Technical Report 1384

Robot Motion Vision By Fixation

M. Ali Taalebinezhaad

MIT Artificial Intelligence Laboratory

This blank page was inserted to preserve pagination.

Robot Motion Vision by Fixation

by

M. Ali Taalebinezhaad

© Massachusetts Institute of Technology 1992
All rights reserved

This empty page was substituted for a blank page in the original document.

Robot Motion Vision by Fixation

by

M. Ali Taalebinezhaad

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

In numerous current and future applications ranging from *autonomous navigation of mobile robots* to *collision avoidance systems for cars*, an imaging system (installed on a moving vehicle) takes $2D$ images of an environment with the *aim* of finding the *motion of the vehicle* (translational and rotational velocities) as well as the *structure of the environment* (shape). In machine vision, this problem is referred to as the *general motion vision problem.*

This thesis introduces a *direct method* called *fixation* for solving this general motion vision problem, arbitrary motion relative to an arbitrary environment. Avoiding *feature correspondence* and *optical flow* has been the motivation behind this *direct method* which uses the spatio-temporal brightness gradients of the images directly. The *fixation* method results in a linear constraint equation (*Fixation Constraint Equation*) which explicitly expresses the rotational velocity in terms of the translational velocity. The combination of this constraint equation with the *Brightness-Change Constraint Equation* (a fundamental equation which relates the motion to the brightness gradients at any image point) solves the general motion vision problem.

In contrast to previous direct methods, the fixation method does not impose any severe restrictions on the motion or the environment. Moreover, the fixation method neither requires *tracked images* as its input nor uses *tracking* for obtaining *fixated images.* Instead, it introduces a novel technique called the *pixel shifting process* to construct *fixated images* for any arbitrary *fixation point.* This is done entirely in software without any need to move the imaging system for *tracking.*

This fixation method has been successfully tested in the real world environment for the recovery of the motion and shape in the general case. The experimental results are presented and the implementation issues and techniques are discussed.

Thesis Supervisor: Dr. Berthold K. P. Horn
Title: Professor of Electrical Engineering and Computer Science

This empty page was substituted for a blank page in the original document.

Acknowledgments

I would like to express my sincere gratitude to Prof. Berthold Horn who introduced me to the field of machine vision, provided me with his valuable ideas and insightful comments, was accessible whenever I needed him, and gave me the freedom to pursue my research in my own way.

Thanks to Professors Eric Grimson and Jean-Jacques Slotine for reviewing my thesis drafts and providing me with their constructive suggestions, to Prof. Ellen Hildreth for her helpful comments on the theoretical part of this work in its early stage, to Professors William Durfee, Tomaso Poggio, Brian Schunck, Shimon Ullman, and David Wormley for their discussions about and interests in this work, and to Prof. Patrick Winston for making the MIT AI laboratory an exciting and outstanding place to conduct research in.

I would like to acknowledge Pawan Sinha who was ready at anytime to provide me with his valuable assistance and who patiently went through the whole thesis document and checked every aspect of it.

Special mention and sincere thanks go to two of my best friends whose kindness, etiquette and patience have always impressed me, and who never gave a negative response to my requests, Pawan Sinha (worth mentioning twice) and Rienk Bakker. Their friendship made my doctoral studies at MIT enjoyable and memorable.

I am grateful to wonderful people such as Gerry Brown, Priscilla Cobb, Noble Larson, Paula Mickevich, Leslie Regan, Maria Sensale, Jeanne Speckman, Bruce Walton, and Ron Wiken not just for their friendship but also for their help in providing me with the essential administrative and technical tools.

Many people in our AI lab have directly or indirectly had a constructive role in my work. Among those are Tao Alter, Todd Cass, Ron Chaney, David Clemens, John Harris, David Jacobs, David Michael, Henry Minsky, Sundar Narasimhan, Jerry Roylance, Amnon Shashua, Mark Torrance, and William Wells. I would specially like to thank Joachim Heel and Thomas Breuel for generously sharing with me their valuable knowledge and experience.

I am also thankful to my parents, brothers and sisters who supported me all along.

Gratitude of a special kind does not fit easily into a mere sentence, and I do indeed owe more than I could possibly say to my wife, Jamil, whose patience and understanding made the completion of this thesis possible.

This work was done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. The financial support for this laboratory was provided in part by DARPA under the ONR contracts N00014-85-K-0124 and N00014-91-J-4038.

To Jamshid

Acknowledgments

This work was done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. The financial support for this laboratory was provided in part by DARPA under the ONR contracts N00014-85-K-0124 and N00014-91-J-4038.

I would like to express my sincere gratitude to Prof. Berthold Horn who introduced me to the field of machine vision, provided me with his valuable ideas and insightful comments, was accessible whenever I needed him, and gave me the freedom to pursue my research in my own way.

Thanks to Professors Eric Grimson and Jean-Jacques Slotine for reviewing my thesis drafts and providing me with their constructive suggestions, to Prof. Ellen Hildreth for her helpful comments on the theoretical part of this work in its early stage, to Professors William Durfee, Tomaso Poggio, Brian Schunck, Shimon Ullman, and David Wormley for their discussions about and interests in this work, and to Prof. Patrick Winston for making the MIT AI laboratory an exciting and outstanding place to conduct research in.

I would like to acknowledge Pawan Sinha who was ready at anytime to provide me with his valuable assistance and who patiently went through the whole thesis document and checked every aspect of it.

Special mention and sincere thanks go to two of my best friends whose kindness, etiquette and patience have always impressed me, and who never gave a negative response to my requests, Pawan Sinha (worth mentioning twice) and Rienk Bakker. Their friendship made my doctoral studies at MIT enjoyable and memorable.

I am grateful to wonderful people such as Gerry Brown, Priscilla Cobb, Noble Larson, Paula Mickevich, Leslie Regan, Maria Sensale, Jeanne Speckman, Bruce Walton, and Ron Wilken not just for their friendship but also for their help in providing me with the essential administrative and technical tools.

Many people in our AI lab have directly or indirectly had a constructive role in my work. Among those are Tao Alter, Todd Cass, Ron Chaney, David Clemens, John Harris, David Jacobs, David Michael, Henry Minsky, Sundar Narasimhan, Jerry Roylance, Amnon Shashua, Mark Torrance, and William Wells. I would specially like to thank Joachim Heel and Thomas Breuel for generously sharing with me their valuable knowledge and experience.

I am also thankful to my parents, brothers and sisters who supported me all along.

Gratitude of a special kind does not fit easily into a mere sentence, and I do indeed owe more than I could possibly say to my wife, Jamil, whose patience and understanding made the completion of this thesis possible.

Contents

1 Introduction **11**
- 1.1 Motion Vision . . . 11
  - 1.1.1 Problem statement . . . 13
- 1.2 Previous Work (Main Approaches) . . . 13
  - 1.2.1 Optical flow . . . 13
  - 1.2.2 Feature correspondence . . . 14
  - 1.2.3 Direct methods . . . 15
- 1.3 Fixation Approach . . . 16
- 1.4 Contributions . . . 17
- 1.5 Thesis Structure . . . 19
  - 1.5.1 Part I: Theory . . . 19
  - 1.5.2 Part II: Implementation . . . 20
  - 1.5.3 Part III: Supplements . . . 21

2 Direct Methods **23**
- 2.1 Modeling and Coordinate System . . . 23
- 2.2 Basic Definitions . . . 24
- 2.3 The Brightness Change Constraint Equation . . . 26
  - 2.3.1 Rigid body motion . . . 27

3 Fixation Formulation **31**
- 3.1 Derivation of the General Fixation Constraint Equation . . . 32
  - 3.1.1 Derivation of special fixation constraint equation . . . 36
  - 3.1.2 Interpretation of the FCE . . . 38
- 3.2 Solving the General Direct Motion Vision Problem . . . 38
  - 3.2.1 Special cases: $\mathbf{t}$ is zero or parallel to $\mathbf{R}_o$ . . . 42

4 Computing the Fixation Velocity and Rotational Component $\omega_{\mathbf{R}_o}$ **45**
- 4.1 Algorithm . . . 46
- 4.2 Discussion . . . 48

5 Constructing a Sequence of Fixated Images **49**
5.1 Equivalent Rotational Velocity 50
5.2 Constructing the 2nd Fixated Image 52

6 An Overview of the Fixation Method **55**
6.1 Main Modules 55

7 Spatial and Temporal Brightness Gradients **57**
7.1 Visual Representation 57
7.2 Interpretation and Significance 58
7.3 Summary 59

8 The Effect of Fixation Patch Size **67**
8.1 Images with Moderate Relative Depth Changes 67
8.2 Images with Significant Relative Depth Changes 69
8.3 Finding a Good Estimate for $\omega_{\mathbf{R}_o}$ Autonomously 70
8.4 Updating the Fixation Velocity Using $\omega_{\mathbf{R}_o}$ 71
8.4.1 Improved estimations 74

9 Autonomous Choice of an Optimum Fixation Patch Size **77**
9.1 Normalized Error 77
9.2 Optimum Patch Size 78
9.2.1 Moderate changes in relative depth 78
9.2.2 Significant changes in relative depth 80
9.3 Autonomous Choice of Optimum Patch Size 81
9.3.1 Algorithm 82
9.3.2 Experimental results 83
9.3.3 Further results 84

10 Autonomous Choice of an Appropriate Fixation Point **91**
10.1 Algorithm 91
10.2 Discussion 92

11 Tracking without Moving the Camera **95**
11.1 Background 95
11.2 Pixel Shifting Process 96
11.2.1 Bilinear Interpolation 97
11.3 Construction of Fixated Images 98
11.4 Spatial and Temporal Gradient Maps 99
11.4.1 Landscape fixated image sequence 100
11.4.2 Cup fixated image sequence 101
11.5 Summary 101

12 Depth Map Recovery **107**

12.1 Introduction 107
12.2 Detecting the Depth Flaws 108
12.3 Constructing a Primary Depth Map 109
12.3.1 Filling in the Missing Depths 110
12.4 Improving the Depth Map 111
12.5 Even Better Depth Maps 112
12.6 Subsampling the Fixated Images 112
12.7 Summary 113

13 Calibration Issues **123**
13.1 Principal Point Calibration 124
13.1.1 Direct optical method 124
13.2 Calibration of the Rotation Axis 125
13.2.1 Generalization 126
13.3 Summary 127

14 Conclusions **129**
14.1 Features 129
14.2 Results 130
14.3 Assumptions 131
14.4 Shortcomings 131
14.5 Relation to Other Works 131
14.6 Future Extensions 132

A Derivation of Brightness Change Constraint Equation **135**

B Computation of Brightness Gradients **137**

C Depth at Fixation Point **139**

12.1 Introduction 107
12.2 Detecting the Depth Flaws 108
12.3 Constructing a Primary Depth Map 109
12.3.1 Filling in the Missing Depths 110
12.4 Improving the Depth Map 111
12.5 Even Better Depth Maps 112
12.6 Subsampling the Fixated Images 112
12.7 Summary 113

13 Calibration Issues **125**
13.1 Principal Point Calibration 124
13.1.1 Direct optical method 124
13.2 Calibration of the Rotation Axis 126
13.2.1 Generalization 126
13.3 Summary 127

14 Conclusions **129**
14.1 Features 129
14.2 Results 130
14.3 Assumptions 131
14.4 Shortcomings 131
14.5 Relation to Other Works 131
14.6 Future Extensions 132

A Derivation of Brightness Change Constraint Equation **135**

B Computation of Brightness Gradients **137**

C Depth at Fixation Point **139**

Introduction

Chapter 1

One of the principal objects of theoretical research in any department of knowledge is to find the point of view from which the subject appears in its greatest simplicity.

-Josiah Willard Gibbs

A little thought about the role of vision in the tasks that humans perform in their everyday life leaves no doubt about its importance. For the past several decades, physiologists and psychophysicists have been striving to understand the underlying mechanisms of human vision. On a parallel track, computer vision scientists have been working on the development of artificial systems for performing different visual tasks.

1.1 Motion Vision

In many applications, an imaging system (installed on a moving vehicle) takes 2D images of the environment. In *motion vision*, the goal is to find the *motion* of the moving vehicle (translational and rotational velocities) as well as the *shape* (structure of the environment), using a sequence of time varying images such as those shown in fig. 1-1.

Like many other vision problems, motion vision is extremely hard to accomplish. The difficulties stem from three major sources:

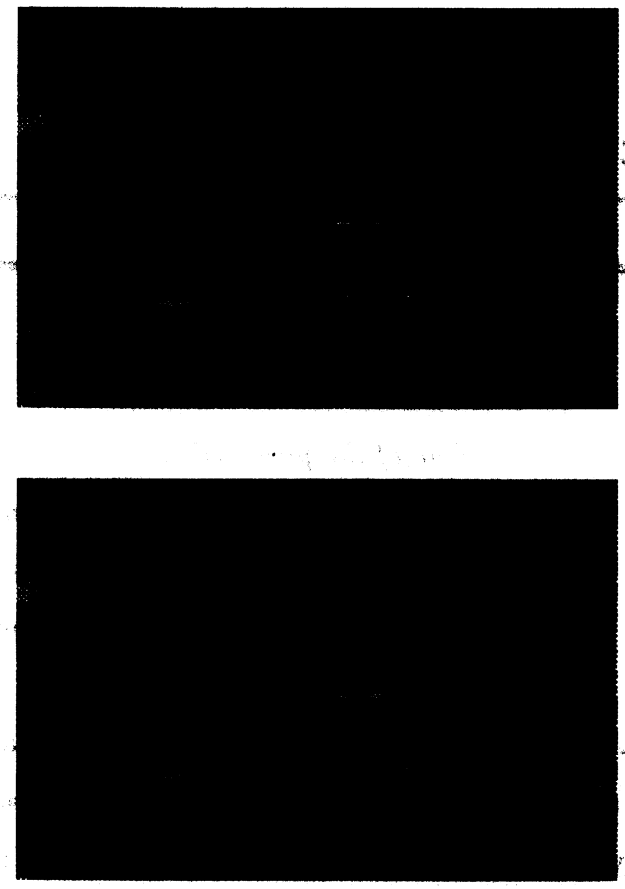

Figure 1-1: A sequence of real images where the motion between two images is a combination of translation and rotation.

- **Underconstrained:**

Deriving 3D information (motion and shape) from 2D data (images) is a severely under constrained problem (i.e. an infinite number of solutions are potentially consistent with the given data).

- **Huge Amount of Data:**

Processing even a single regular size image (512 $\times$ 512 *pixels*) requires handling of about a quarter million pixels worth of data.

- **Noise:** Real image data are very noisy.

1.1.1 Problem statement

The problem which we have addressed in this thesis can be summarized as follows:

> Finding the **motion** (*relative translation and rotational velocities*), and **shape** (*environment structure)* from a sequence of two real images by a **direct method** (**not** using either *optical flow* or *feature correspondence*) in the **general case** (*without* restricting the motion or shape).

1.2 Previous Work (Main Approaches)

People have been working on *motion vision* problems for several decades using three major techniques which are *optical flow, feature correspondence*, and *direct method.*

A survey of previous literature on machine vision is given in [11] and a partial list of last year papers in computer vision is compiled in [51]. Some of the current issues in image flow theory and motion vision are discussed in [88, 4, 55]. Much of the earlier work on recovering motion has been based on either establishing correspondences between the images of prominent features (points, lines, contours, and so on) in an image sequence, the so called *feature correspondence* [48, 80, 81, 35, 3] or establishing the velocity of points over the whole image, commonly referred to as the *optical flow* [8, 14, 2].

Each of the main approaches (*optical flow, feature correspondence*, and *direct methods*) are described briefly in this section and an example is given for each case using the real image sequence in fig. 1-1.

1.2.1 Optical flow

The computation of the local flow field exploits a constraint equation between the local brightness changes and the two components of the *optical flow.* This only gives the components of flow in the direction of the brightness gradient. To compute the full flow field, one needs additional constraints such as the heuristic assumption that

the flow field is locally smooth [30, 28]. This leads to an estimated optical flow field which may not be the same as the true motion field.

Figure 1-2 shows an optical flow field for the image sequence given in fig. 1-1. The size and direction of the apparent velocity at any pixel is shown by an arrow. Instead of the original images, such optical flow fields are used as a primary source of information in the optical flow techniques.

The irregular optical flows on the upper edge of this map are probably due to the noise and inherent errors involved in the computations at the image borders.

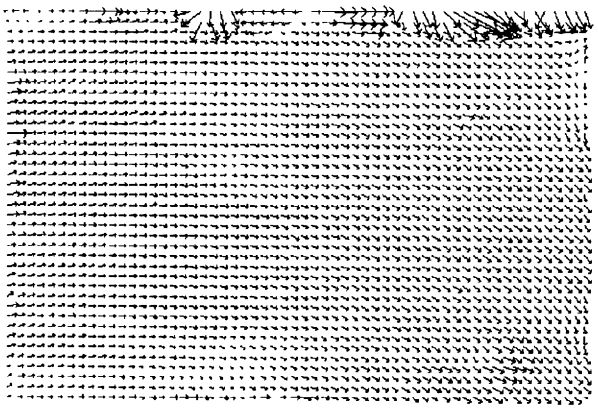

Figure 1-2: The optical flow map for the given real image sequence. The arrows show the magnitude and direction of the apparent motion at each point.

1.2.2 Feature correspondence

In general, identifying features here means determining gray-level corners. For images of smooth objects, it is difficult to find good features or corners. Furthermore, the *correspondence* problem has to be solved, that is, feature points from consecutive frames have to be matched.

Figure 1-3 shows the edge map for the top image in fig. 1-1. Several correspondence methods use such edge maps as the basic source of data instead of the original image. Then, they try to find some common features in different edge maps and relate them together.

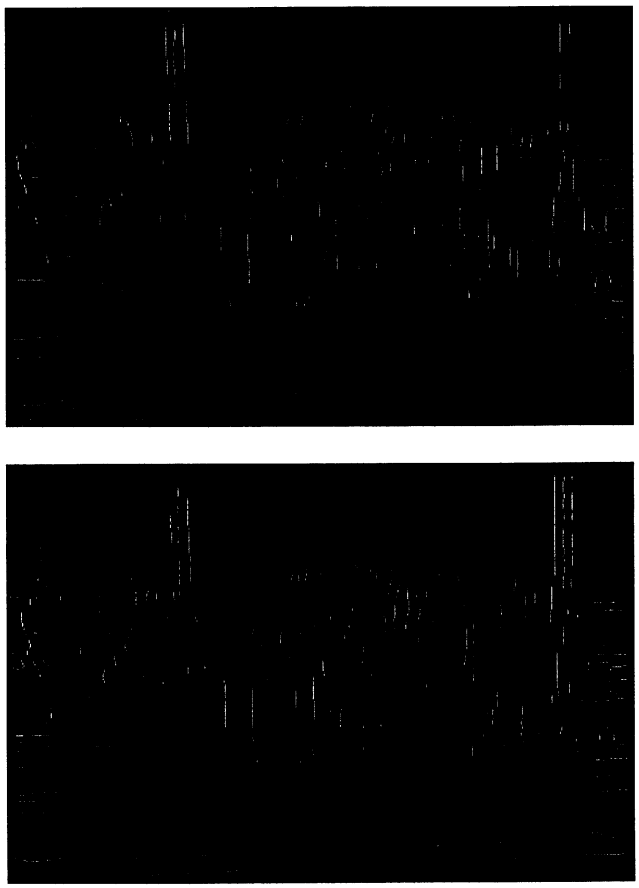

Figure 1-3: The edge maps for each of the images in the sequence.

1.2.3 Direct methods

The use of *optical flow* or *correspondence* techniques for solving motion vision problems has proven to be rather unreliable and computationally expensive [84, 83, 34].

These techniques spend a lot of effort on transforming the original images to the optical flow or the edge maps. The assumptions made in these procedures result in errors and loss of some useful information which exists in the original images.

