_{11} \\ f_v - g_u = e\Gamma^1_{22} + f(\Gamma^2_{22} - \Gamma^1_{12}) - g\Gamma^2_{12} \end{cases} \tag{B.27}$$

This theorem tells us that an arbitrary smooth surface shape is captured by six scalar functions: E, F, G, e, f, g.

Unfortunately, it is difficult to interpret what each of these functions are individually telling us about surface shape. However, there are several combinations of these functions that yield more easily interpretable surface shape characteristics, particularly, mean curvature and Gaussian curvature or the principal curvatures and their directions. These curvature functions usually do not contain all the 3D shape information contained in E, F, G, e, f, g, but they do contain a substantial amount of useful information in some special cases. Details can be found in [13].

A set of Gaussian curvatures and mean curvatures are analytically equivalent to the set of principal curvatures. The principal curvatures have been chosen as candidate

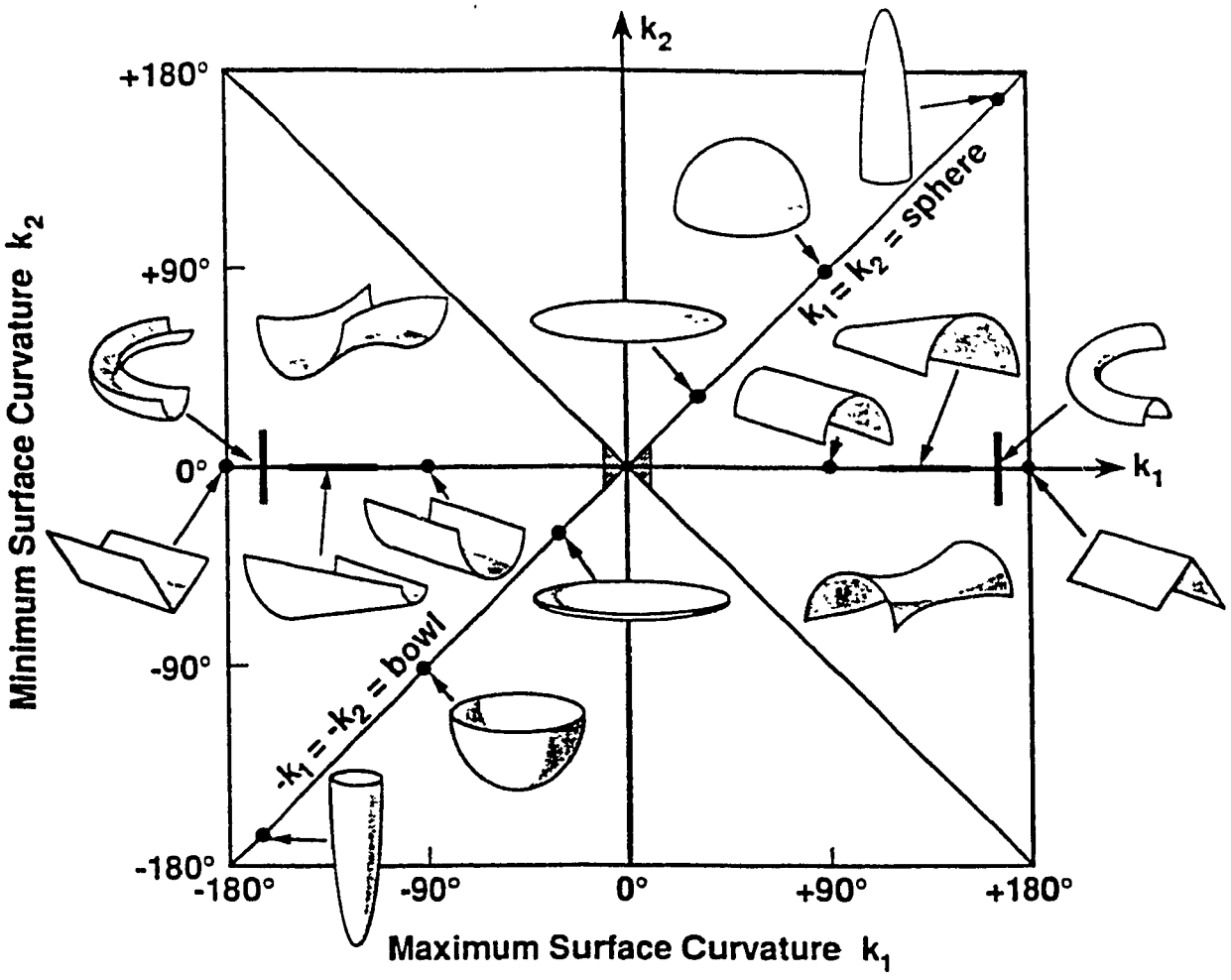


Figure B.4: Surface shapes and principal curvatures.