These problems have motivated the investigation of *direct methods* which use the image brightness information directly to recover the motion and shape without any need to preprocess the original image.

Previous work in direct motion vision has used the *Brightness-Change Constraint*

Equation (BCCE) for rigid body motion [44]

$$E_t + \mathbf{v} \cdot \omega + \frac{\mathbf{s} \cdot \mathbf{t}}{Z} = 0 \tag{1.1}$$

to solve special cases such as *known depth* [30], *pure translation* or *known rotation* [31], *pure rotation* [31], and *planar world* [44]. Chapter 2 describes the details of this nonlinear equation which relates *depth* Z, *translational velocity* $\mathbf{t}$, and *rotational velocity* ω together.

All these direct methods are restricted in the types of motion or shape that they can handle. Our aim is to solve the *motion vision problem* in the general case using a direct method but without restricting either the motion or the shape to any special case.

1.3 Fixation Approach

This thesis presents a direct method called *fixation* for solving the motion vision problem in the general case without placing any restrictions on the motion or the shape [65, 69, 60]. The fixation method is based on the theoretical proof that for a sequence of fixated images (a sequence of images with one stationary image point in them), the 3D rotational velocity ω can always be explicitly expressed in terms of a linear function of the 3D translational velocity $\mathbf{t}$. Namely,

$$\omega = \omega_{\mathbf{R}_o} \hat{\mathbf{R}}_o + \frac{1}{\|\mathbf{R}_o\|} (\mathbf{t} \times \hat{\mathbf{R}}_o) \tag{1.2}$$

where $\hat{\mathbf{R}}_o$ is the unit vector along the position vector of the *fixation point* (a point in the image plane which stays stationary) and $\omega_{\mathbf{R}_o}$ is the component of rotational velocity about the fixation axis $\mathbf{R}_o$.

It should be emphasized that we do not need to know the real fixation point, if there is any, to take advantage of this *fixation constraint equation* (FCE), eqn. 1.2. In

fact, our algorithm allows us to choose virtually any point as the fixation point and obtain a sequence of fixated images [65, 69] by a simple software manipulation of one of the original images

The combination of the *Fixation Constraint Equation* (FCE), eqn. 1.2, and the BCCE, eqn. 1.1 offers a solution to the motion vision problem of arbitrary motion relative to an arbitrary rigid environment. That is, it allows recovery of the depth map Z, total 3D rotational velocity, and 3D translational velocity $\mathbf{t}$ without placing severe restrictions on the motion or the shape [65, 69].

1.4 Contributions

A summary of the principal contributions of this thesis are as follows.

- *Derivation of the Fixation Constraint Equation:*

Deriving a strong constraint equation called the *fixation constraint equation* (FCE). This constraint equation has a solid mathematical foundation. It expresses that for a sequence of fixated images, the rotational velocity can always be explicitly expressed as a linear function of translational velocity [69, 62, 61]. This equation is general and no hidden assumptions were made in its derivation.

- *Obtaining a solution to the general motion problem:*

Introducing a direct method called the *fixation method* which provides a solution for the general motion vision problem and has the following properties [69, 60, 63] :

- Finds the *motion* (*translational* and *rotational velocities*), and *shape* (the *environment structure*) from two monocular images.
- *Does not* restrict the motion or shape,
- *Does not* use either *optical flow* or *feature correspondence.*
- Is computationally *simple.*

- *Tracking without moving the camera:*

Presenting a novel method called the *pixel shifting process* for constructing a sequence

of *fixated (tracked) images* from any *arbitrary image sequence*, [65, 64]. It allows an *arbitrary* choice of *fixation point*, is fully software based, and does not require moving the camera for tracking.

- *Autonomous choice of an optimum fixation patch size:*

Finding a technique for autonomous choice of an *optimum fixation patch size* which results in good estimates for the motion parameters. This technique is based on defining a norm called *normalized error* and has been successfully implemented and tested on real images [68, 72, 66].

- *Autonomous choice of an appropriate fixation point location:*

Some regions of a given image are better for using a fixation patches. We have developed a method for autonomous choice of an *appropriate fixation point location* [67, 72].

- *Rotation axis calibration:*

Introducing a procedure for the *calibration* of a rotation axis in imaging systems. This technique is simple but useful and results in avoiding potential implementation errors [70, 72].

- *Representing image gradients:*

A novel method has been presented for visual representation of the *spatio-temporal gradients.* These intensity gradient maps allow one to visually understand the characteristics and significance of the brightness gradients [73, 70].

- *Constructing fixated (tracked) image sequences:*

Using the *pixel shifting process* and a *bilinear interpolation technique* we have constructed fixated images from real images [73, 70].

- *Depth map recovery from two monocular real images:*

We have recovered good depth maps from two monocular real images using the fixation method [71, 70].

1.5 Thesis Structure

This work comprises of three parts: *Theory*, *Implementation*, and *Appendices*.

1.5.1 Part I: Theory

This part covers the mathematical background of *direct methods* and the detailed theory of *fixation*.

- **Chapter 2**

We begin with a description of the camera model and coordinate system used in this work. Then, the *brightness change constraint equation* (BCCE) used by direct methods is explained.

- **Chapter 3**

This chapter presents the main idea behind our *fixation* method. It shows how the *Fixation Constraint Equation* (FCE) is derived and how it can be combined with the BCCE in order to solve for the *translational velocity* $\mathbf{t}$, *rotational velocity* $\boldsymbol{\omega}$, and the *depth* Z at any image point.

- **Chapter 4**

In an arbitrary image sequence, a point chosen as the *fixation point* does not necessarily stay stationary in the image plane. This chapter introduces the algorithms for the estimation of the apparent velocity at the fixation point (*fixation velocity*) which are required for the construction of a sequence of fixated images. Simultaneously, these algorithms find an estimate for the component of the rotational velocity along the fixation axis, $\omega_{\mathbf{R}_o}$, which appears in the FCE.

- **Chapter 5**

The fixation method requires a sequence of fixated images. This chapter shows how a sequence of fixated images can be constructed from an arbitrary image sequence using the components of the fixation velocity.

- **Chapter 6**

This chapter ends the basic theoretical part of the thesis by giving an overview of the main modules involved in the fixation method.

1.5.2 Part II: Implementation

This part presents the experimental results of applying the algorithms given in *Part I* to real image sequences. The implementation issues are described along with techniques for dealing with some practical problems.

- **Chapter 7**

The spatio-temporal brightness gradients of the images are the primary source of data used in our fixation method. This chapter introduces a novel technique for representing the gradients of real images. Such representations allow us to have a better insight about the characteristics and significance of gradients.

- **Chapter 8**

The experimental results in this chapter show that the estimated values for the components of the fixation velocity and $\omega_{\mathbf{R}_o}$ depend heavily on the size of the image patch used in the computation. It will be shown that depending on the image, and the fixation point location, there are some patch sizes which result in good estimates for the desired motion parameters.

- **Chapter 9**

This chapter presents a novel and reliable technique for autonomous choice of an *optimum fixation patch size* that results in good estimations for the motion parameters from real noisy images.

- **Chapter 10**

The fixation method does not place any restrictions on the choice of the fixation point and virtually any point can be chosen as the fixation point. However, some considerations should be taken into account when choosing a fixation point. For example, choosing a point at the center of a patch which has uniform brightness is not good because the motion is not detectable. This chapter introduces an autonomous

technique for choosing an appropriate fixation point.

- **Chapter 11**

Not only in our fixation technique but also in many other methods there is a substantial need for a sequence of fixated (tracked) images. This chapter introduces a novel method (*pixel shifting process*) for constructing a sequence of fixated images from an arbitrary image sequence using the components of the fixation velocity.

- **Chapter 12**

Using the estimated motion parameters and the constructed sequence of fixated images, this chapter describes the issues involved in recovering depth maps. Detailed techniques are presented for overcoming practical problems such as noise and inherent image deficiencies.

- **Chapter 13**

Camera calibration is usually an unavoidable requirement for working with real images. This chapter discusses some of the calibration issues that we faced in this work.

- **Chapter 14**

We conclude this work by giving a summary of the fixation method, results, features, assumptions, shortcomings, relation to other works, and finally some thoughts on the possible future extensions.

1.5.3 Part III: Supplements

Some of the relevant theoretical proofs and formulations are summarized in this part.

- **Appendix A**

Provides a detailed derivation of the BCCE.

- **Appendix B**

Presents the formulations for computing the spatio-temporal gradients.

- **Appendix C**

Describes a technique for computing the depth at the fixation point, Z_o.

technique for choosing an appropriate fixation point.

- **Chapter 11**

Not only in our fixation technique but also in many other methods there is a substantial need for a sequence of fixated (tracked) images. This chapter introduces a novel method (pixel shifting process) for constructing a sequence of fixated images from an arbitrary image sequence using the components of the fixation velocity.

- **Chapter 12**

Using the estimated motion parameters and the constructed sequence of fixated images, this chapter describes the issues involved in recovering depth maps. Detailed techniques are presented for overcoming practical problems such as noise and inherent image deficiencies.

- **Chapter 13**

Camera calibration is usually an unavoidable requirement for working with real images. This chapter discusses some of the calibration issues that we faced in this work.

- **Chapter 14**

We conclude this work by giving a summary of the fixation method, results, features, assumptions, shortcomings, relation to other works, and finally some thoughts on the possible future extensions.

1.5.3 Part III: Supplements

Some of the relevant theoretical proofs and formulations are summarized in this part.

- **Appendix A**

Provides a detailed derivation of the BCCE.

- **Appendix B**

Presents the formulations for computing the spatio-temporal gradients.

- **Appendix C**

Describes a technique for computing the depth at the fixation point, Z_o.

Direct Methods

Chapter 2

Images are usually obtained from a regular electronic camera where the projection is perspective. In this chapter, we first describe the camera model and the coordinate system used in this work. Then, a mathematical background of the BCCE is presented.

2.1 Modeling and Coordinate System

As shown in fig. 2-1, the coordinate system is attached to the camera so that its origin is located at the projection center.

The image plane is where the environment image is projected to. In an electronic camera, a CCD (*Charge Coupled Device*) plays the role of the image plane. The CCD is an electronic light-sensitive plane. It consists of a tessellation of small rectangular or square photo-sensitive cells which are called pixels. Each pixel of the CCD is electronically charged depending on the number of the photons it receives. Thus, the charge level of each pixel is a representation of the brightness at the corresponding point in the image plane. By reading and appropriate conversion of the camera charge level of all pixels, the image can be written in a file or displayed on a screen.

The image plane in our coordinate system is parallel to the $X - Y$ plane and is located at a distance equal to the *focal length* from it. The *optical axis* Z pierces the image plane at a point which is called the *principal point.* Any environment point $\mathbf{R}$ is projected to an image point $\mathbf{r}$ in this coordinate system.

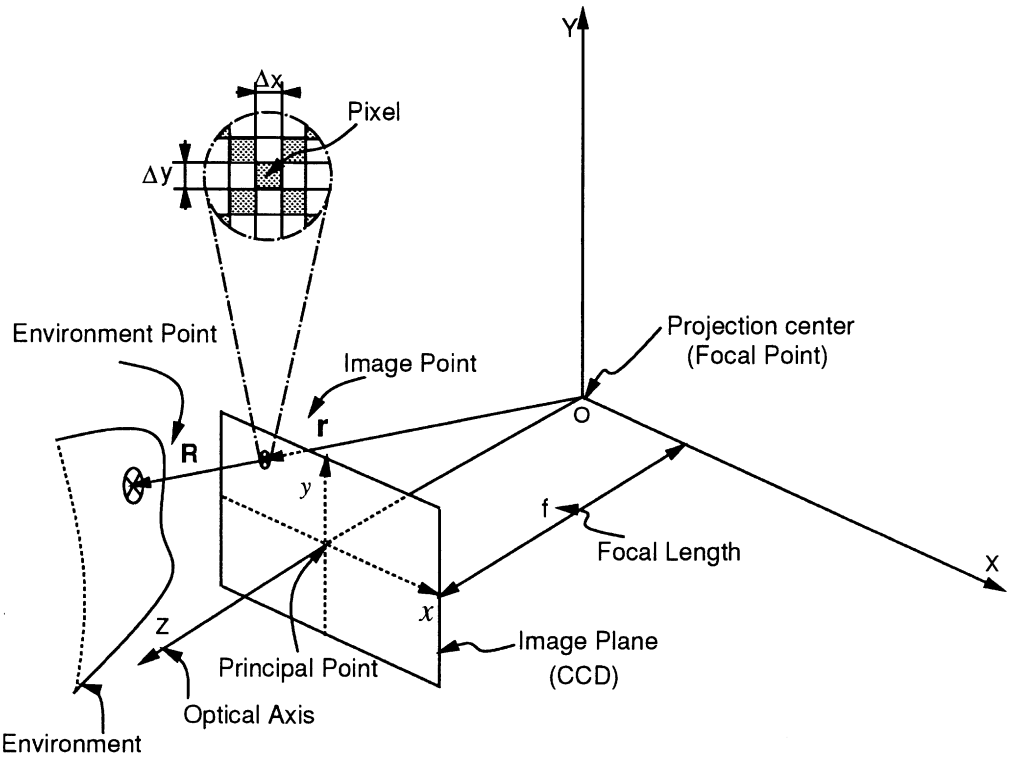

Figure 2-1: The coordinate system is attached to the camera and the projection is perspective.

2.2 Basic Definitions

Using a viewer-centered coordinate system which is adopted from Longuet-Higgins & Prazdny [36] is very common in direct motion vision. Figure 2-2 depicts the coordinate system under consideration.

In such a coordinate system, a world point

$$\mathbf{R} = (X\ Y\ Z)^T \tag{2.1}$$

is imaged at

$$\mathbf{r} = (x\ y\ 1)^T. \tag{2.2}$$

That is, the image plane has the equation $Z = 1$ or in other words the *focal length* f is 1. The origin is at the projection center and the Z-axis runs along the optical

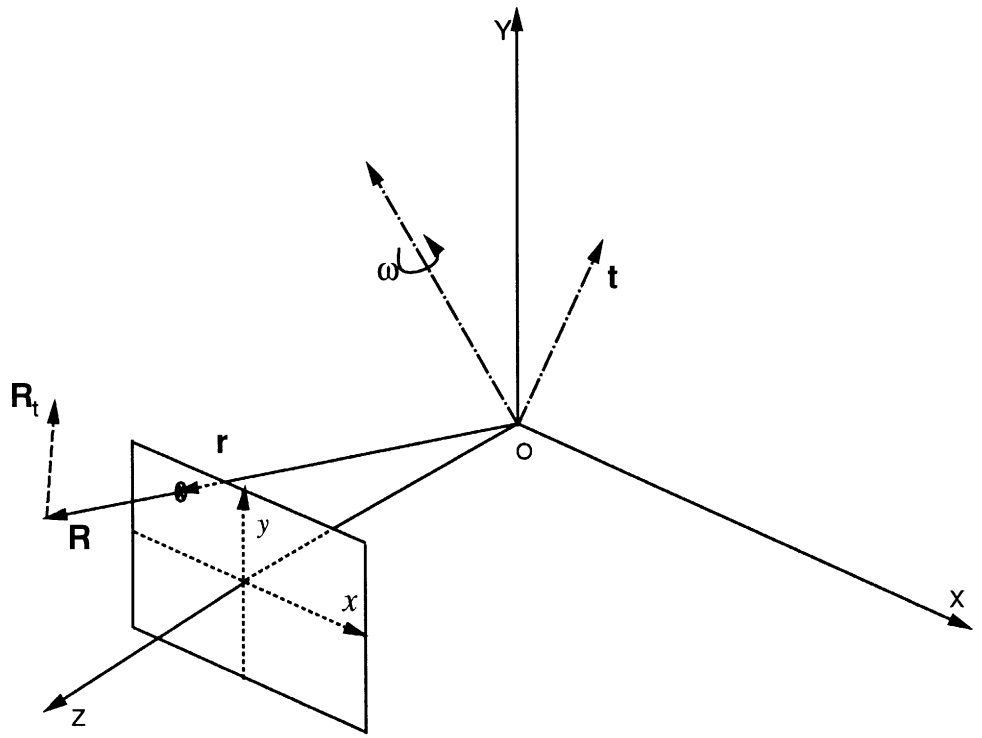

Figure 2-2: Under the effect of translational velocity of the viewer is $\mathbf{t} = (U\ V\ W)^T$ and rotational velocity is $\omega = (A\ B\ C)^T$, any environment point $\mathbf{R}$ has the velocity $\mathbf{R}_t$ from the observer's point of view.

axis. The X and Y axes are parallel to the x and y axes of the image plane. Image coordinates are measured relative to the *principal point*, the point $(0\ 0\ 1)^T$ where the optical axis pierces the image plane. The position vectors $\mathbf{r}$ and $\mathbf{R}$ are related by the perspective projection equation

$$\mathbf{r} = (x\ y\ 1)^T = \left(\frac{X}{Z}\ \frac{Y}{Z}\ \frac{Z}{Z}\right)^T = \frac{\mathbf{R}}{\mathbf{R}\cdot\hat{z}} \tag{2.3}$$

where $\hat{\mathbf{z}}$ denotes the unit vector along the $Z-$axis and $\mathbf{R}\cdot\hat{\mathbf{z}} = Z$.

When the observer moves with instantaneous translational velocity $\mathbf{t} = (U\ V\ W)^T$ and instantaneous rotational velocity $\omega = (A\ B\ C)^T$ relative to an environment, then the time derivative of the position vector of a point in the environment, $\mathbf{R}$, relative

to the observer can be written as

$$\mathbf{R}_t = -\mathbf{t} - \omega \times \mathbf{R}. \tag{2.4}$$

The motion of the world point $\mathbf{R}$ results in the motion of its corresponding image point $\mathbf{r}$. It can be shown that the motion field in the image plane is obtained by differentiating eqn. 2.3 with respect to time as in [44]

$$\mathbf{r}_t = \frac{d}{dt}\left(\frac{\mathbf{R}}{\mathbf{R}\cdot\hat{\mathbf{z}}}\right) = \frac{\hat{\mathbf{z}} \times (\mathbf{R}_t \times \mathbf{r})}{\mathbf{R}\cdot\hat{\mathbf{z}}}. \tag{2.5}$$

Substituting for $\mathbf{R}$, $\mathbf{r}$ and $\mathbf{R}_t$ from equations 2.1, 2.2, and 2.4 into eqn. 2.5 gives [36, 14]

$$\mathbf{r}_t = \begin{pmatrix} x_t \\ y_t \\ z_t \end{pmatrix} = \begin{pmatrix} \frac{-U+xW}{Z} + Axy - B(x^2+1) + Cy \\ \frac{-V+yW}{Z} - Bxy + A(y^2+1) - Cx \\ 0 \end{pmatrix}. \tag{2.6}$$

This result is just the *parallax equations of photogrammetry* that occur in the incremental adjustment of relative orientation [23, 42]. It shows how, given the environment motion, the motion field can be calculated for every image point.

2.3 The Brightness Change Constraint Equation

Image brightness changes are primarily due to the relative motion between an environment and an observer provided that the surfaces of the objects have sufficient texture and the lighting condition varies slowly enough both spatially and with time. In such cases (which may occur in practical applications), brightness changes due to the variations in the surface orientation and illumination can be neglected. Consequently, we may assume that the brightness of a small patch on a surface in the scene does not change during motion. As shown in appendix A, when the motion is small

the expansion of the total derivative of brightness E leads to

$$\frac{dE}{dt} = E_t + x_t E_x + y_t E_y = 0^1 \tag{2.7}$$

known as the *Brightness Change Constraint Equation* (BCCE) where (E_x, E_y) and E_t are spatial and temporal gradients of the image brightness at any given pixel [30, 54, 29].

Note that eqn. 2.7 does not hold for the special case that the viewer and the light source are stationary and the environment moves relative to them because the brightness of a surface patch does not remain constant in this case.