features instead of Guassian and mean curvatures mainly because two extra principal curvature directions could be useful for motion analysis as discussed later. Besides, principal curvatures provide better geometric explanations to commonly occurred surface shapes in our daily life as may be seen in Figure B.4. It is clear that if κ_1 and κ_2 are equal, then the local shape is spherical, and if κ_1 is a constant (larger than zero) and κ_2 is zero, then the local shape is cylindrical, etc. Since κ_1 and κ_2 are functions of E, F, G, e, f, g, they are invariant to a rigid motion.

Appendix C

Simulating 3D Rigid Motion

Simulating a 3D motion of an arbitrary object is a quite complex task since the z values are known only at the grid points. Theoretically, a point $\mathbf{p}$ is moved to point $\mathbf{p_1}$ according to the following relation

$$\mathbf{p_1} = \mathbf{Rp} + \mathbf{T} \tag{C.1}$$

where $\mathbf{R}$ and $\mathbf{T}$ denote rotation matrix and translation vector. However, $\mathbf{p_1}$ may not coincide with the grid point, it may become occluded, or may disappear from the field of view. A resampling of the transformed surface is needed. In addition, a point on the grid in the second image may correspond to a point which has not been on the grid in the first image or it comes from the surface which is invisible in the first image, and therefore some kind of extrapolation is necessary. How to deal with these problems is a subject of computer graphics, which is beyond of the scope of this work. In this appendix, an algorithm is presented to simulate 3D rigid motion used in the experiments in this thesis. It may not be as good as some algorithms used for animation. However, some inaccuracy, which can be considered as noise in data acquisition, is allowed on purpose.

The outline of the algorithm is as follows:

1. Given four grid points $(i,j),(i,j+1),(i+1,j),(i+1,j+1)$ in the first image, approximate the surface inside them by two plane facets $T1$ and $T2$, which are determined by points $P1P2P3$ and $P1P3P4$ (see Figure C.1(a)).

2. Move $P1P3P4$ to $P1'P3'P4'$ according to equation (C.1), $P1'P3'P4'$ forms facet $T1$ (see Figure C.1).

3. Find the external square of the facet $T1$ on the image plane (see Figure C.1(b)).

4. For each grid point inside the external square in the second image, find the intercepting point P of the light ray starting from the grid point and the facet $T1$, the z value of the intercepting point is taken as the z value of the grid point.

5. If the light ray intercepts with more than one facet, then the largest z value among all intercepting points is selected.

6. If the z value is smaller than 0, discard it since z value is the distance from the point to a reference surface, therefore, it has to be positive.

7. Repeat the above steps until all possible grid points in the first image have been considered.

8. Examine all grid points in the second image, and for the grid point whose z value is undetermined, calculate the z value by extrapolating the nearest useful facet.