2.3.1 Rigid body motion

In rigid body motion, there is only one relative motion between the observer and the environment. For this case, we can substitute for x_t and y_t from eqn. 2.6 into eqn. 2.7, to obtain the brightness-change constraint equation for the rigid body motion [44] as

$$E_t + \mathbf{v} \cdot \omega + \frac{s \cdot \mathbf{t}}{Z} = 0. \tag{2.8}$$

This equation is nonlinear in terms of unknowns rotation ω, translation $\mathbf{t}$, and depth Z. The auxiliary vectors $\mathbf{s}$ and $\mathbf{v}$ are known at any pixel (x, y) and are defined as

$$\mathbf{s} = \begin{pmatrix} -E_x \\ -E_y \\ xE_x + yE_y \end{pmatrix} \tag{2.9}$$

[1] To account for smooth variations in the image brightness due to other factors such as shading, spatial and temporal illumination changes, and variations in reflectance properties, the BCCE can be extended to

$$E_t + x_t E_x + y_t E_y = m_t E + c_t$$

where in general m_t and c_t are time and position dependent [21, 45]. Cornelius & Kanade [17] also propose a method which allows gradual changes in $\frac{dE}{dt}$. These extensions are not discussed here.

and

$$\mathbf{v} = \begin{pmatrix} +E_y + y(xE_x + yE_y) \\ -E_x - x(xE_x + yE_y) \\ yE_x - xE_y \end{pmatrix}. \tag{2.10}$$

Since $\mathbf{s} \cdot \mathbf{r} = 0$, $\mathbf{v} \cdot \mathbf{r} = 0$ and $\mathbf{s} \cdot \mathbf{v} = 0$, the vectors $\mathbf{r}$, $\mathbf{s}$, and $\mathbf{v}$ form an orthogonal triad; see fig. 2-3. The vectors $\mathbf{s}$ and $\mathbf{v}$ represent inherent properties of the image. Also it can be shown that $\mathbf{v} = \mathbf{r} \times \mathbf{s}$. The vector $\mathbf{s}$ indicates the direction in which translation of a given magnitude will contribute maximally to the temporal brightness change of a given picture cell. The vector $\mathbf{v}$ plays a similar role for rotation.

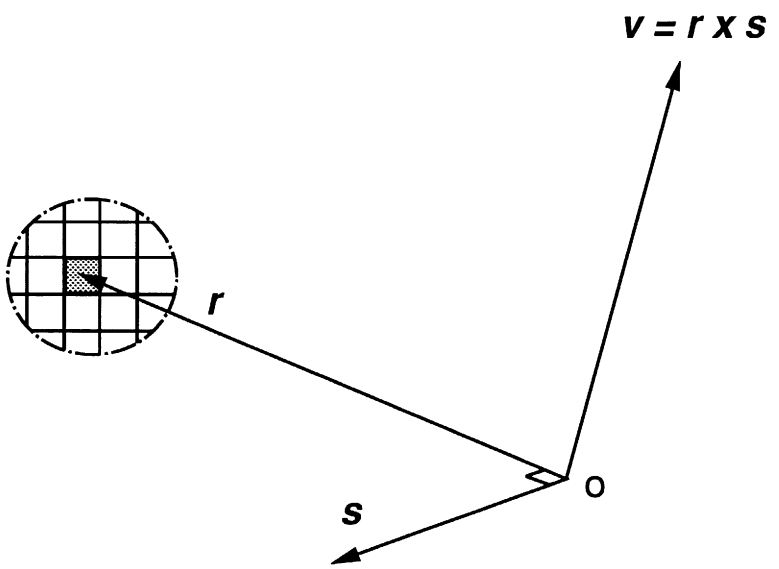

Figure 2-3: At any pixel, vectors $\mathbf{r}$ (pixel position), $\mathbf{s}$, and $\mathbf{v}$ form an orthogonal triad. Also $\mathbf{v} = \mathbf{r} \times \mathbf{s}$.

The BCCE, eqn. 2.7, does not change if we scale both Z and $\mathbf{t}$ by the same factor. Consequently, we can determine only the *direction* of translational velocity and the *relative depth* of points in the scene. This ambiguity is known as the *scale-factor ambiguity* in motion vision.

Equation 2.7 is obtained under the following assumptions:

- No noise,
- Sufficient surface texture,

- Slow spatio-temporal variations in lighting,
- Small motions between frames.

In real images, violation of any of these conditions may cause eqn. 2.7 not to be held at any single pixel. However, later we will show how this equation can be used in a least squares method for recovery of shape and motion from real image sequences.

- Slow spatio-temporal variations in lighting,
- Small motions between frames.

In real images, violation of any of these conditions may cause eqn. 2.7 not to be held at any single pixel. However, later we will show how this equation can be used in a least squares method for recovery of shape and motion from real image sequences.

Fixation Formulation

Chapter 3

Our common visual experience suggests that fixation may play an important role in the analysis of moving objects. When we want to understand the motion of an object, we do not keep our eyes and head stationary in front of the moving object. Instead, our head and/or eyes follow the moving object, in order to keep the image of a point of interest stationary in the retina. There are also some formal studies that support such observations [6, 7, 9]. In this computer vision work, the *fixation* is defined as:

> Given two subsequent images, *1st* and *2nd initial images*, and an arbitrary point in the 1st initial image, find a new image, *a 2nd fixated image*, such that the image of the selected point in the new image is located at its original position as in the 1st initial image.

This definition of fixation is shown schematically in fig. 3-1. If we choose point 1 in the *1st initial image* as the *fixation point*, its image in the *2nd initial image* may move to a new location such as 2. In chapter 5, we introduce a simple technique for converting the 2nd initial image in order to bring image point 2 to the same physical location as point 1. This process will construct the *2nd fixated image* and form a sequence of images fixated at point 1.

As shown in fig. 3-2, we refer to this arbitrary selected image point as the *fixation point*, $\mathbf{r}_o$, and to its corresponding point on the object as the *interest point*, $\mathbf{R}_o$.

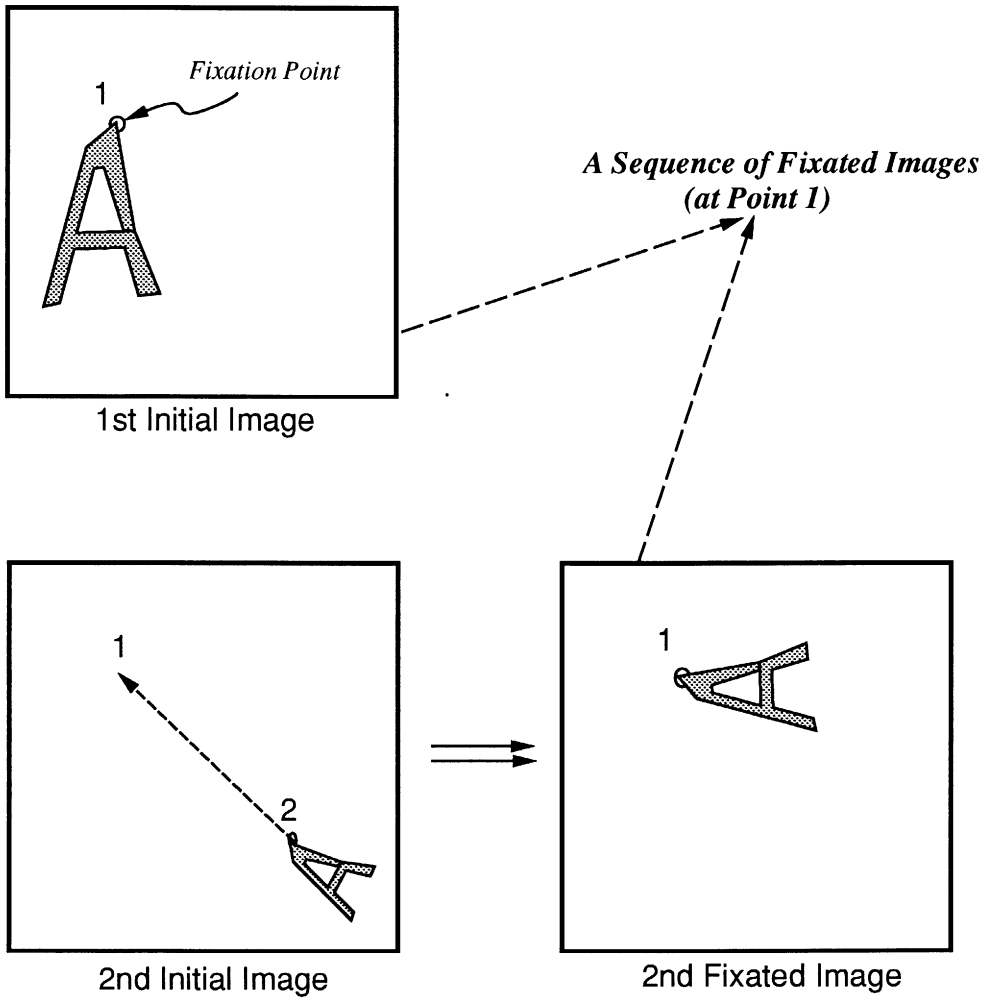

Figure 3-1: A schematic interpretation of *fixation point* and *fixated image sequence.*

3.1 Derivation of the General Fixation Constraint Equation

For a sequence of two fixated images, at the fixation point r_o we should have

$$\mathbf{r}_{ot} = 0 \tag{3.1}$$

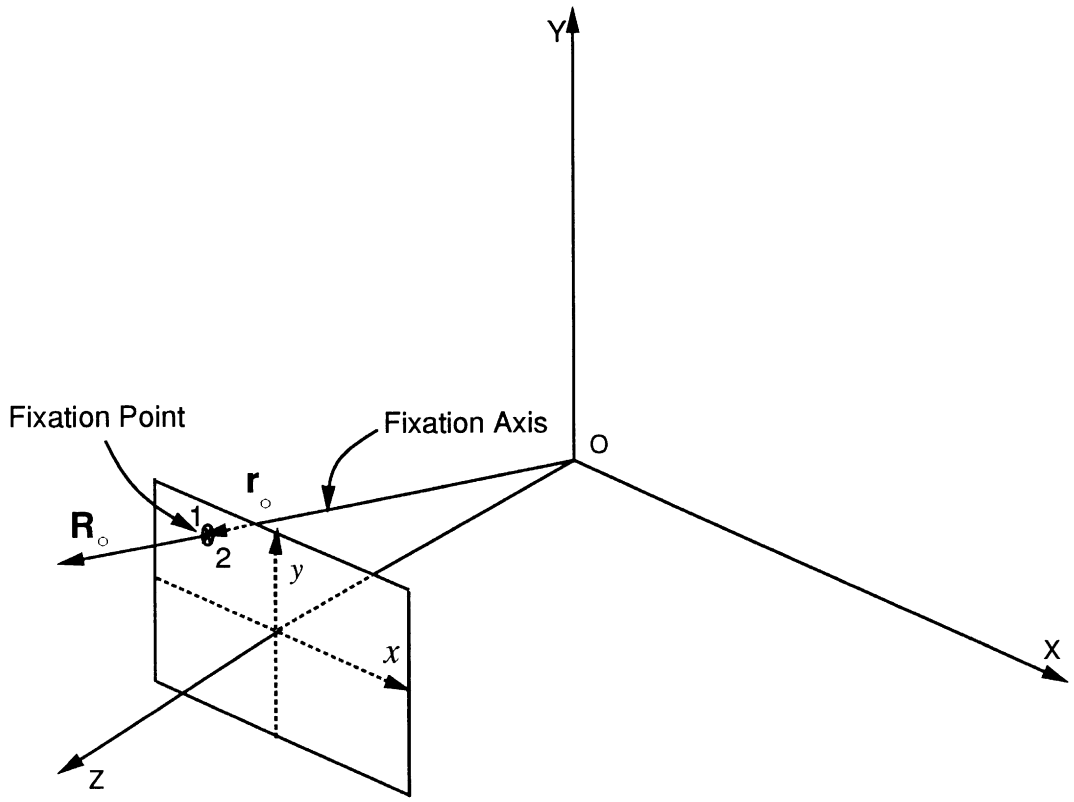

Figure 3-2: In the *fixation* method, the image of the *interest point,* the *fixation point,* is kept stationary in the image plane despite the relative motion between the camera and the environment.

where $\mathbf{r}_{ot}$ is the time derivative of the fixation point vector and similar to eqn. 2.5 it can be written as

$$\mathbf{r}_{ot} = \frac{\hat{\mathbf{z}} \times (\mathbf{R}_{ot} \times \mathbf{r}_o)}{\mathbf{R}_o \cdot \hat{\mathbf{z}}}. \tag{3.2}$$

$\mathbf{R}_{ot}$ is the time derivative of the interest point vector. Combination of equations 3.1 and 3.2 shows that for fixation we need to have

$$\hat{\mathbf{z}} \times (\mathbf{R}_{ot} \times \mathbf{r}_o) = 0. \tag{3.3}$$

In other words, we want to find out when $\mathbf{R}_{ot} \times \mathbf{r}_o$ is zero or parallel to $\hat{\mathbf{z}}$. For $\mathbf{R}_{ot} \times \mathbf{r}_o$ to be parallel to $\hat{\mathbf{z}}$, we should have $\mathbf{r}_o$ perpendicular to $\hat{\mathbf{z}}$ which is not possible with a finite field of view, so only $\mathbf{R}_{ot} \times \mathbf{r}_o = 0$ applies. Consequently, considering that $\mathbf{R}_o$

and $\mathbf{r}_o$ have the same direction, eqn. 3.3 is simplified as

$$\mathbf{R}_{ot} \times \mathbf{R}_o = 0 \tag{3.4}$$

Now substituting for $\mathbf{R}_{ot} = -\mathbf{t} - \boldsymbol{\omega} \times \mathbf{R}_o$, eqn. 2.4, into eqn. 3.4 gives

$$(\boldsymbol{\omega} \times \mathbf{R}_o) \times \mathbf{R}_o + \mathbf{t} \times \mathbf{R}_o = 0. \tag{3.5}$$

Expansion of eqn. 3.5 by using $(\mathbf{a} \times \mathbf{b}) \times \mathbf{c} = (\mathbf{c} \cdot \mathbf{a})\mathbf{b} - (\mathbf{c} \cdot \mathbf{b})\mathbf{a}$ results in

$$(\mathbf{R}_o \cdot \boldsymbol{\omega})\mathbf{R}_o - (\mathbf{R}_o \cdot \mathbf{R}_o)\boldsymbol{\omega} + \mathbf{t} \times \mathbf{R}_o = 0. \tag{3.6}$$

As long as the translational velocity $\mathbf{t}$ is neither zero nor parallel to the interest point vector $\mathbf{R}_o$, then any vector, including $\boldsymbol{\omega}$, can be expressed in terms of the triad of vectors $\mathbf{R}_o$, $\mathbf{t} \times \mathbf{R}_o$ and $\mathbf{t}$. So we can write $\boldsymbol{\omega}$ in its general form as

$$\boldsymbol{\omega} = \alpha\mathbf{R}_o + \beta(\mathbf{t} \times \mathbf{R}_o) + \gamma\mathbf{t} \tag{3.7}$$

where α, β and γ are parameters to be determined. Later in this section we will consider the special cases where $\mathbf{t}$ is zero or parallel to $\mathbf{R}_o$ by defining $\boldsymbol{\omega}$ based on another triad of vectors.

Substituting for $\boldsymbol{\omega}$ from eqn. 3.7 into eqn. 3.6 gives

$$[1 - \beta(\mathbf{R}_o \cdot \mathbf{R}_o)](\mathbf{t} \times \mathbf{R}_o) + \gamma(\mathbf{R}_o \cdot \mathbf{t})\mathbf{R}_o - \gamma(\mathbf{R}_o \cdot \mathbf{R}_o)\mathbf{t} = 0. \tag{3.8}$$

Now, we should find the parameters β and γ such that eqn. 3.8 holds without placing any restrictions on either $\mathbf{R}_o$ or $\mathbf{t}$. We start by finding the dot product of eqn. 3.8 with $\mathbf{t} \times \mathbf{R}_o$ which results in

$$[1 - \beta(\mathbf{R}_o \cdot \mathbf{R}_o)]\|\mathbf{t} \times \mathbf{R}_o\|^2 = 0. \tag{3.9}$$

Equation 3.9 will hold without restricting either $\mathbf{R}_o$ or $\mathbf{t}$ if

$$\beta = \frac{1}{\|\mathbf{R}_o\|^2}. \tag{3.10}$$

Another possibility for satisfying eqn. 3.9 is to have $\|\mathbf{t} \times \mathbf{R}_o\| = 0$ which implies that either $\mathbf{t}$ or $\mathbf{R}_o$ is zero, or $\mathbf{t}$ is parallel to $\mathbf{R}_o$. But $\mathbf{R}_o$ cannot be zero and also we assumed that here $\mathbf{t}$ is neither zero nor parallel to $\mathbf{R}_o$. As a result, $\|\mathbf{t} \times \mathbf{R}_o\|$ cannot be zero.

Similarly the dot product of eqn. 3.8 with $\mathbf{t}$ gives

$$\gamma(\mathbf{R}_o \cdot \mathbf{t})(\mathbf{R}_o \cdot \mathbf{t}) - \gamma(\mathbf{R}_o \cdot \mathbf{R}_o)(\mathbf{t} \cdot \mathbf{t}) = 0. \tag{3.11}$$

Knowing that $(\mathbf{a} \times \mathbf{b}) \cdot (\mathbf{c} \times \mathbf{d}) = (\mathbf{c} \cdot \mathbf{a})(\mathbf{b} \cdot \mathbf{d}) - (\mathbf{b} \cdot \mathbf{c})(\mathbf{d} \cdot \mathbf{a})$, eqn. 3.11 can be simplified as

$$\gamma\|\mathbf{t} \times \mathbf{R}_o\|^2 = 0. \tag{3.12}$$

We discussed that $\|\mathbf{t} \times \mathbf{R}_o\|$ cannot be zero here, so eqn. 3.12 is satisfied only if γ is zero

$$\gamma = 0. \tag{3.13}$$

Substituting for β from eqn. 3.10 and γ from eqn. 3.13 into eqn. 3.7 gives

$$\boldsymbol{\omega} = \alpha\mathbf{R}_o + \frac{1}{\|\mathbf{R}_o\|^2}(\mathbf{t} \times \mathbf{R}_o) \tag{3.14}$$

where α is still unknown. This means that the component of the rotational velocity along $\mathbf{R}_o$ cannot be determined by the *fixation formulation.* Physically this makes sense because the rotational velocity along $\mathbf{R}_o$, denoted by $\omega_{\mathbf{R}_o}$, does not move the fixation point. This observation leads us to find $\omega_{\mathbf{R}_o}$ in a separate step before using the fixation formulation results. Derivation of $\omega_{\mathbf{R}_o}$ will be shown in chapter 4.

As a result, the *fixation constraint equation* (FCE) is written as

$$\omega = \omega_{\mathbf{R}_o}\hat{\mathbf{R}}_o + \frac{1}{\|\mathbf{R}_o\|}(\mathbf{t} \times \hat{\mathbf{R}}_o) \tag{3.15}$$

where $\mathbf{t}$ is the translational velocity and $\hat{\mathbf{R}}_o = \hat{\mathbf{r}}_o$ is the unit vector along the position vector of an arbitrary fixation point, an arbitrary point in the image chosen for fixation. Equation 3.15 shows that after *fixation*, the rotational velocity ω can be explicitly expressed as a linear function of the translational velocity $\mathbf{t}$.