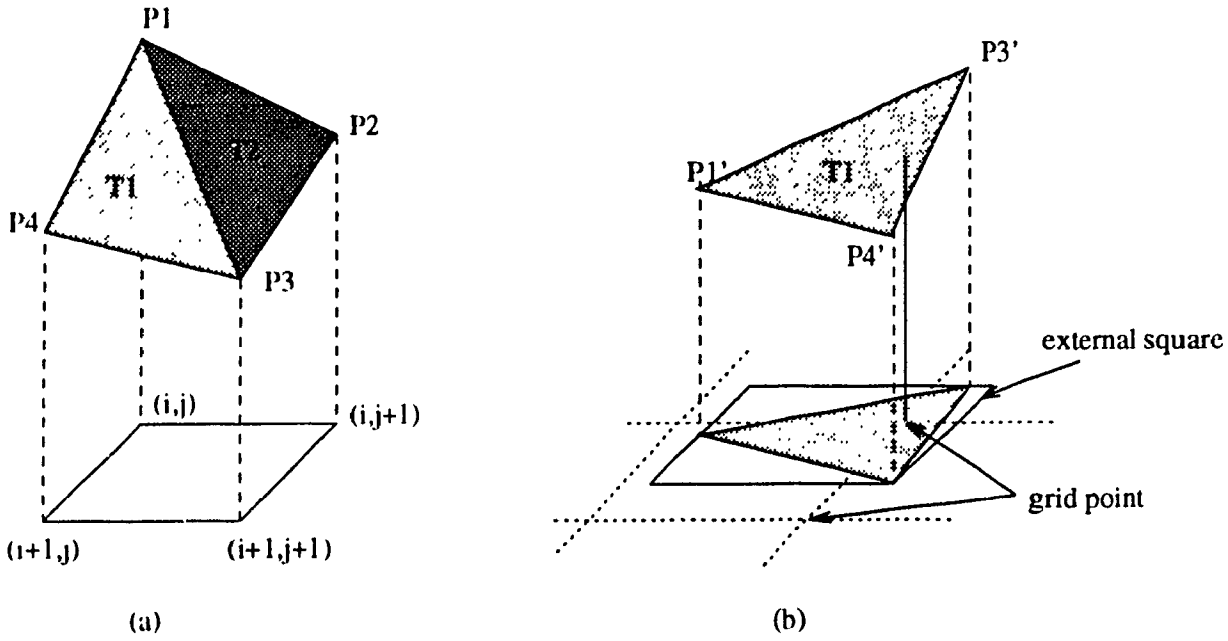


Figure C.1: Simulating 3D motion. (a) Approximating a local surface by plane facets. (b) Resampling by locating the intercepting point of a light ray and the facet.

Appendix D

Approximating Quadric

In this appendix, an algorithm will be described to approximate a quadratic surface. This algorithm is based on the one presented by Nagel in [80].

Given a $(2k+1) \times (2k+1)$ domain for the digitized range function $F(x,y)$ with $x,y = -k, -k+1, \cdots, -1, 0, 1, \cdots, k.$, the free parameters in the quadric

$$F(x,y) = F_0 + F_x x + F_y y + 1/2 F_{xx} x^2 + F_{xy} xy + 1/2 F_{yy} y^2 \tag{D.1}$$

are determined in such a way that the sum of the squared difference between the measurement and the approximating expression is minimized.

Let $(x_1, y_1), \cdots, (x_i, y_i), \cdots, (x_N, y_N)$ denote the raster points in the given domain, where the points are numbered by starting with the leftmost pixel in the top row, and proceeding row by row from left to right. N is the number of points in the given domain ($N = (2k+1)^2$). This order is used to introduce column vectors with N components for measurement values

$$\mathbf{g} = F(x_i, y_i), \qquad i = 1, 2, \cdots, N \tag{D.2}$$

and the approximation function

$$F(\mathbf{p}) = F(x_i, y_i; \mathbf{p}), \qquad i = 1, 2, \cdots, N, \tag{D.3}$$

where $\mathbf{p}$ represents the column vector parameters

$$\mathbf{p} = (p_1, p_2, p_3, p_4, p_5, p_6)^T = (F_0, F_x, F_y, F_{xx}, F_{xy}, F_{yy})^T \tag{D.4}$$

With these conventions, the expression to be minimized can be written

$$M = (F(\mathbf{p}) - \mathbf{g})^T (F(\mathbf{p}) - \mathbf{g}) \tag{D.5}$$

Let $\mathbf{A}$ denote the matrix of derivatives of F with respect to the components of $\mathbf{p}$, i.e.,

$$a_{il} = \partial F(x_i, y_i; \mathbf{p})/\partial p_l, \qquad i = 1, 2, \cdots, N, \quad l = 1, 2, \cdots, 6, \tag{D.6}$$

$$\mathbf{a_i} = (1, x_i, y_i, 1/2x_i^2, x_i y_i, 1/2y_i^2) \qquad i = 1, \cdots, N \tag{D.7}$$

Then $\mathbf{F}$ can be written as

$$\mathbf{F} = \mathbf{Ap} \tag{D.8}$$

Substituting $\mathbf{F}$ into equation (D.5) leads

$$M = (\mathbf{Ap} - \mathbf{g})^T (\mathbf{Ap} - \mathbf{g}) \tag{D.9}$$

The requirement to minimize this expression by appropriate choice of parameters yields

$$\partial M/\partial \mathbf{p} = 0 = \mathbf{A^T}(\mathbf{Ap} - \mathbf{g}) \tag{D.10}$$