3.1.1 Derivation of special fixation constraint equation

When the translational velocity $\mathbf{t}$ is zero or parallel to the interest point vector $\mathbf{R}_o$, eqn. 3.6 is simplified as

$$(\mathbf{R}_o \cdot \omega)\mathbf{R}_o - (\mathbf{R}_o \cdot \mathbf{R}_o)\omega = 0. \tag{3.16}$$

This time, ω is defined based on the triad consisting of vectors $\mathbf{R}_o$, $\hat{\mathbf{x}}$, and $\hat{\mathbf{x}} \times \mathbf{R}_o$ as

$$\omega = l\mathbf{R}_o + m(\hat{\mathbf{x}} \times \mathbf{R}_o) + n\hat{\mathbf{x}} \tag{3.17}$$

where l, m, and n are parameters to be determined. Here we assume that $\mathbf{R}_o$ is not parallel to $\hat{\mathbf{x}}$. This is a reasonable assumption because otherwise we should at least have a field of view of 180° to be able to choose an awkward interest point along the X-axis, which results in a fixation point at an infinite distance from the principal point and near the border of an infinite image plane.

Substituting for ω from eqn. 3.17 into eqn. 3.16 gives

$$n(\mathbf{R}_o \cdot \hat{\mathbf{x}})\mathbf{R}_o - m(\mathbf{R}_o \cdot \mathbf{R}_o)(\hat{\mathbf{x}} \times \mathbf{R}_o) - n(\mathbf{R}_o \cdot \mathbf{R}_o)\hat{\mathbf{x}} = 0. \tag{3.18}$$

The dot product of eqn. 3.18 with $(\hat{\mathbf{x}} \times \mathbf{R}_o)$ results in

$$-m(\mathbf{R}_o \cdot \mathbf{R}_o)\|\hat{\mathbf{x}} \times \mathbf{R}_o\|^2 = 0. \tag{3.19}$$

Considering that $\mathbf{R}_o$ cannot be either zero or parallel to $\hat{\mathbf{x}}$, eqn. 3.19 is satisfied only if m is zero

$$m = 0. \tag{3.20}$$

Substituting for m into eqn. 3.18 and finding its dot product by $\hat{\mathbf{x}}$ results in

$$n(\mathbf{R}_o \cdot \hat{\mathbf{x}})(\mathbf{R}_o \cdot \hat{\mathbf{x}}) - n(\mathbf{R}_o \cdot \mathbf{R}_o)(\hat{\mathbf{x}} \cdot \hat{\mathbf{x}}) = 0. \tag{3.21}$$

Using $(\mathbf{a} \times \mathbf{b}) \cdot (\mathbf{c} \times \mathbf{d}) = (\mathbf{c} \cdot \mathbf{a})(\mathbf{b} \cdot \mathbf{d}) - (\mathbf{b} \cdot \mathbf{c})(\mathbf{d} \cdot \mathbf{a})$, eqn. 3.21 can be written as

$$n\|\hat{\mathbf{x}} \times \mathbf{R}_o\|^2 = 0. \tag{3.22}$$

Again $\mathbf{R}_o$ cannot be either zero or parallel to $\hat{\mathbf{x}}$. As a result, eqn. 3.22 will hold for arbitrary $\mathbf{R}_o$ if $n = 0$. Substituting for n and m into eqn. 3.17 gives

$$\omega = l\mathbf{R}_o \tag{3.23}$$

where l is still unknown. We can substitute $\omega_{\mathbf{R}_o}\hat{\mathbf{R}}_o$ for $l\mathbf{R}_o$. The procedure for computing the component of rotational velocity along the fixation axis, $\omega_{\mathbf{R}_o}$, will be given in chapter 4. Consequently, for the special cases we obtain the *special fixation constraint equation* (SFCE) as

$$\omega = \omega_{\mathbf{R}_o}\hat{\mathbf{R}}_o \tag{3.24}$$

which means that when the translational velocity $\mathbf{t}$ is zero or parallel to $\mathbf{R}_o$ then the corresponding rotational velocity may only have a component along $\mathbf{R}_o$.

This procedure for deriving the SFCE, eqn. 3.24, is not essentially different from what we did for deriving the FCE, eqn. 3.15. In fact, eqn. 3.24 is a special case of

eqn. 3.15. But we did not directly derive eqn. 3.24 from eqn. 3.15 because eqn. 3.15 was derived based on the assumption that $\mathbf{t}$ is neither zero nor parallel to $\mathbf{R}_o$. As a result, for implementation it is enough to use the FCE, eqn. 3.15, without knowing whether the present condition is a special case or not.

3.1.2 Interpretation of the FCE

We gave a detailed mathematical proof for derivation of the *fixation constraint equation* (FCE), eqn. 3.15. This constraint equation indicates that for a sequence of fixated images, the *rotational velocity* $\boldsymbol{\omega}$ can always be expressed as a linear function of the *translational velocity* $\mathbf{t}$. This section examines whether the FCE makes sense phsically.

The first term $\omega_{\mathbf{R}_o}\hat{\mathbf{R}}_o$ says that $\boldsymbol{\omega}$ can have an unrestricted component along the fixation axis $\hat{\mathbf{R}}_o$. This is correct because such a component does not cause the fixation point to move and as a result the fixation is not violated.

The term of the FCE, $\frac{1}{\|\mathbf{R}_o\|}(\mathbf{t} \times \hat{\mathbf{R}}_o)$, conveys two points:

- The translation $\mathbf{t}$ can have an arbitrary component along the fixation axis $\hat{\mathbf{R}}_o$ because such a component does not move the fixation point in the image plane.
- The rotational velocity $\boldsymbol{\omega}$ should have a component perpendicular to $\hat{\mathbf{R}}_o$ and be large enough to compensate for the component of the translational velocity $\mathbf{t}$ which is perpendicular to $\hat{\mathbf{R}}_o$ in order to keep the fixation stationary in the image plane.

We can conclude that the FCE has a meaningful physical interpretation.

3.2 Solving the General Direct Motion Vision Problem

At this stage, we assume that a sequence of two fixated images have been constructed. In other words, we have made the fixation point stationary in the image plane. This can be done first by finding the *fixation velocity*, the apparent velocity at the fixation

point in the 1st image, as shown in chapter 4. Then the *pixel shifting process* explained in chapter 5 can be used for constructing a new image, the *2nd fixated image*, in which the image of the interest point is positioned at the same point as in the 1st initial image.

We start by studying the general case where the translational velocity $\mathbf{t}$ is neither zero nor parallel to the interest point vector $\mathbf{R}_o$. The special cases of $\mathbf{t}$ will be discussed later.

Substituting for ω from the fixation constraint equation 3.15 into the brightness-change constraint equation 2.8 gives

$$E_t + \omega_{\mathbf{R}_o}\mathbf{v}\cdot\hat{\mathbf{R}}_o + \frac{1}{\|\mathbf{R}_o\|}[\mathbf{v}\cdot(\mathbf{t}\times\hat{\mathbf{R}}_o)] + \frac{1}{Z}(\mathbf{s}\cdot\mathbf{t}) = 0. \tag{3.25}$$

Knowing that $\mathbf{a}\cdot(\mathbf{b}\times\mathbf{c}) = (\mathbf{a}\times\mathbf{b})\cdot\mathbf{c}$ and doing some manipulations on eqn. 3.25 results in

$$E'_t + [\frac{1}{Z}\mathbf{s} - \frac{1}{\|\mathbf{R}_o\|}(\mathbf{v}\times\hat{\mathbf{R}}_o)]\cdot\mathbf{t} = 0 \tag{3.26}$$

where E'_t is a notation for $E_t + \omega_{\mathbf{R}_o}\mathbf{v}\cdot\hat{\mathbf{R}}_o$ which is computable at any pixel assuming that $\omega_{\mathbf{R}_o}$ is known. In chapter 4, we will introduce a technique which finds a good estimate for $\omega_{\mathbf{R}_o}$.

In general, eqn. 3.26 can be solved numerically for $\mathbf{t}$ and Z using images of any size and with any field of view. For a small patch around the fixation point, called a *fixation patch*, eqn. 3.26 can be simplified as

$$E'_t + (\frac{1}{Z} - \frac{1}{Z_o})(\mathbf{s}\cdot\mathbf{t}) \approx 0.^1 \tag{3.27}$$

[1]Considering that $\|\mathbf{R}_o\| = Z_o\|\mathbf{r}_o\|$ and $\mathbf{v} = \mathbf{r}\times\mathbf{s}$, the term $\frac{1}{\|\mathbf{R}_o\|}(\mathbf{v}\times\hat{\mathbf{R}}_o)$ from eqn. 3.26, let's call it K, can be expanded as

$$K = \frac{1}{Z_o\|\mathbf{r}_o\|}(\mathbf{r}\times\mathbf{s})\times\frac{\mathbf{r}_o}{\|\mathbf{r}_o\|}.$$

Further expansion of K by using the relation $(\mathbf{a}\times\mathbf{b})\times\mathbf{c} = (\mathbf{c}\cdot\mathbf{a})\mathbf{b} - (\mathbf{c}\cdot\mathbf{b})\mathbf{a}$, results in

$$K = \frac{1}{Z_o\|\mathbf{r}_o\|^2}[(\mathbf{r}_o\cdot\mathbf{r})\mathbf{s} - (\mathbf{r}_o\cdot\mathbf{s})\mathbf{r}].$$

As described in the footnote, the approximation made here is based on a purely geometric assumption and is not related to the image properties. For example, we are not making any assumptions about the depth topology. We simply assume that motion parameters can be obtained using a small *fixation patch.* As shown in fig. 3-3, the smallness of such a patch translates into the smallness of an angle α. Numerous

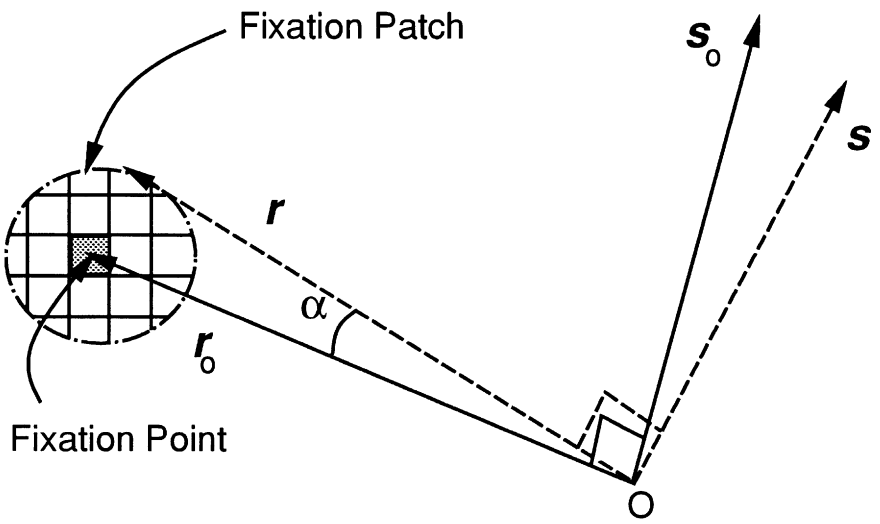

Figure 3-3: A schematic interpretation of *fixation point* and *fixated image sequence.*

experimental results in chapter 9 show that indeed good motion estimates are obtained using *optimum patch* sizes with a field of view small enough to justify this assumption.

In analogy to the pure translation case of [31], we can find the translational velocity $\mathbf{t}$. Equation 3.27 shows that $1/(\frac{1}{Z} - \frac{1}{Z_o}) = -\frac{\mathbf{s}\cdot\mathbf{t}}{E'_t}$. At the points where E'_t is very small, even a small error in computing $\mathbf{t}$ will result in large error in $1/(\frac{1}{Z} - \frac{1}{Z_o})$ which translates into large error in the estimation of depth Z. Considering this fact, the true translational velocity $\mathbf{t}$ can be found from eqn. 3.27 by minimizing

$$J = \iint (\frac{1}{Z} - \frac{1}{Z_o})^{-2} dxdy = \iint (\frac{\mathbf{s}\cdot\mathbf{t}}{E'_t})^2 dxdy \tag{3.28}$$

It is clear that at the fixation point, where $\mathbf{r} = \mathbf{r}_o$ and $\mathbf{s} = \mathbf{s}_o$, $K = \frac{1}{Z_o}\mathbf{s}_o$ and for the points near the fixation point $K \approx \frac{1}{Z_o}\mathbf{s}$.

with respect to $\mathbf{t}$. In other words, we are looking for the true motion $\mathbf{t}$ which minimizes the sum of squares of $\frac{\mathbf{s}\cdot\mathbf{t}}{E'_t}$ over the fixation patch. Note that this minimization does not force Z towards Z_0 because at $Z = Z_0$ the value of J becomes infinite.

We also put the $\|\mathbf{t}\| = 1$ constraint on this minimization problem to avoid the trivial solution $\mathbf{t} = 0$. This is a valid constraint on $\mathbf{t}$ because due to the *scale factor ambiguity* we can only find the direction of $\mathbf{t}$. This constraint on $\mathbf{t}$ can be written as

$$\mathbf{t}^T\mathbf{t} = 1. \tag{3.29}$$

Moreover we can rewrite J as

$$J = \mathbf{t}^T M \mathbf{t} \tag{3.30}$$

where M is a fully computable 3×3 symmetric matrix

$$M = \int\int (\frac{1}{E'_t})^2 \mathbf{s}\mathbf{s}^T dxdy. \tag{3.31}$$

Minimizing J in eqn. 3.30 under the constraint eqn. 3.29 is an ordinary calculus constrained minimization problem which can be solved by minimizing

$$I(\mathbf{t},\ \lambda) = \mathbf{t}^T M \mathbf{t} + \lambda(1 - \mathbf{t}^T\mathbf{t}) \tag{3.32}$$

with respect to $\mathbf{t}$ and the Lagrange multiplier λ. Then, we will obtain

$$\frac{\partial I}{\partial t} = 2M\mathbf{t} - 2\lambda\mathbf{t} = 0 \tag{3.33}$$

which is simplified as

$$M\mathbf{t} = \lambda\mathbf{t}. \tag{3.34}$$

Equation 3.34 is an eigenvalue problem where λ is an eigenvalue of the known matrix M and $\mathbf{t}$ is the corresponding eigenvector. The eigenvalues of M are real and nonnegative because M is a positive semidefinite Hermitian matrix. Substituting for $M\mathbf{t}$ from

eqn. 3.34 into eqn. 3.32 gives $I = \lambda$ which implies that under the given constraint, $\mathbf{t}^T M \mathbf{t}$ is minimized when the smallest of three real and nonnegative eigenvalues is used for computing the eigenvector $\mathbf{t}$.

It is shown that the fixation method can be used for solving the motion vision problem in its general case. The translational velocity $\mathbf{t}$ is obtained from eqn. 3.34 by using the smallest eigenvalue and computing its corresponding eigenvector. Then we can use eqn. 3.26 for finding the depth map, a depth at each image point, as

$$Z = \frac{(\mathbf{s} \cdot \mathbf{t})}{\frac{(\mathbf{v} \times \hat{\mathbf{R}}_o) \cdot \mathbf{t}}{\|\mathbf{R}_o\|} - E'_t}. \tag{3.35}$$

Then, eqn. 3.15 gives the partial rotational velocity ω

$$\omega = \omega_{\mathbf{R}_o} \hat{\mathbf{R}}_o + \frac{1}{\|\mathbf{R}_o\|} (\mathbf{t} \times \hat{\mathbf{R}}_o) \tag{3.36}$$

where $\|\mathbf{R}_o\| = Z_o \|\mathbf{r}_o\|$ and Z_o is the depth at the fixation point. Appendix C introduces a technique for estimating Z_o.

The total rotational velocity of the observer relative to the environment is obtained by adding ω to the *equivalent rotational velocity* Ω given in chapter 5. It can be seen that for the general case, the *fixation formulation* lets us find the shape and motion by choosing virtually any point as the fixation point.

3.2.1 Special cases: t is zero or parallel to $\mathbf{R}_o$

When the translational velocity $\mathbf{t}$ is zero, we showed that the partial rotational velocity ω has only a component about the fixation axis $\mathbf{R}_o$, eqn. 3.24. The technique for computing this component of rotational velocity is given in chapter 4. For this special case, pure rotation, there are also methods for finding the total rotational velocity using the initial unfixated images [31]. In the case of $\mathbf{t} = 0$, we basically cannot obtain any estimation for the depth Z.

For the other special case that $\mathbf{t}$ is parallel to $\mathbf{R}_\circ$, we substitute for ω from eqn. 3.24 into the BCCE eqn. 2.8 to obtain

$$E'_t + \frac{1}{Z}(\mathbf{s} \cdot \mathbf{t}) = 0 \tag{3.37}$$

where E'_t is again a notation for the computable term $E_t + \omega_{\mathbf{R}_\circ}\mathbf{v} \cdot \hat{\mathbf{R}}_\circ$. Because no approximation is involved in deriving eqn. 3.37, an exact closed form solution exists for $\mathbf{t}$ and Z without any restriction on the field of view or the size of fixation patch. This exact solution for finding $\mathbf{t}$ and Z is the same as the solution given in the general case, starting from eqn. 3.28, except that J is defined as $\iint Z^2 dxdy$ for this special case.

For the other special case that t is parallel to $\vec{R}_o$, we substitute for ω from eqn. 3.24 into the BCCE eqn. 2.8 to obtain

$$E'_t + \frac{1}{Z}(a \cdot t) = 0 \tag{3.37}$$

where E'_t is again a notation for the computable term $E_t + \omega_p \nu \cdot \vec{R}_o$. Because no approximation is involved in deriving eqn. 3.37, an exact closed form solution exists for t and Z without any restriction on the field of view or the size of fixation patch. This exact solution for finding t and Z is the same as the solution given in the general case, starting from eqn. 3.36, except that λ is defined as $\iint Z^2 dxdy$ for this special case.

Computing the Fixation Velocity and Rotational Component $\omega_{\mathbf{R}_o}$

Chapter 4

In an arbitrary image sequence, a point chosen as the *fixation point* does not necessarily stay stationary in the image plane. We use the term *fixation velocity* to refer to the apparent velocity at the fixation point in the initial 1st image. As shown in fig. 4-1, the x and y components of the fixation velocity are represented by u_o and v_o respectively.

The fixation method requires a sequence of two fixated images in which the fixation point stays stationary, $\mathbf{r}_{ot} = 0$. A fixated image sequence can be obtained by first finding u_o and v_o, and then using these components to construct a new image, the *fixated 2nd image*. The technique for the construction of the fixated 2nd image (*pixel shifting process*) is explained in chapter 5.

We also saw that the component of the rotational velocity along the fixation axis, $\omega_{\mathbf{R}_o}$, cannot be obtained from the fixation formulation because this component does not move the fixation point.

In this chapter, we will introduce an algorithm for obtaining not only the rotation $\omega_{\mathbf{R}_o}$ but also the components of the fixation velocity, u_o and v_o.

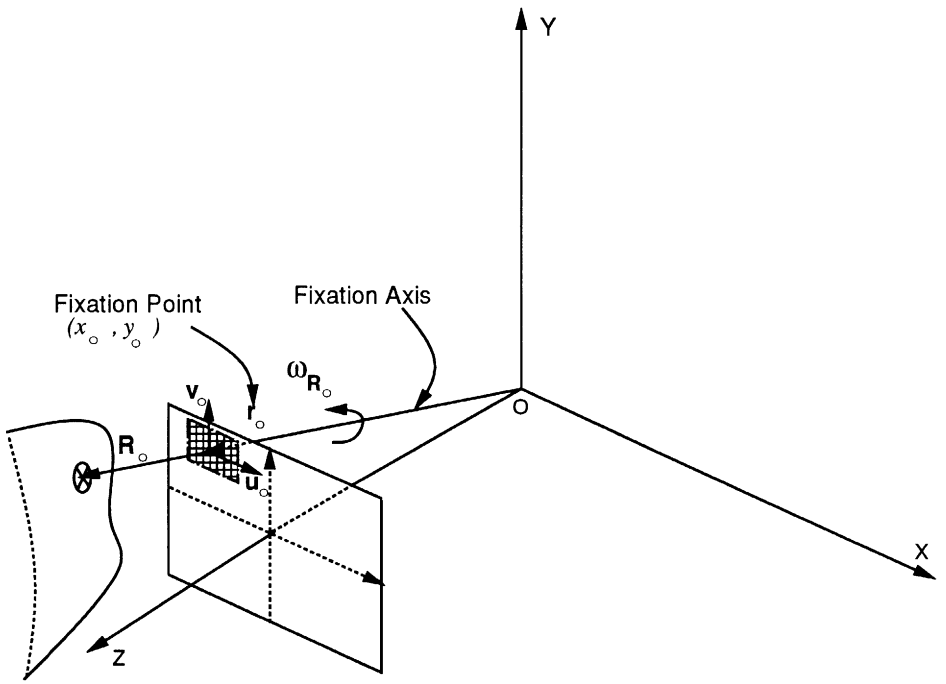

Figure 4-1: In general, for any point chosen as the *fixation point*, there is an associated apparent velocity (*fixation velocity*), and a rotational component along the *fixation axis*, $\omega_{\mathbf{R}_o}$. The components of *fixation velocity* are shown by (u_o, v_o).

4.1 Algorithm

The motion field velocity due to the rotational velocity component $\omega_{\mathbf{R}_o}$ is given by $-(\omega_{\mathbf{R}_o} \times \mathbf{r}) = -\omega_{\mathbf{R}_o}(\hat{\mathbf{R}}_o \times \mathbf{r}) = -\frac{\omega_{\mathbf{R}_o}}{\|\mathbf{r}_o\|}(\mathbf{r}_o \times \mathbf{r})$, where $\hat{\mathbf{R}}_o = \hat{\mathbf{r}}_o$ is the unit vector along the fixation axis $\mathbf{r}_o$. Considering a small patch around the fixation point, and substituting $\mathbf{r}_o = (x_o\ y_o\ 1)^T$ and $\mathbf{r} = (x\ y\ 1)^T$, the components of the total motion field velocity due to the fixation velocity and $\omega_{\mathbf{R}_o}$, are given by

$$\begin{cases} x_t &= u_o - \frac{\omega_{\mathbf{R}_o}}{\|\mathbf{r}_o\|}\hat{\mathbf{x}} \cdot (\mathbf{r}_o \times \mathbf{r}) &= u_o + \bar{\omega}_{\mathbf{R}_o}(y - y_o) \\ y_t &= v_o - \frac{\omega_{\mathbf{R}_o}}{\|\mathbf{r}_o\|}\hat{\mathbf{y}} \cdot (\mathbf{r}_o \times \mathbf{r}) &= v_o - \bar{\omega}_{\mathbf{R}_o}(x - x_o) \end{cases} \tag{4.1}$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ are the unit vectors along the x and y axes and $\bar{\omega}_{\mathbf{R}_o}$ is a notation for $\frac{\omega_{\mathbf{R}_o}}{\|\mathbf{r}_o\|}$. Substituting for x_t and y_t from the above equations into the BCCE, eqn. 2.7, gives

$$[u_o + \bar{\omega}_{\mathbf{R}_o}(y - y_o)]E_x + [v_o - \bar{\omega}_{\mathbf{R}_o}(x - x_o)]E_y + E_t = 0. \tag{4.2}$$

Due to noise, eqn. 4.2 does not necessarily hold for any point (x, y). Thus, we try to find $u_\circ, v_\circ$ and $\bar{\omega}_{\mathbf{R}_\circ}$ by minimizing the sum of squares of errors over the fixation patch. In other words we want to minimize

$$\iint [(u_\circ + \bar{\omega}_{\mathbf{R}_\circ}(y - y_\circ))E_x + (v_\circ - \bar{\omega}_{\mathbf{R}_\circ}(x - x_\circ))E_y + E_t]^2 dx\, dy \tag{4.3}$$

with respect to $u_\circ$, $v_\circ$ and $\bar{\omega}_{\mathbf{R}_\circ}$. This results in a system of three linear equations that can be solved for the three unknowns

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{pmatrix} u_\circ \\ v_\circ \\ \bar{\omega}_{\mathbf{R}_\circ} \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix}. \tag{4.4}$$

Matrix **A** is symmetric and its elements are given by

$$\begin{cases} a_{12} = \iint E_x E_y dx\, dy \\ a_{13} = \iint E_x[E_x(y - y_\circ) - E_y(x - x_\circ)]dx\, dy \\ a_{23} = \iint E_y[E_x(y - y_\circ) - E_y(x - x_\circ)]dx\, dy \\ a_{11} = \iint E_x^2 dx\, dy \\ a_{22} = \iint E_y^2 dx\, dy \\ a_{33} = \iint [E_x(y - y_\circ) - E_y(x - x_\circ)]^2 dx\, dy \end{cases} \tag{4.5}$$

and the components of vector **C** are as follows:

$$\begin{cases} c_1 = -\iint E_t E_x dx\, dy \\ c_2 = -\iint E_t E_y dx\, dy \\ c_3 = -\iint E_t[E_x(y - y_\circ) - E_y(x - x_\circ)]dx\, dy. \end{cases} \tag{4.6}$$

Considering that the fixation point coordinates $x_\circ$ and $y_\circ$ are known, the sets of equations 4.5 and 4.6 show that the elements of matrix **A** and the components of vector **C** are fully computable.

4.2 Discussion

When the spatio-temporal gradients are zero, matrix A is irreversible because all of its elements are zero. As a result, we will not be able to compute the motion components in such a case. Chapter 10 explains how to avoid this by an autonomous choice of an appropriate fixation point that is not located in a patch with uniform brightness. Furthermore, for implementation we make sure that the determinant of matrix A is nonzero before advancing into the computations.

In the special case where the fixation point is at the principal point, $x_o = y_o = 0$, elements of matrix **A** are simplified as

$$\begin{cases} a_{12} &= \iint E_x E_y dx\, dy \\ a_{13} &= \iint E_x(yE_x - xE_y)dx\, dy \\ a_{23} &= \iint E_y(yE_x - xE_y)dx\, dy \\ a_{11} &= \iint E_x^2 dx\, dy \\ a_{22} &= \iint E_y^2 dx\, dy \\ a_{33} &= \iint (yE_x - xE_y)^2 dx\, dy \end{cases} \tag{4.7}$$

and components of vector **C** are given as follows

$$\begin{cases} c_1 &= -\iint E_t E_x dx\, dy \\ c_2 &= -\iint E_t E_y dx\, dy \\ c_3 &= -\iint E_t(yE_x - xE_y)dx\, dy. \end{cases} \tag{4.8}$$

After finding $\bar{\omega}_{\mathbf{R}_o}$, we can easily compute $\omega_{\mathbf{R}_o} = \bar{\omega}_{\mathbf{R}_o}\sqrt{x_o^2 + y_o^2 + 1}$. Clearly, when the fixation point is at the principal point, $\omega_{\mathbf{R}_o}$ becomes equal to $\bar{\omega}_{\mathbf{R}_o}$.

The algorithm given in this chapter has been successfully implemented on real images and good estimates have been obtained for the fixation velocity components and $\omega_{\mathbf{R}_o}$. Chapter 8 describes the implementation results.

Constructing a Sequence of Fixated Images

Chapter 5

The fixation method requires a sequence of two images in which the fixation point is kept stationary. However, the input can be an *arbitrary* sequence of two images that we shall call the *1st initial* and *2nd initial* images. The *1st initial* image is used directly as the *1st fixated* image but we need to find a *2nd fixated* image using the *2nd initial* image.

Physical rotation of the camera relative to the observer base is a hardware solution to this problem which is basically a *tracking* problem. Considering that in general the interest point has a motion relative to the observer, the 2nd fixated image cannot be obtained in one step. As a result, a feedback control loop is required for the camera rotation system to compensate for the errors resulting from the new position of the fixation point. This tracking approach is to be avoided not only because of the potential errors involved but also because of concern about real time applications.

In this chapter, we will show how a 2nd fixated image can be constructed by a purely software technique, the *pixel shifting process.* It involves applying an imaginary rotation to the vision system and determining the corresponding transformation which affects the 2nd initial image.

5.1 Equivalent Rotational Velocity

If point 1 is chosen as the fixation point in the 1st initial image, then in general its corresponding image point in the *2nd initial* image moves to a new location such as point 2; see fig. 5-1.

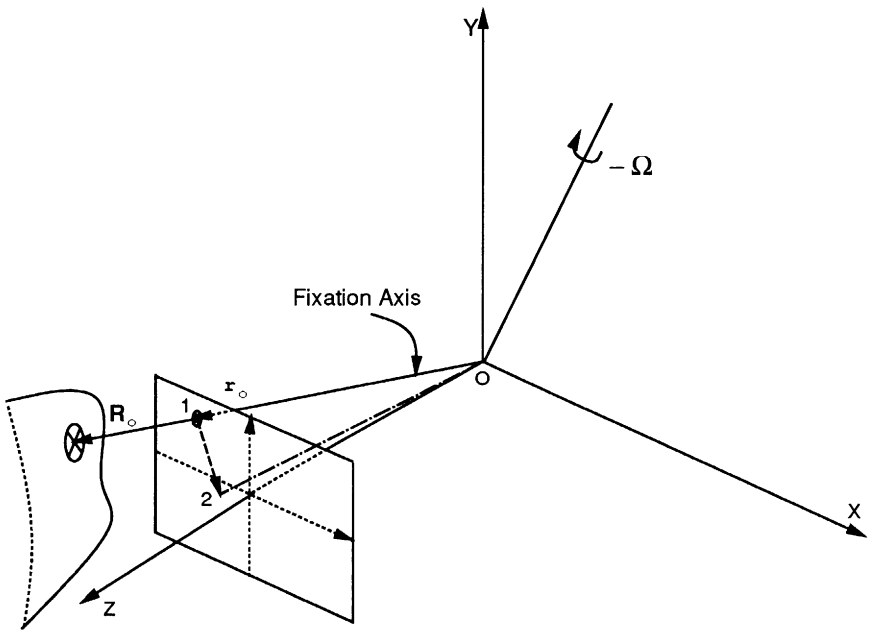

Figure 5-1: An imaginary rotation opposite to the *equivalent rotational velocity*, $-\Omega$, is applied to the vision system to bring point 2 to point 1. This rotation transforms the *2nd initial* image into the *2nd fixated* image.

Determining the location of point 2 is equivalent to the estimation of the fixation velocity. Chapter 4 introduced a technique for the estimation of the fixation velocity. The experimental results in chapters 8 and 9 will also show that the fixation velocity can be estimated reliably even from real and noisy images. As a result, it is assumed here that the fixation velocity has been already computed from eqn. 4.4.

There are infinite combinations of translations and rotations which can be applied to the vision system or camera to bring the image point at 2 to the location 1. Among all these combinations, we choose to accomplish the task by a pure rota-

tion. To find the desired rotation, we first introduce an *equivalent rotational velocity*, $\Omega = (\Omega_x, \Omega_y, \Omega_z)$, as a rotation which can result in the same fixation velocity (u_o, v_o) at the fixation point (x_o, y_o). According to eqn. 2.6, the components of $\mathbf{\Omega}$ must satisfy the following set of equations

$$\begin{cases} u_o = x_o y_o \Omega_x - (x_o^2 + 1)\Omega_y + y_o \Omega_z \\ v_o = (y_o^2 + 1)\Omega_x - x_o y_o \Omega_y - x_o \Omega_z. \end{cases} \tag{5.1}$$

There are also infinite configurations of $\mathbf{\Omega}$ that satisfy the system of equations in 5.1. However, we choose the only one that does not introduce any new rotational velocity along the fixation axis $\mathbf{r}_o$. Mathematically it is equivalent to having $\mathbf{\Omega} \cdot r_o = 0$ which results in an extra constraint on the components of $\mathbf{\Omega}$,

$$x_o \Omega_x + y_o \Omega_y + \Omega_z = 0. \tag{5.2}$$

This constraint guarantees that the value of $\omega_{\mathbf{R}_o}$ obtained by applying the system of equations 4.4 to the two initial images is also valid for the fixated images. As a result, no adjustment in $\omega_{\mathbf{R}_o}$ is needed before using it in equations 3.35 and 3.36 which must be applied to a sequence of fixated images.

Considering that the fixation velocity (u_o, v_o) and the fixation point coordinates x_o and y_o are known here, the equivalent rotational velocity $\mathbf{\Omega}$ is obtained by solving the combination of three linear equations in 5.1 and 5.2. For example, in the case that the fixation point is at the principal point, $x_o = y_o = 0$, the equivalent rotational velocity becomes,

$$\mathbf{\Omega} = (v_o, -u_o, 0). \tag{5.3}$$

However, it should be emphasized that fixation point is not restricted to the principal point and virtually any point can be chosen as the fixation point.

5.2 Constructing the 2nd Fixated Image

After obtaining the equivalent rotational velocity Ω, the task of constructing the 2nd fixated image is equivalent to finding the transformation experienced by the 2nd initial image when the imaginary rotation $-\Omega$ is applied to the vision system.

Considering eqn. 5.1, the following set of equations give the component of the corresponding shifting vector $(u,\ v)$ for any pixel $(x,\ y)$ of the 2nd initial image

$$\begin{cases} u = & -xy\Omega_x + (x^2+1)\Omega_y - y\Omega_z \\ v = & -(y^2+1)\Omega_x + xy\Omega_y + x\Omega_z. \end{cases} \tag{5.4}$$

Here Ω_x, Ω_y and Ω_z are known values. As a result, the *shifting vector* (u, v) can be obtained for every pixel of the 2nd initial image.

Figure 5-2 shows the process of constructing the 2nd fixated image using the 2nd initial image, called the *pixel shifting process*. The brightness at pixel (x, y) of the 2nd

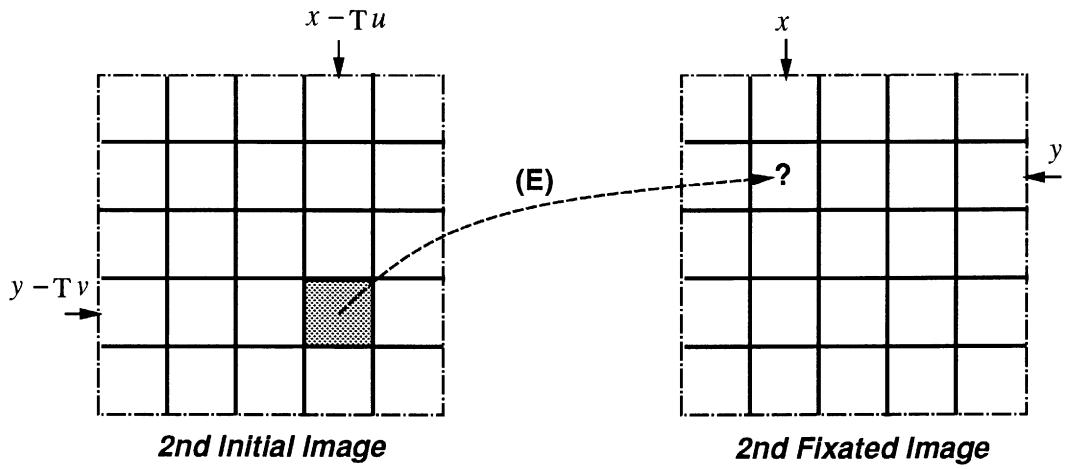

Figure 5-2: The *pixel shifting process* for constructing the *fixated 2nd* image from the *2nd initial* image.

fixated image is the same as the brightness at the corresponding point $(x-Tu, y-Tv)$ in the 2nd initial image, where T is the time interval between two initial images. In general, a computed original point is not located at the center of a pixel in the 2nd

initial image. As a result, its brightness cannot be read directly from the image file and should be computed by averaging, bilinear interpolation or bicubic interpolation of the brightnesses at its neighboring pixels.

It should be clear by now that we neither require the fixated images to be provided in advance nor do we use mechanical tracking for obtaining the fixated images. Construction of the 2nd fixated image is based on the *pixel shifting process.* This is done entirely in software and no tracking is involved in this technique. In chapter 11, we will show the results of implementing this purely software based technique for constructing a sequence from fixated images for several real image sequences.

initial image. As a result, its brightness cannot be read directly from the image file and should be computed by averaging, bilinear interpolation or bicubic interpolation of the brightnesses at its neighboring pixels.

It should be clear by now that we neither require the fixated images to be provided in advance nor do we use mechanical tracking for obtaining the fixated images. Construction of the 2nd fixated image is based on the pixel shifting process. This is done entirely in software and no tracking is involved in this technique. In chapter 11, we will show the results of implementing this purely software based technique for constructing a sequence from fixated images for several real image sequences.

An Overview of the Fixation Method

Chapter 6

The algorithms and formulations presented in the previous chapters show how to solve directly for the motion and shape in the general case. In contrast to previous work done in the area of motion vision, our technique is general and does not put any severe restrictions on the motion or the environment. More importantly, the fixation method uses neither *optical flow* nor *feature correspondence*. Instead, image information such as temporal and spatial brightness gradients are used directly. This method neither requires tracked images as input nor uses tracking for obtaining fixated images. Instead, it introduces a *pixel shifting process* for constructing fixated images at any arbitrary *fixation point*. This process is done entirely in software without moving the camera for tracking.

In the previous chapters, we gave the theory underlying the *fixation method* in detail. This chapter presents a summary of the main steps involved in the *fixation method*.

6.1 Main Modules

Figure 6-1 shows a block diagram of the ideas behind our fixation based motion vision system. Referring to this figure, the fixation method can be implemented in the following steps:

- *Step 1:* Finding the *fixation velocity* components (u_o, v_o) and the component of

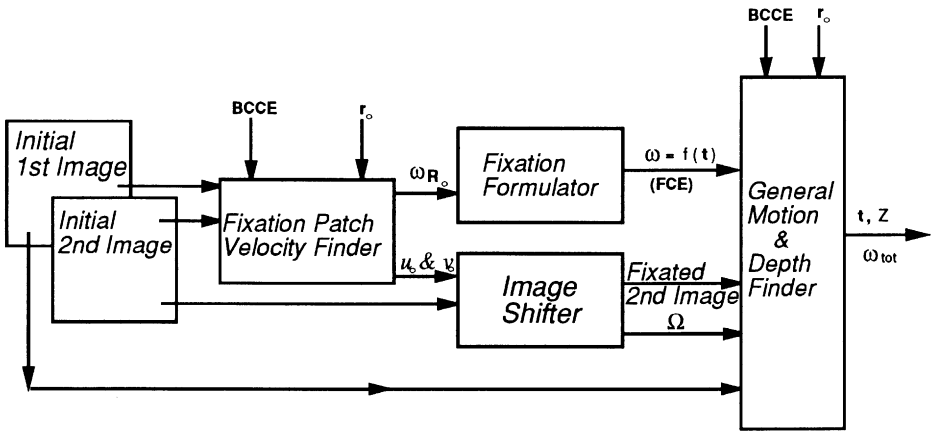

Figure 6-1: The modules of the *fixation* based general motion vision system.

rotational velocity along $\mathbf{R}_o$, $\omega_{\mathbf{R}_o}$, by applying the system of eqn. 4.4 to the brightness gradients from two *initial images.*

• *Step 2:* Knowing the fixation velocity components (u_o, v_o) the *2nd fixated image* is constructed by the *pixel shifting process* explained in chapter 5. This is done entirely in software without any need to move the camera for tracking. This step also results in the estimation of the *equivalent rotational velocity* $\mathbf{\Omega}$.

• *Step 3:* Knowing $\omega_{\mathbf{R}_o}$, and using the fixation constraint equation 3.15, the 1st initial image, and the 2nd fixated image, the method presented in chapter 3 can be used for recovering the *translational velocity* $\mathbf{t}$, the *partial rotational velocity* ω, and the *depth* Z at all image points.

• *Step 4:* The *total rotational velocity* ω_{tot} is obtained simply by adding the *equivalent rotational velocity* $\mathbf{\Omega}$, from equations 5.1 and 5.2, to the *partial rotational velocity* ω from eqn. 3.15.