or

$$\mathbf{p} = (\mathbf{A^T A})^{-1} \mathbf{A^T g} \tag{D.11}$$

Define

$$S_{pqr} = \sum_{i=1}^{N} x_i^p y_i^q g_i^r \tag{D.12}$$

Due to the symmetry of the domain, term with $r = 0$ and odd powers of x_i or y_i vanish. Using this convention, the matrix $\mathbf{A^T A}$ can be written in the form

$$\mathbf{A^T A} = \begin{pmatrix} S_{000} & 0 & 0 & 1/2S_{200} & 0 & 1/2S_{020} \\ 0 & S_{200} & 0 & 0 & 0 & 0 \\ 0 & 0 & S_{020} & 0 & 0 & 0 \\ 1/2S_{200} & 0 & 0 & 1/4S_{400} & 0 & 1/4S_{220} \\ 0 & 0 & 0 & 0 & S_{220} & 0 \\ 1/2S_{020} & 0 & 0 & 1/4S_{220} & 0 & 1/4S_{040} \end{pmatrix} \tag{D.13}$$

where

$$S_{000} = N = (2k+1)^2 \tag{D.14}$$

$$S_{200} = S_{020} = \frac{1}{3}k(k+1)(2k+1)^2 \tag{D.15}$$

$$S_{220} = \frac{1}{9}k^2(k+1)^2(2k+1)^2 \tag{D.16}$$

$$S_{400} = \frac{1}{15}k(k+1)(2k+1)^2(3k^2+3k-1) \tag{D.17}$$

from which it follows that

$$S_{000}S_{220} = S_{200}^2 = S_{020}^2 \tag{D.18}$$

$$\det(\mathbf{A}^{\mathrm{T}}\mathbf{A}) = \frac{1}{16}S_{000}^2S_{220}^2(S_{400}-S_{220})^2 \tag{D.19}$$

After tedious algebra, the element of **p** are

$$p_1 = F_0 = \frac{(S_{400}+S_{200})S_{001} - S_{200}S_{201} - S_{020}S_{021}}{S_{000}(S_{400}-S_{220})} \tag{D.20}$$

$$p_2 = F_x = S_{200}S_{101}/(S_{000}S_{220}) = S_{101}/S_{200} \tag{D.21}$$

$$p_3 = F_y = S_{020}S_{011}/(S_{000}S_{220}) = S_{011}/S_{020} \tag{D.22}$$

$$p_4 = F_{xx} = \frac{2(S_{000}S_{201} - S_{200}S_{001})}{S_{000}(S_{400}-S_{220})} \tag{D.23}$$

$$p_5 = F_{xy} = S_{111}/S_{220} \tag{D.24}$$

$$p_6 = F_{yy} = \frac{2(S_{000}S_{021} - S_{020}S_{001})}{S_{000}(S_{400}-S_{220})} \tag{D.25}$$

Appendix E

Representing a Moving Object in a World Coordinate System

Let $xyzt$ and $XYZt$ be the world coordinate system (WCS) and object-centered coordinate system (OCS) respectively as shown in Fig.E.1, and let for simplicity the OCS be aligned with the WCS at the initial time $t = t_0 = 0$. Assume that the surface of the moving object be represented by an analytic function in the OCS,

$$Z = F(X, Y, t) \tag{E.1}$$

where Z is allowed to be a function of spatial variables (X, Y) and t means that the surface of the moving object may change with the time, as in the case of a nonrigid object. In the case of a rigid object, Z is a function of spatial variables (X, Y) only.

Theorem E.5 *A uniformly translating second degree polynomial surface can be represented by the following analytic function in the WCS,*

$$z = z_0 + F_x x + F_y y + F_t t + F_{xt} xt + F_{yt} yt + F_{xy} xy \tag{E.2}$$

$$+1/2F_{xx}x^2 + 1/2F_{yy}y^2 + 1/2F_{tt}t^2 \tag{E.3}$$

Proof:

Since the object is undergoing rigid motion, the shape of its surface is unchanged with respect to the OCS, and therefore, at any time, it can be represented by

$$Z = f(X, Y) = a_0 + a_1 X + a_2 Y + a_3 XY + a_4 X^2 + a_5 Y^2 \tag{E.4}$$

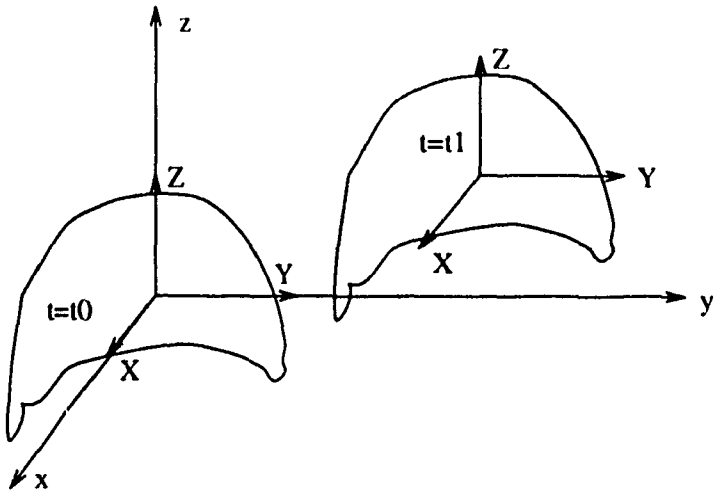


Figure E.1: $xyzt$ and $XYZT$ represent the world coordinate system (WCS) and object-centered coordinate system (OCS), where the OCS be aligned with WCS at the initial time $t = t_0 = 0$.

where $f(X, Y)$ is the second order polynomial function.

Let the object translate with a constant velocity $\mathbf{u} = [\ u_1 \quad u_2 \quad u_3\ ]^T$. A point $\mathbf{p_0} = [\ x_0 \quad y_0 \quad z_0\ ]^T$ on the object at time $t_0 = 0$ is moved to point $\mathbf{p_t} = [\ x \quad y \quad z\ ]^T$ at time t. The following relation between $\mathbf{p_0}$ and $\mathbf{p_t}$ holds in WCS

$$\mathbf{p_t} = \mathbf{p_0} + \mathbf{u}t \tag{E.5}$$

Since the WCS and the OCS are aligned with each other at time $t = 0$, then $p_0 = [\ X \quad Y \quad Z\ ]^T$ and

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} u_1 t \\ u_2 t \\ u_3 t \end{bmatrix} \tag{E.6}$$

Substituting the above relation into Eq.(E.4) yields

$$\begin{aligned} z &= F(x, y, t) \\ &= f(x - u_1 t, y - u_2 t) + u_3 t \\ &= a_0 + a_1(x - u_1 t) + a_2(y - u_2 t) + a_3(x - u_1 t)(y - u_2 t) \\ &\quad + a_4(x - u_1 t)^2 + a_5(y - u_2 t)^2 + u_3 t \\ &= b_0 + b_1 x + b_2 y + + b_3 t + b_4 xy + b_5 xt + b_6 yt + b_7 x^2 + b_8 y^2 + b_9 t^2 \end{aligned} \tag{E.7}$$

where

$$
\begin{aligned}
b_0 &= a_0 \\
b_1 &= a_1 \\
b_2 &= a_2 \\
b_3 &= u_3 - a_1 u_1 - a_2 u_2 \\
b_4 &= a_3 \\
b_5 &= -a_1 u_1 - a_3 u_2 - 2a_4 u_1 \\
b_6 &= -a_2 u_2 - a_3 u_1 - 2a_5 u_2 \\
b_7 &= -a_4 \\
b_8 &= -a_5 \\
b_9 &= a_3 u_1 u_2 + a_4 u_1^2 + a_5 u_2^2
\end{aligned}
$$

Obviously, Eq.(E.3) is equivalent to Eq.(E.7). Q.E.D.

If a second order surface is undergoing a translation and rotation, then the resulting surface in the WCS can no longer be expressed by an explicit function.

$$z = F(x, y, t) \tag{E.8}$$

Instead, it should be represented by an implicit form

$$F(x, y, z, t) = 0 \tag{E.9}$$

However, when the motion is small, this implicit function can be well approximated by an explicit function (theoretical proof can be found in [26]). $F(x, y, t)$ may be approximated by a polynomial function using Talor's series. In order to estimate rotation, the third order terms in the polynomial expression have to be kept. Therefore, the following polynomial function can be used to represent a surface in the WCS

$$
\begin{aligned}
Z &= F_x X + f_y Y + f_t t + f_{xy} XY + f_{xt} Xt + f_{yt} Yt + 1/2 f_{xx} X^2 + 1/2 f_{yy} Y^2 \\
&\quad + 1/2 f_{tt} t^2 + f_{xyt} XYt + 1/2 f_{xxt} X^2 t + 1/2 f_{yyt} Y^2 t
\end{aligned} \tag{E.10}
$$