In the following chapters, we apply our fixation based motion vision system to the real world environment to recover motion and shape in the general case. At every step, we discuss the implementation issues and introduce practical techniques for dealing with them.

Spatial and Temporal Brightness Gradients

Chapter 7

Brightness gradients are the primary source of information for direct method algorithms. Appendix B describes the formulations for obtaining spatial brightness gradients E_x and E_y, and the temporal brightness gradient E_t from a sequence of two time varying images.

This chapter applies those formulations to two real image sequences to obtain the corresponding brightness gradients. Then, we will introduce a technique for the visual representation of the brightness gradients and finally, we will study those representations to explain the significance and characteristics of brightness gradients.

7.1 Visual Representation

Two successive frames of the *landscape* image sequence (taken at the *Imaging Laboratory of Carnegie Mellon University*) are shown in fig. 7-1. These are $8-bit$ images but the last two digits are usually too noisy to be reliable.

The true motion between these frames is a combination of translation and rotation. The real rotation is 0.3 deg about the optical axis Z and the real translation is 2 mm along the horizontal axis X.

Using the formulation in appendix B, we can compute the brightness gradients. The corresponding spatial and temporal brightness gradients for the landscape image sequence are shown in figures 7-2 and 7-3, respectively.

Figure 7-4 shows another image sequence (*cup* image sequence) used in the experiments. The motion between these successive frames is a $3D$ translation of (2.5, 0, 4) mm. The spatial and temporal gradients for the cup image sequence are shown in figures 7-5 and 7-6, respectively.

In these maps, larger gradient values are shown brighter. Such gradient maps suggest a way of visually representing the brightness gradients which renders them more intuitively meaningful.

7.2 Interpretation and Significance

The top gradient maps in figures 7-2, and 7-5 show that horizontal gradients (E_x's) capture the vertical lines and feature in the images. Similarly, the bottom gradient maps in these figures demonstrate that vertical gradients (E_y's) pick up the horizontal lines and feature in the image.

These experimental results show that the spatial gradients capture the geometric and shading characteristics of the images. It is important to notice that the computation behind spatial gradients is very simple. However, they indirectly capture the edges, features, and boundaries of the scene.

The temporal brightness gradient in fig. 7-3 tells us about the motion between two *landscape* images. First of all, the vertical lines and features are seen all over this temporal gradient map. This observation indicates that the motion has a horizontal translation component.

Secondly, there are also horizontal lines in this gradient map but they become weaker as they get close to the left side of the map (this argument becomes more obvious if one compares the horizontal lines in here with those of E_y in fig. 7-2). This means that motion has a rotational component which is centered in the left side of the image. In section 13.2, we will show that this is really the case.

Also, we can observe that at any vertical stripe of the spatial gradient map,

the horizontal lines become stronger as their distance from the center of the stripe increases. This observation indicates that the rotation center is located in the middle of the image.

Figure 7-6 shows that the temporal brightness gradient map captures the vertical edges and features in the cup image sequence. The uniform strength of the vertical lines in fig. 7-6 is an indication of the fact that the motion in the *cup* image sequence is a pure horizontal translation.

7.3 Summary

The gradient maps and discussions presented in this chapter show that the spatial gradients capture the geometric and shading characteristics of the images and the temporal gradients contain important information about the motion.

As shown in appendix B, the computational procedure behind gradient estimation is very simple. In fact, it only involves the subtraction of neighboring pixel values. Such a simple computation indirectly results in capturing the motion and detecting the features, edges, and boundaries in the images.

However, we should emphasize that we neither intended to obtain such edges and features nor did we use such representation of the gradient maps in our algorithms. The intention was to demonstrate that the brightness gradient maps not only contain the motion information (which is usually represented by the optical flow maps) but also have a flavor of features and edges (used in edge maps and feature correspondence algorithms).

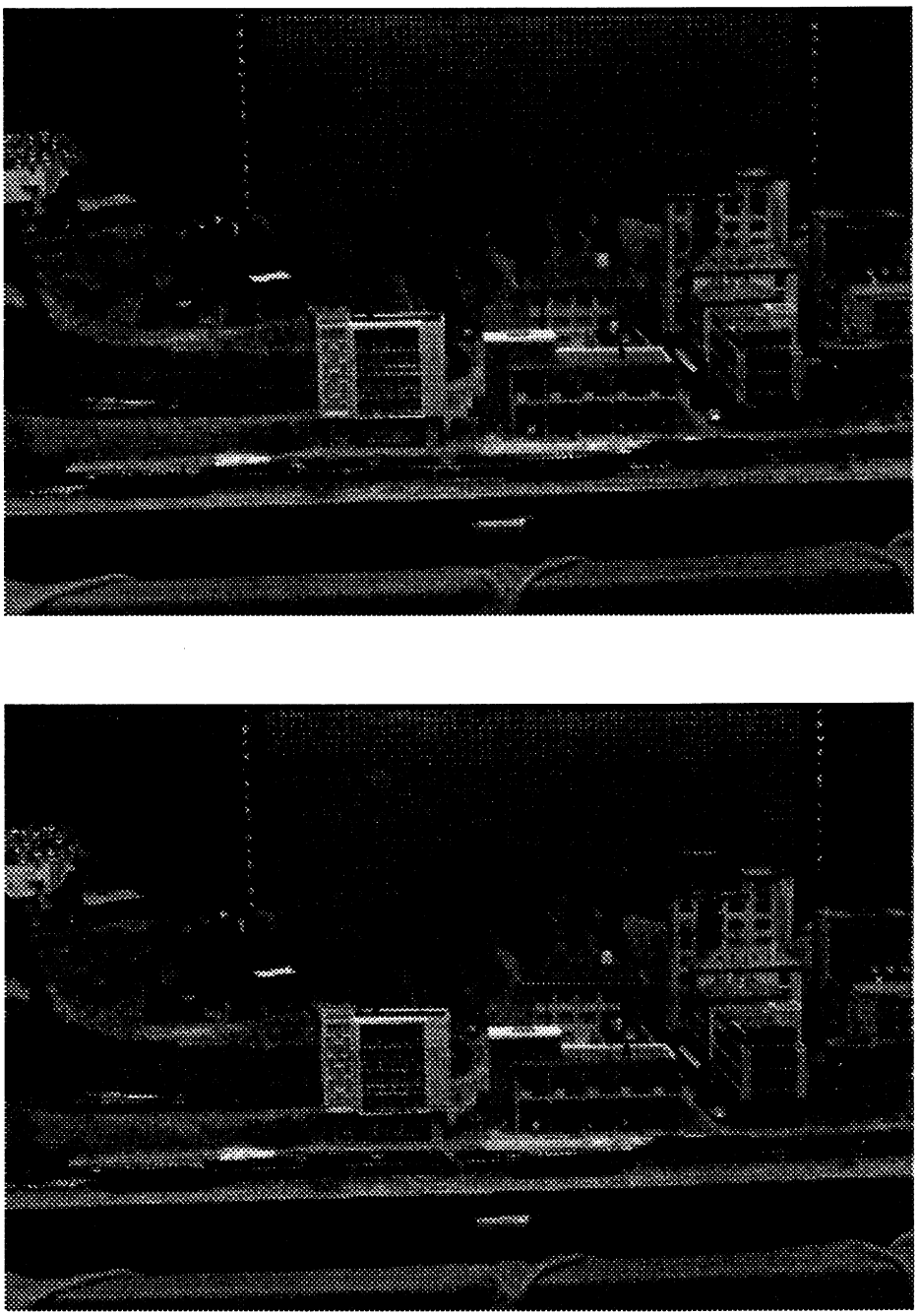

Figure 7-1: The first and second frames in the landscape image sequence. The true motion is a 0.3 deg rotation about the nominal optical axis Z, and a 2 mm translation along the horizontal axis X.

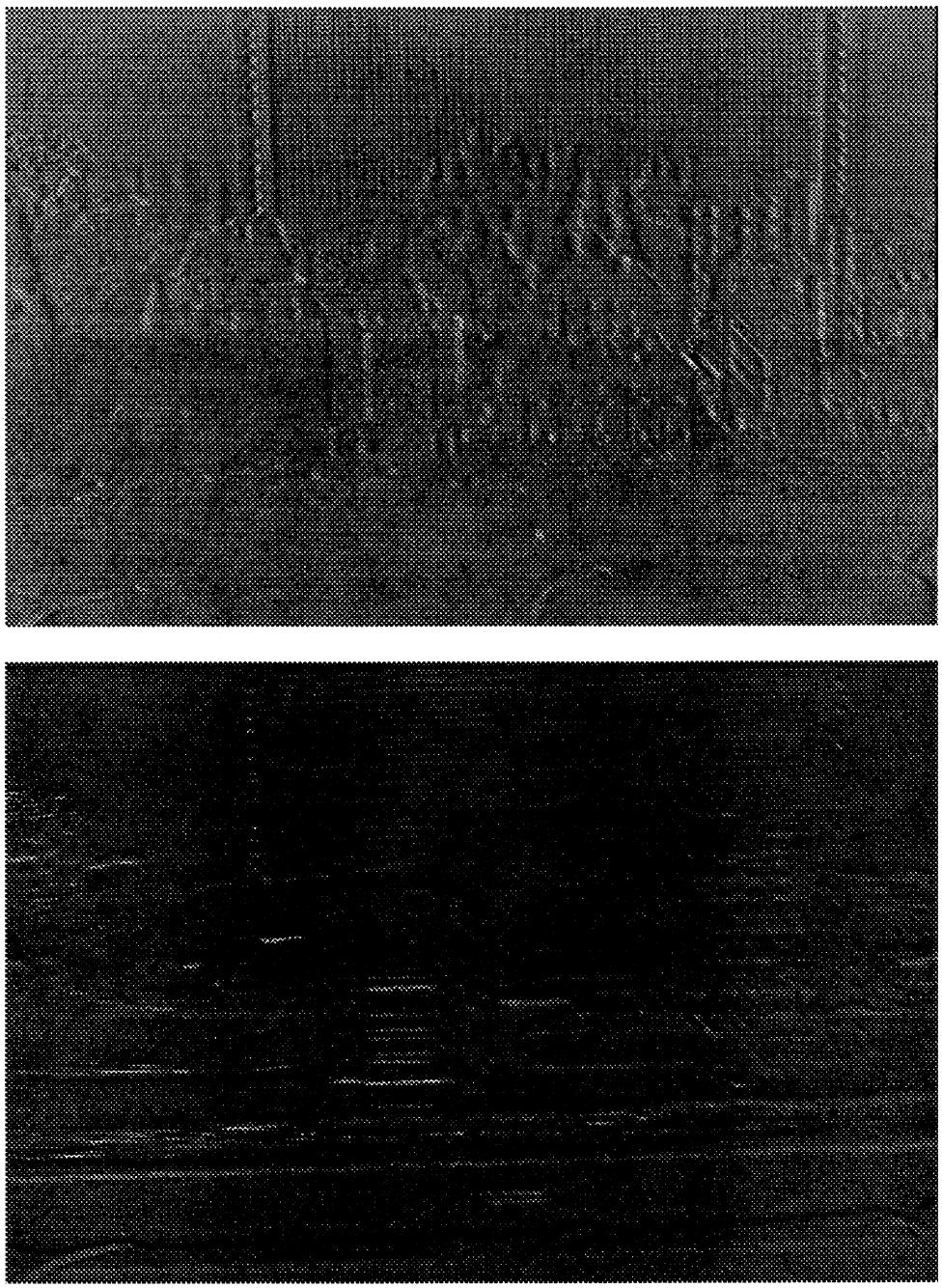

Figure 7-2: The visual representation of the spatial brightness gradients for the landscape image sequence in the horizontal direction (top) and vertical direction (bottom), E_x and E_y. The horizontal gradient map (top) has captured the vertical edges and features in the image. Similarly, the vertical gradient map (bottom) has picked up the horizontal edges and features.

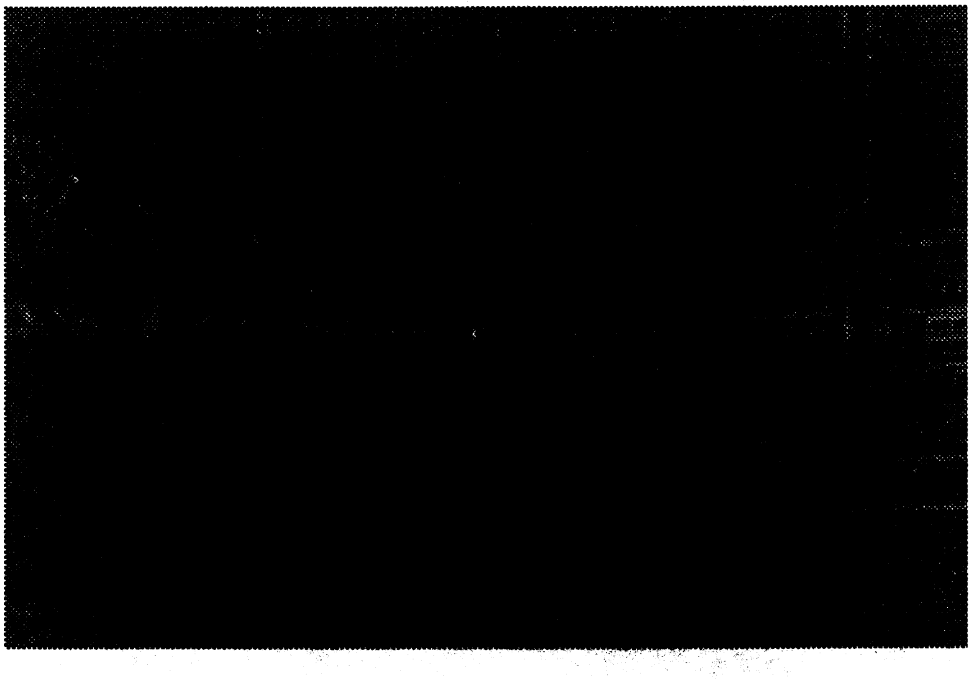

Figure 7-3: The visual representation of the temporal brightness gradient for the landscape image sequence, E_t. The vertical edges with relatively uniform strength suggest that motion has a horizontal translation component. The horizontal edges with decreasing strength towards left indicate that there is also a rotation centered at the left of the image center.

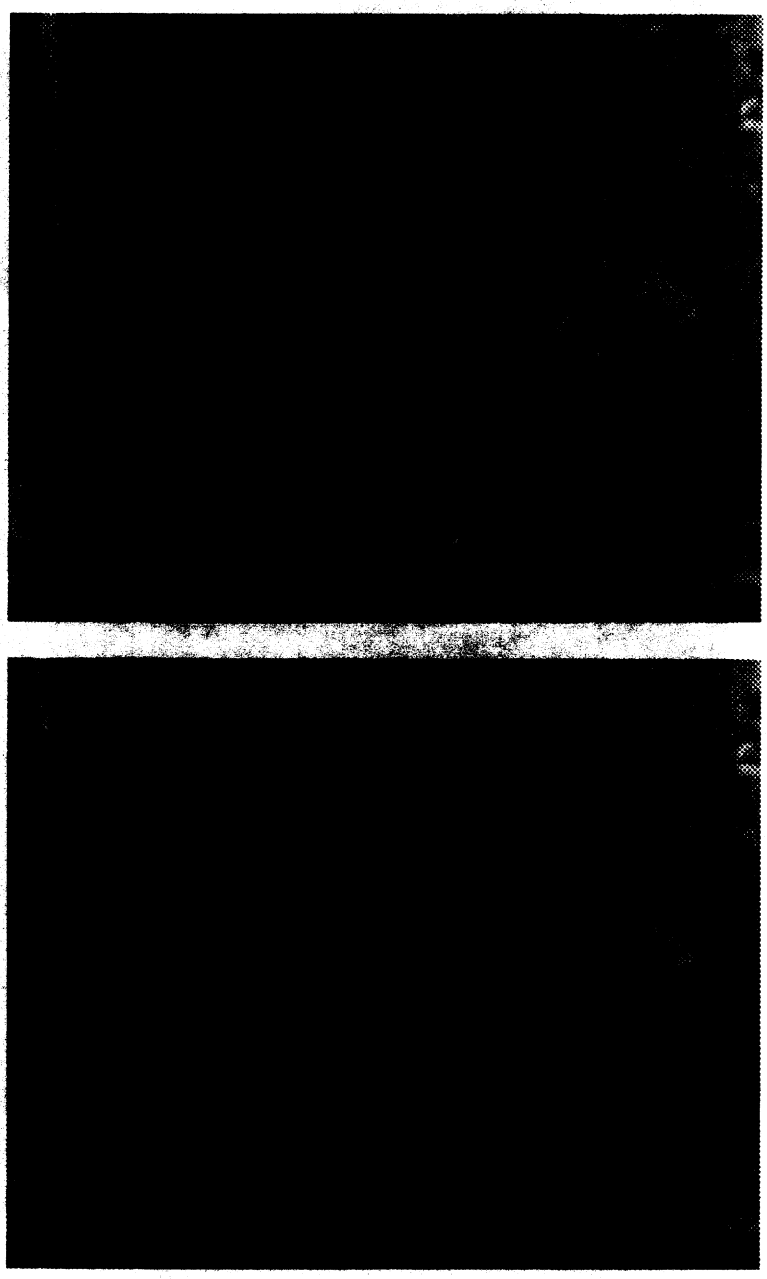

Figure 7-4: The first and second images in the cup image sequence. The true motion between these frames is a $3D$ translation of (2.5, 0, 4) mm.

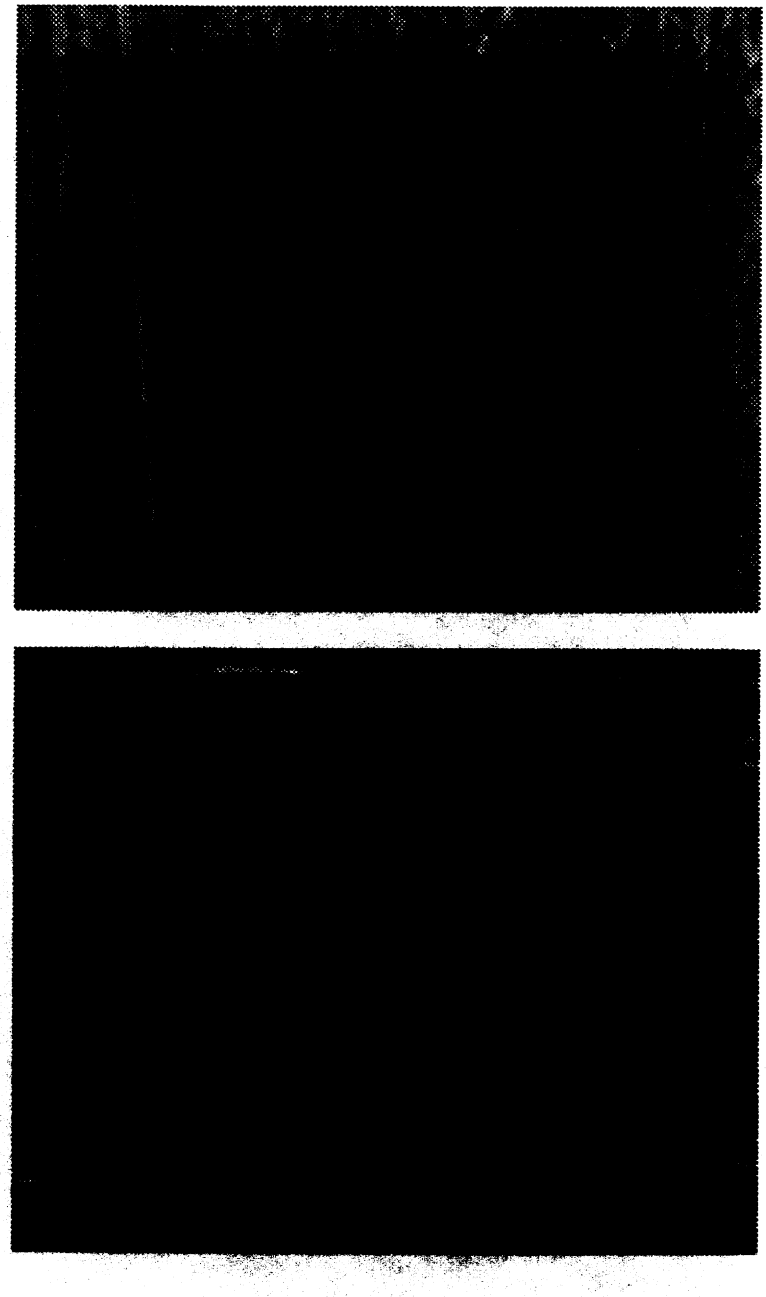

Figure 7-5: The visual representation of the spatial brightness gradients for the cup images in the horizontal direction (top) and vertical direction (bottom), E_x and E_y. The horizontal gradient map (top) has captured the vertical edges and features in the image. Similarly, the vertical gradient map (bottom) has picked up the horizontal edges and features.

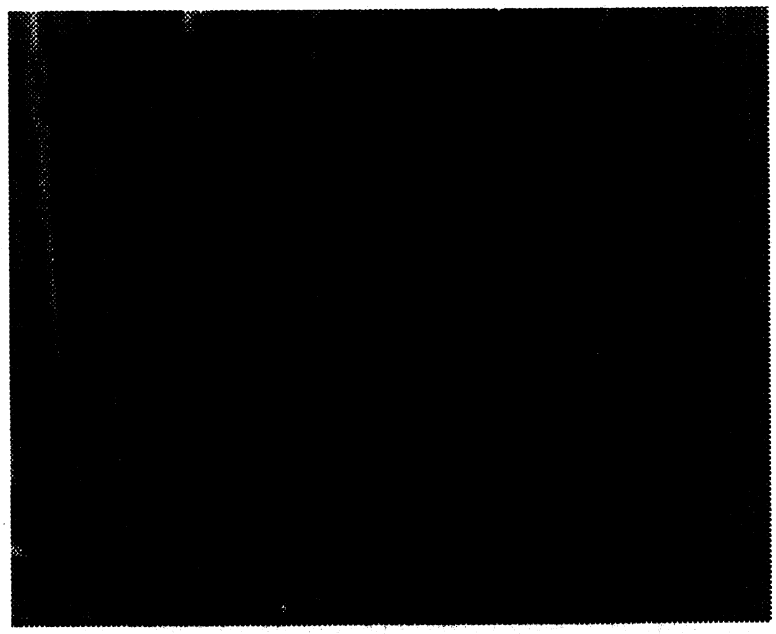

Figure 7-6: The visual representation of the temporal brightness gradient for the cup image sequence, E_t. The presence of relatively uniform vertical edges and features here indicates that the motion is predominantly a horizontal translation

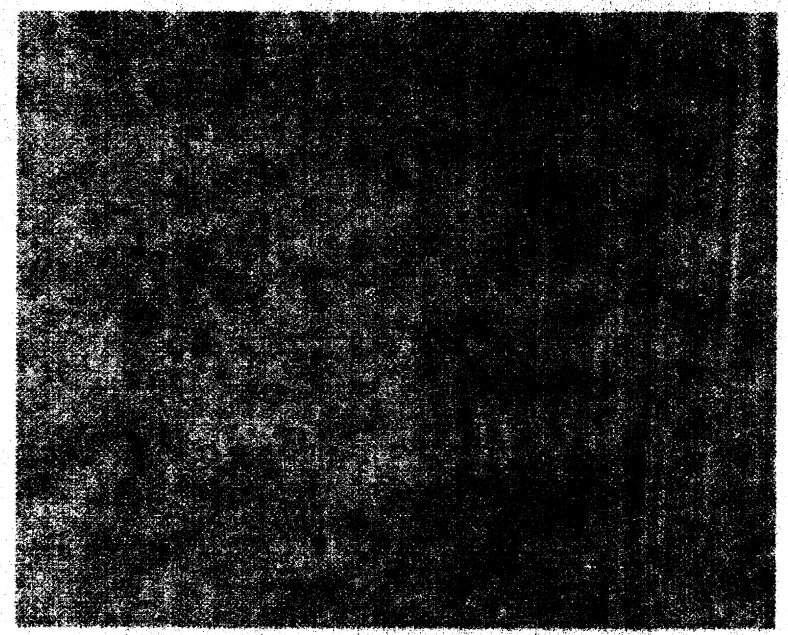

Figure 7-6: The visual representation of the temporal brightness gradient for the cup image sequence, E_3. The presence of relatively uniform vertical edges and features here indicates that the motion is predominantly a horizontal translation

The Effect of Fixation Patch Size

Chapter 8

Finding the fixation velocity (velocity at the fixation point), and the component of rotational velocity about the fixation axis, $\omega_{\mathbf{R}_o}$, is the most important part of our fixation based method for recovering the shape and motion from an arbitrary sequence of input images. This is because in our method a *pixel shifting process* uses the fixation velocity to construct a sequence of fixated images from an arbitrary sequences of input images (chapter 5). We also need $\omega_{\mathbf{R}_o}$ for computing the total rotational velocity (chapter 3).

In chapter 4 we introduced the algorithms for recovering the fixation velocity and $\omega_{\mathbf{R}_o}$ using the information from the fixation patch (an image patch around the fixation point). In this chapter, we study the effect of the fixation patch size on the estimation of the desired motion parameters using two different sequence of images where the motion is a combination of translation and rotation.

8.1 Images with Moderate Relative Depth Changes

Here, we have used a sequence of real images acquired at the *Imaging Laboratory of Carnegie Mellon University.* Figure 7-1 shows two of these 576×384 pixels images. The relative depth is moderate (1250 mm to 1625 mm, about 30% change) in the image portion used in our computations. The camera has a nominal focal length of 24 mm, and a pixel size of 0.02×0.02 mm. The calibrated *principal point* has been used as a fixation point. The calibration technique is explained in section 13.1.

In a raster format system (origin at the top left corner of the image), the calibrated principal point is located near the center of image, pixel (275, 205). The frontal depth of this point is about 1450 *mm*.

The real motion between these two images has both translational and rotational components. The real rotation is -0.3 deg about the optical axis Z and the real translation is -2 *mm* along the horizontal axis X. Testing our algorithms using such real images is valuable because the observed motion is relatively large (more than subpixel motion in the image plane). For very large motions it is enough to use higher frame grabbing rates. These days, there are commercially available frame grabbers which are capable of capturing up to 7,500 frame per second at 12 bits gray scale resolution on personal computers [82].

Using the algorithm described in chapter 4 we can find the horizontal and vertical translations and the rotational component $\omega_{\mathbf{R}_o}$ for any given fixation patch size. The corresponding plots are shown in figures 8-1, 8-2 and 8-3. It is evident that these estimations strongly depend on the fixation patch size especially when the fixation patch is small. Figure 8-1 shows that the horizontal translation converges to its real value (-2 *mm*). On the other hand, the vertical translation (fig. 8-2) converges to 0.9 *mm* which is not its true value. The reason for this disparity is described in section 13.2.

Figure 8-3 shows that for small patch sizes (less than 30×30 pixels in this case) the estimated value for $\omega_{\mathbf{R}_o}$ oscillates wildly and results in unacceptable values. As the patch size increases, the estimated $\omega_{\mathbf{R}_o}$ converges towards the real value of rotation. For large patch sizes (around 100×100 pixels in this case) the estimated rotation, -0.309 deg, becomes roughly the same as the real rotation, -0.3 deg.

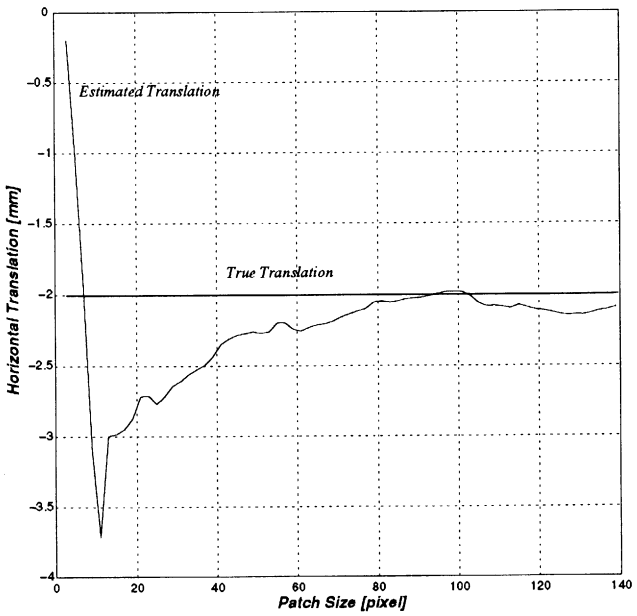

Figure 8-1: The estimated value for the horizontal translation versus the fixation patch size for the landscape image sequence. The true horizontal translation is -2 mm.

8.2 Images with Significant Relative Depth Changes

In this section we will study another image sequence (cup images) which have considerable relative depth changes within the fixation patch (584 mm to 914 mm, about 60% difference). Figure 7-4 shows two of these 227×280 pixels images (cup images).

The real motion of the viewer is a horizontal translation of 2.5 mm to the right. The camera has a nominal focal length of 18.66 mm, pixel-width of 0.032 mm, and pixel-height of 0.029 mm. We have used the nominal principal point (image center) as our fixation point.

Figure 8-4, shows the estimates for the horizontal translation, vertical translation, and the rotational velocity component $\omega_{\mathbf{R}_o}$. It is obvious that the estimated values depend strongly on the size of the fixation patch. We can find good estimates for these motion parameters if we use the right fixation patch size.

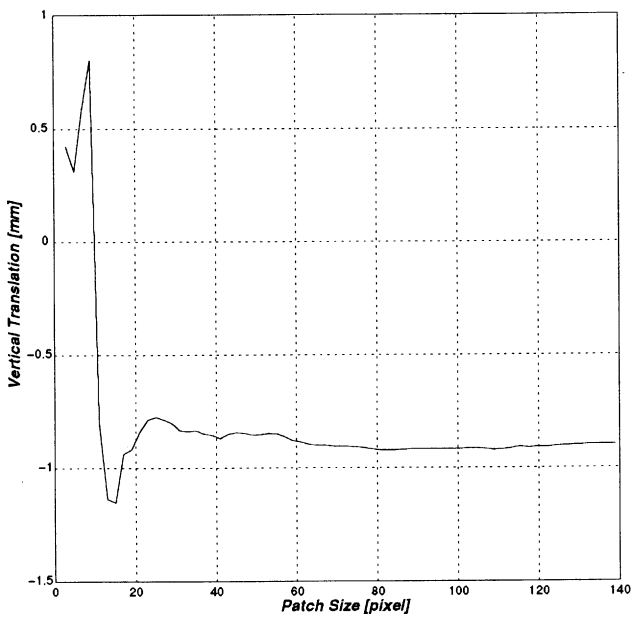

Figure 8-2: The estimated value for the vertical translation versus the fixation patch size for the landscape image sequence. The true vertical translation is zero which is apparently different from the experimental results (about $\approx -0.9\ mm$). In chapter 13, we will show that this considerable difference is due to a calibration problem.

8.3 Finding a Good Estimate for $\omega_{\mathbf{R}_o}$ Autonomously

It can be seen that the size of fixation patch has a critical effect on the estimated values of the component of rotational velocity about the fixation axis, $\omega_{\mathbf{R}_o}$. A small patch size results in a value for $\omega_{\mathbf{R}_o}$ which is usually far distant from the true value. This is possibly because in a small patch, small translations can be interpreted as large rotations. Figure 8-5 shows a hypothetical situation where (a) and (b) are a sequence of a small 3×3 pixels patch. The real motion in this case is most likely a pixel high vertical translation. But if we try to interpret it as a rotation about the patch center we will end up with a 45 deg rotation which is not acceptable, considering the assumed small motion between images.

As a conclusion, we can autonomously find a good estimate for the rotational velocity component $\omega_{\mathbf{R}_o}$ simply by using a relatively large fixation patch size.

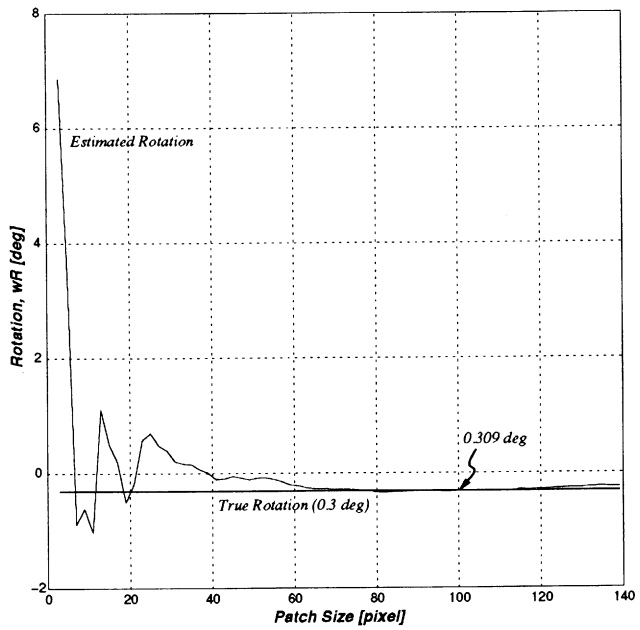

Figure 8-3: The estimated value of the component of rotation velocity about the fixation axis, $\omega_{\mathbf{R}_o}$, versus the fixation patch size for the landscape image sequence. For large patch sizes, the estimated value of $\omega_{\mathbf{R}_o}$ (about -0.309 deg) converges towards the real value of $\omega_{\mathbf{R}_o}$, -0.3 deg.

8.4 Updating the Fixation Velocity Using $\omega_{\mathbf{R}_o}$

In the previous section, we saw that a good estimate for $\omega_{\mathbf{R}_o}$ can be found using a relatively large patch but the corresponding fixation velocity estimate from such a large patch is usually not reliable. This observation suggests that we may be able to obtain better estimates for the fixation velocity components if we use the estimated value of $\omega_{\mathbf{R}_o}$ and recompute the fixation velocity.

Using only the estimate for $\omega_{\mathbf{R}_o}$ from a large patch, we can compute the total motion field at any point (x, y) on a small patch around the fixation point (*fixation*

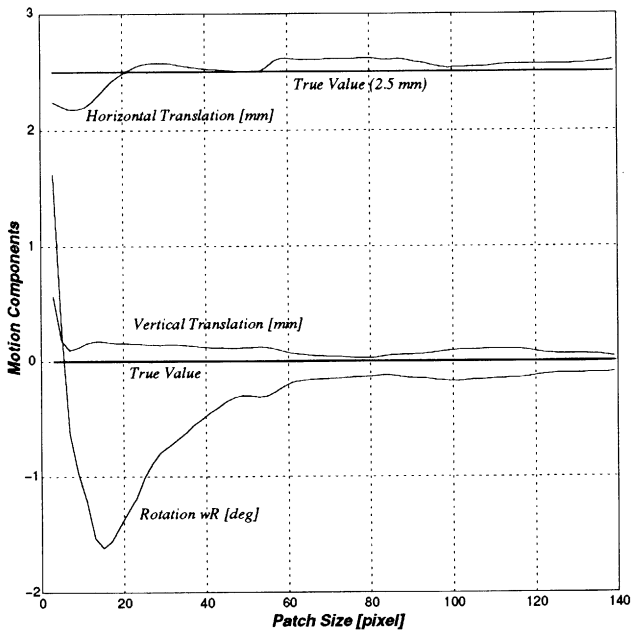

Figure 8-4: The estimated values for the horizontal and vertical translations and the rotational component $\omega_{\mathbf{R}_o}$ versus the fixation patch size for the cup image sequence. The true motion is a horizontal translation of 2.5 mm.

patch). As we showed in chapter 4

$$\begin{cases} x_t &= u_o + \frac{\omega_{\mathbf{R}_o}}{\sqrt{x_o^2+y_o^2+1}}(y - y_o) \\ y_t &= v_o - \frac{\omega_{\mathbf{R}_o}}{\sqrt{x_o^2+y_o^2+1}}(x - x_o) \end{cases} \tag{8.1}$$

where $(x_o,\ y_o)$ is the position of fixation point (located in the image plane), and (u_o, v_o) is the fixation velocity that we are about to estimate. After substituting x_t and y_t into the BCCE, eqn. (2.7), we will have

$$\left(u_o + \frac{\omega_{\mathbf{R}_o}}{\sqrt{x_o^2+y_o^2+1}}(y - y_o)\right) E_x + \left(v_o - \frac{\omega_{\mathbf{R}_o}}{\sqrt{x_o^2+y_o^2+1}}(x - x_o)\right) E_y + E_t = 0. \tag{8.2}$$

However, due to noise, the above equation does not necessarily hold for any pixel. As a result, we can find u_o and v_o by minimizing the sum of the errors over the whole

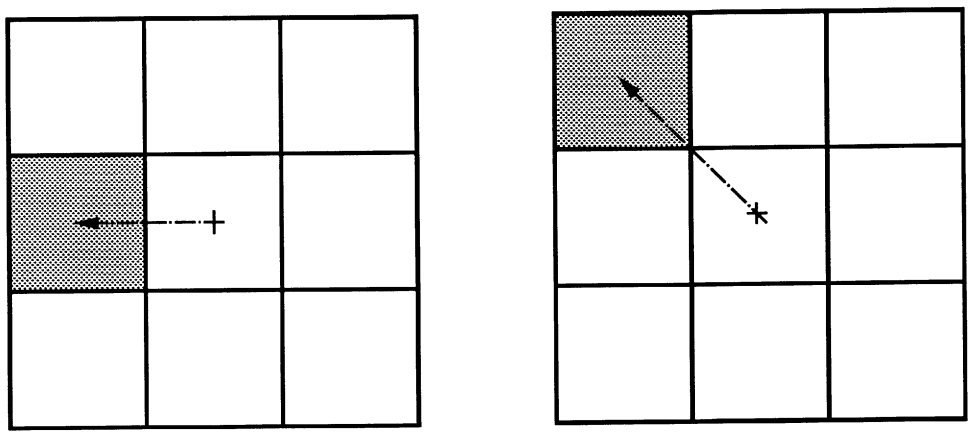

Figure 8-5: Using small fixation patch can result in wrong interpretation of large rotation. In a patch of 3×3 pixels, a pixel high vertical translation can be interpreted as 45 deg rotation which is not an acceptable answer at all, considering the finite motion between images.

fixation patch, namely by minimizing

$$\iint_p \left[\left(u_o + \frac{\omega_{\mathbf{R}_o}}{\sqrt{x_o^2 + y_o^2 + 1}} (y - y_o) \right) E_x + \left(v_o - \frac{\omega_{\mathbf{R}_o}}{\sqrt{x_o^2 + y_o^2 + 1}} (x - x_o) \right) E_y + E_t \right]^2 dx\, dy \tag{8.3}$$

with respect to u_o and v_o. This will result in the following system of linear equations,

$$\begin{bmatrix} \iint_p E_x^2 dx\, dy & \iint_p E_x E_y dx\, dy \\ \iint_p E_x E_y dx\, dy & \iint_p E_y^2 dx\, dy \end{bmatrix} \begin{pmatrix} u_o \\ v_o \end{pmatrix} = \begin{pmatrix} \iint_p \left(\frac{\omega_{\mathbf{R}_o}}{\sqrt{x_o^2+y_o^2+1}} ((x - x_o)E_y - (y - y_o)E_x) - E_t \right) E_x dx\, dy \\ \iint_p \left(\frac{\omega_{\mathbf{R}_o}}{\sqrt{x_o^2+y_o^2+1}} ((x - x_o)E_y - (y - y_o)E_x) - E_t \right) E_y dx\, dy \end{pmatrix} \tag{8.4}$$

that can be solved for the two unknowns u_o, and v_o. Note that $\omega_{\mathbf{R}_o}$ has been already computed and is a known value in this equation.

8.4.1 Improved estimations

Here, we use the updated algorithms (which take advantage of a good $\omega_{\mathbf{R}_0}$ estimation) to find estimations for the translational components of the fixation velocity.

Figures 8-6 and 8-7 compare the updated and previous estimations of the horizontal and vertical translations in the landscape images. These figures show that there are some improvements in the updated estimations especially for the vertical translation (fig. 8-7). The improvements in the updated estimations are more pronounced

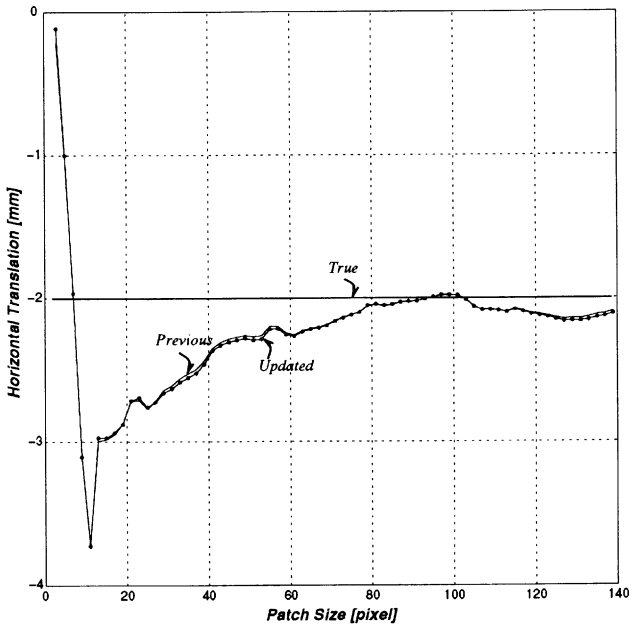

Figure 8-6: The updated and previous estimations of the horizontal translation, along the X-axis, versus the fixation patch size for the landscape image sequence.

in the plots corresponding to the cup images (figures 8-8 and 8-9). Note that we have better improvements where there is the most need for it, namely in the cup images where relative depth variations is large compared to the landscape images.

Despite improvements, the dependency of the updated translational components on the fixation patch size is still quite clear in these figures. However, we can find good estimates for these motion parameters if we choose the right fixation patch size. In

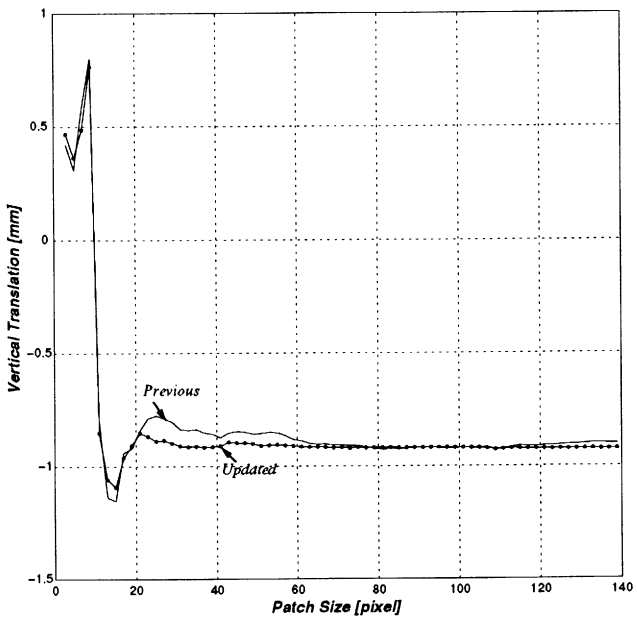

Figure 8-7: The updated and previous estimations of the vertical translation, along the Y-axis, versus the fixation patch size for the landscape image sequence.

practice, we do not know the real fixation velocity, and therefore we cannot select an appropriate fixation patch size by checking the computed values of the translational components. The next chapter introduces a technique for autonomous choice of an optimum fixation patch size.

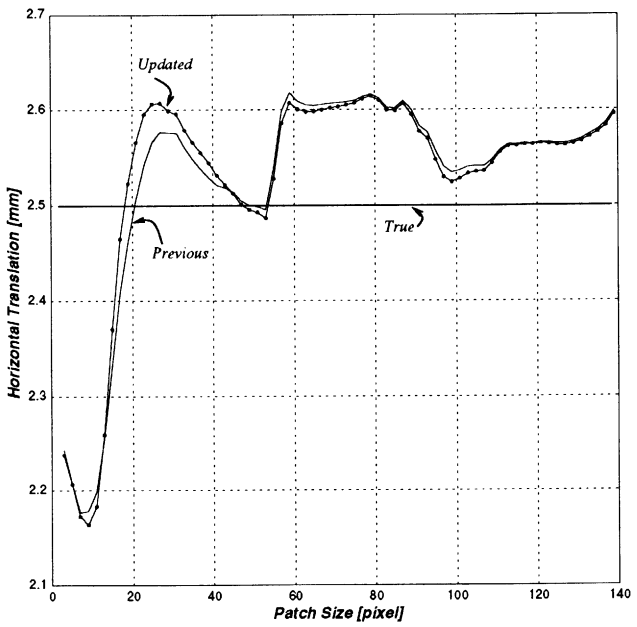

Figure 8-8: The updated and previous estimations of the horizontal translation, along the X-axis, versus the fixation patch size for the cup image sequence.

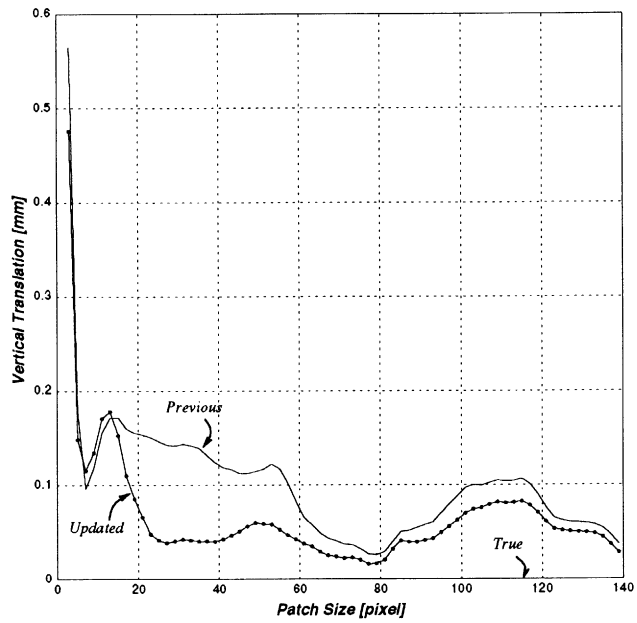

Figure 8-9: The updated and previous estimations of the vertical translation, along the Y-axis, versus the fixation patch size for the cup image sequence.

Autonomous Choice of an Optimum Fixation Patch Size

Chapter 9

The experimental results and explanations in the previous chapter suggest that relatively large patch sizes should be used in order to get a good estimate for the component of the rotation along the fixation axis, $\omega_{\mathbf{R}_o}$. On the other hand, we know that in general using a very large patch size will result in a wrong estimate for the fixation velocity because depth variations usually increase as the patch size increases.

Figures 8-1 and 8-4 showed that for any image sequence, there is an *optimum patch size* which results in good estimates for the fixation velocity components. The corresponding optimum patch size is about 100×100 pixels for the landscape image sequence (fig. 8-1) and about 50×50 pixels for the cup image sequence (fig. 8-4).

In this chapter, we will describe an autonomous technique for finding the optimum fixation patch size which results in good estimates for the fixation velocity components for any image sequence.

9.1 Normalized Error

We showed that for any given size of the fixation patch, we can find the fixation velocity components, u_o and v_o. Also the component of the rotational velocity about the fixation axis, $\omega_{\mathbf{R}_o}$, can be estimated reliably using a relatively large patch. Knowing these values, the motion field velocity (x_t, y_t) at any point (x, y) in the image plane is given by eqn. 8.1. Ideally, for any given image point (x, y) the BCCE, eqn. 2.7, must

be satisfied. However, in practice we are dealing with real images which are noisy and as a result, the term $x_t E_x + y_t E_y + E_t$ does not usually become zero. This term can be considered as an error term for the corresponding pixel. In a patch of size $p \times p$ pixels, we can add these error terms to define the *normalized error*, e, as

$$e = \frac{\sum [x_t E_x + y_t E_y + E_t]^2}{p^2}. \tag{9.1}$$

This definition allows us to compare the performance of different patch sizes by studying the behavior of the normalized error e with respect to the changes in the patch size p.

9.2 Optimum Patch Size

In this section, we show how the normalized error can be used for finding an optimum patch size which results in good estimates for the components of the fixation velocity. Any patch of a real image may include a substantial depth range. In general, there are two main groups of images. In the first group, there are moderate changes in depth variation as the patch size increases. The second group represents images where the depth variation increases significantly as the patch size increases.

9.2.1 Moderate changes in relative depth

Figure 9-1 shows the normalized error versus the fixation patch size for the landscape image sequence. Although this plot corresponds to a specific image and motion, it shows one of the two typical representations of the normalized error behavior as the patch size increases. As shown in this figure, the normalized error first increases with the patch size, reaches a peak and then dips down.

This is because initially for the smallest patch size (3×3 pixels) the algorithm finds the motion estimates that makes the BCCE error term ($x_t E_x + y_t E_y + E_t$) as

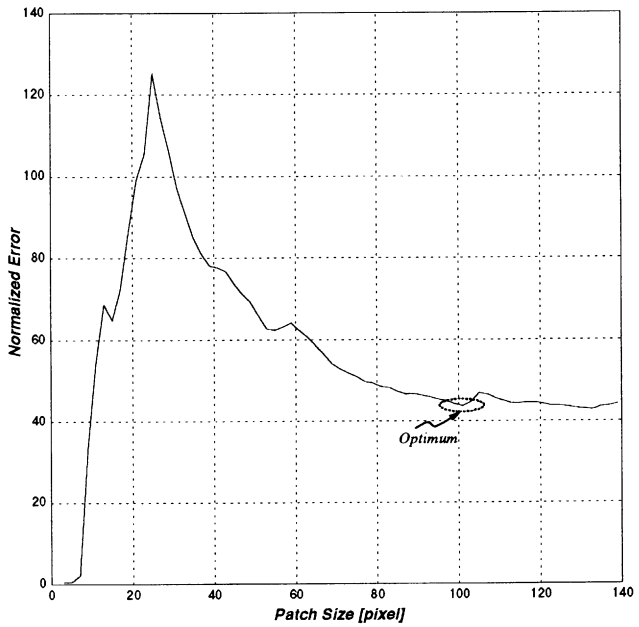

Figure 9-1: The estimated value of the normalized error e versus the fixation patch size for the landscape image sequence. The optimum patch size occurs around 100 *pixels*.

small as possible. The algorithm does a good job in minimizing the total of 9 error terms in this small patch but the motion estimates are usually very bad at this level because basically there are not enough data available to the algorithm.

In the next level, we have a patch of 5×5 pixels size which provides more data. While there is still not enough data for the algorithm to come up with good motion estimates, it finds parameters which minimize the sum of the BCCE error terms. However, the algorithm is not usually as successful as it was for the 3×3 pixels patch size because it has to deal with more error terms and this results in higher normalized error.

As we increase the patch size, the struggle between providing more data to the algorithm and satisfying more error terms continues. For relatively small patch sizes, this results in higher normalized error. The normalized error increases until it reaches a peak where the role of extra input data becomes more important than satisfying more error terms. Then by increasing the patch size, we are providing more data

to the algorithm and this gives a better motion estimate and results in a smaller normalized error.

After dipping down, the normalized error stays roughly the same in this case, because the relative depth variation does not change much with the patch size, (fig. 9-1). The optimum patch size in this example occurs around 100 $\times$ 100 pixels which corresponds to the start of the small slope in normalized error, a roughly flat portion after the first peak. In this example, relative depth changes are moderate (1250 mm to 1625 mm, about 30% difference) and stay roughly the same as the patch size increases.

9.2.2 Significant changes in relative depth

The normalized error for the cup image sequence is shown in fig. 9-2. As before, the

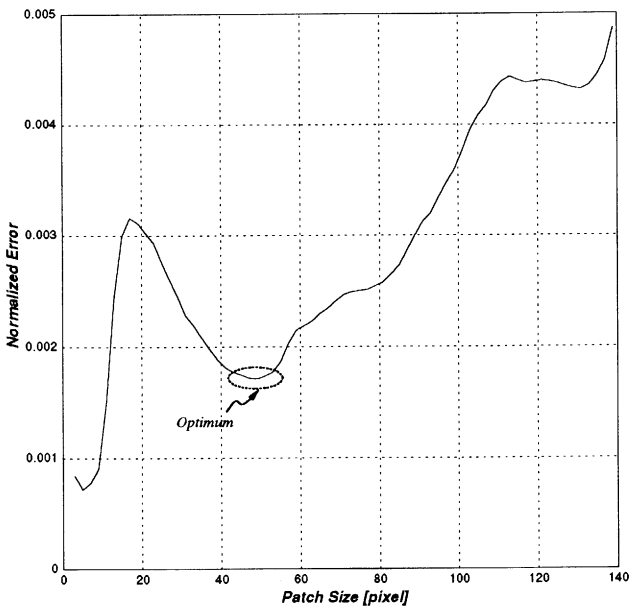

Figure 9-2: The normalized error versus fixation patch size for the cup image sequence. The optimum patch size occurs around 50 *pixels*.

normalized error first increases and after reaching a peak it dips down and then grows with the patch size again. This is because in the beginning, insufficient information

results in extremely wrong estimates and this causes the normalized error to increase with the patch size. As we are providing more and more data to the algorithm, we obtain better estimates for the motion components and this decreases the normalized error. If we increase the patch size beyond an optimum size, which occurs at about 50 pixels in this example, the normalized error starts increasing again. In this 50×50 pixels patch, we have a considerable amount of relative depth changes (from 584 mm to 914 mm, about 57% increase). Such significant relative depth variation leads to larger errors in the fixation velocity estimates which in turn results in a larger normalized error as p grows.

9.3 Autonomous Choice of Optimum Patch Size

As one might expect, the optimum fixation patch size depends on the patch topology and texture which may vary from image to image. However, the general pattern of the normalized error allows us to autonomously find an optimum fixation patch size which gives good estimates for the fixation velocity components.

In the case where considerable changes in the relative depth occur with patch size increase, as in the cup image sequence, the optimum fixation patch size corresponds to the minimum normalized error that occurs after the peak value of the normalized error. And in cases where the relative depth does not change significantly with patch size, as in the landscape image sequence, the optimum fixation patch size is where the normalized error does not change considerably as the patch size increases.

A human operator may not have much problem identifying the optimum patch size on the normalized error plots. But our aim is to come up with a simple algorithm which allows a machine to autonomously find the optimum patch size from any given normalized error data set.

9.3.1 Algorithm

This section describes the algorithm for obtaining the optimum fixation patch size from any normalized error data set. The general algorithm is composed of the following steps:

- **Step 1:** *Setting the patch size bounds*

All the experimental results unanimously show that the motion estimates from a small patch are not reliable at all. As a result, we can put a lower bound on the patch size. By taking into account the camera parameters and the image size, we have used a 15×15 pixel patch as the lower bound of the patch size. Moreover, the square shape of the patch, the location of the fixation point, and the image size dictates an upper bound on the patch size. As a result, we have used 140×140 *pixels* as the upper bound in our experiments.

- **Step 2:** *Computing the normalized error slope*

Denoting the normalized error at patch i as $e[i]$, we define the slope at patch i as

$$S[i] = \frac{e[i+1] - e[i]}{e[i]}. \tag{9.2}$$

The slope $S[i]$ is dimensionless and shows the relative change of the normalized error as the patch i changes to patch $i+1$.

- **Step 3:** *Setting a slope index*

By searching through the slope space, we can find the steepest (most negative) slope and denote it as $Smax$. This definition allows the algorithm to get a sense of steepness (or flatness) at any point on the normalized error curve. We define the slope index $Sind$ as a small percentage (about 15%) of the steepest slope $Smax$. Study of many normalized errors plots has shown that this choice of the $Sind$ allows us to identify relatively flat portions in a typical normalized error curve.

- **Step 4:** *Searching for the optimum patch size*

We choose the lower bound patch size as the first candidate for the optimum size.

Then, we move to the next patch size and select it as the new nominated optimum patch size if it satisfies the following two conditions:

- *First condition:* Its normalized error e[i] should be less than the normalized error value of the previously nominated optimum point.

- *Second condition:* Its corresponding slope $S[i]$ should be steeper (more negative) than the slope index, *Sind*.

We continue this search process until we reach the upper bound of the patch size.

• **Step 5:** *Locating the optimum patch size*

After checking all the data, the point immediately after the last nominated point is selected as the optimum point.

9.3.2 Experimental results

The above algorithm has been applied to the normalized error data set of the landscape and the cup image sequences (figures 9-1 and 9-2) to obtain the optimum patch sizes. The corresponding experimental results of locating the optimum patch size are shown in figures 9-3 and 9-4. In these figures, the nominated optimum points are shown by small circles on the normalized error curves. It can be seen that for both cases the algorithm finds the optimum points correctly.

Figure 9-3 shows that the optimum patch size for the landscape image sequence is selected at 101 *pixels* which corresponds to a small field of view (about 2×2.4 deg). If we go back to figures 8-6 and 8-7 again, we see that one of the best estimations for the translational components occur at this optimum patch size (101 *pixels*). The optimum patch size for the cup image sequence is selected at 47 *pixels* (fig. 9-2). Similarly, figures 8-8 and 8-9 show that we obtain one of the best combined motion estimates at this optimum point (47 *pixels*). This *optimum patch size* for the cup image sequence makes approximately the same field of view as the one for the landscape image sequence (about 2×2.4 deg). This is an important observation considering that we have obtained roughly the same *optimum field of view* for two totally different

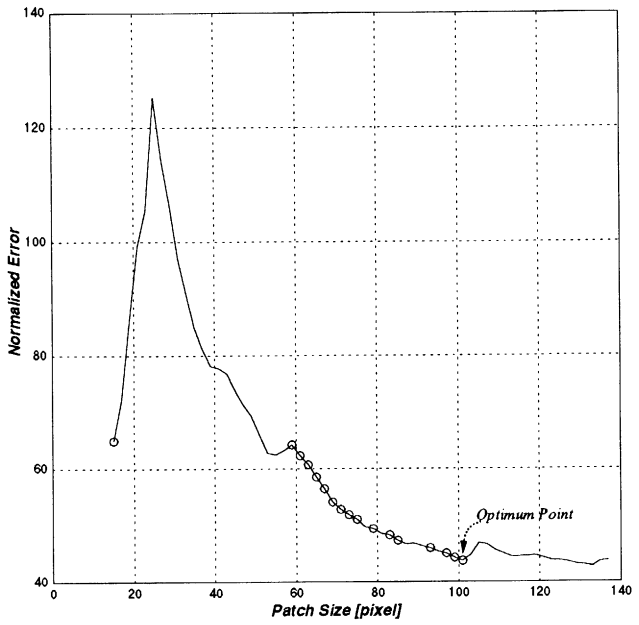

Figure 9-3: Searching process of finding the optimum patch size for the *landscape* image sequence. The nominated points are shown by small circles. The last point represents the optimum point which occurs at 101 *pixels* in this case.

images, cameras, and focal lengths.

9.3.3 Further results

In order to test our algorithm further, we have run it on many other image sequences with smaller and larger motions. The algorithm has worked successfully in finding the optimum patch sizes in all cases. Some of the corresponding experimental results are shown in figures 9-5, 9-6, 9-7, and 9-8. These experimental results for the other images sequences show that the corresponding optimum patch sizes are close but not necessarily the same as the values we obtained before. However, in every case the obtained optimum point represents the patch size which results in one of the best motion estimates.

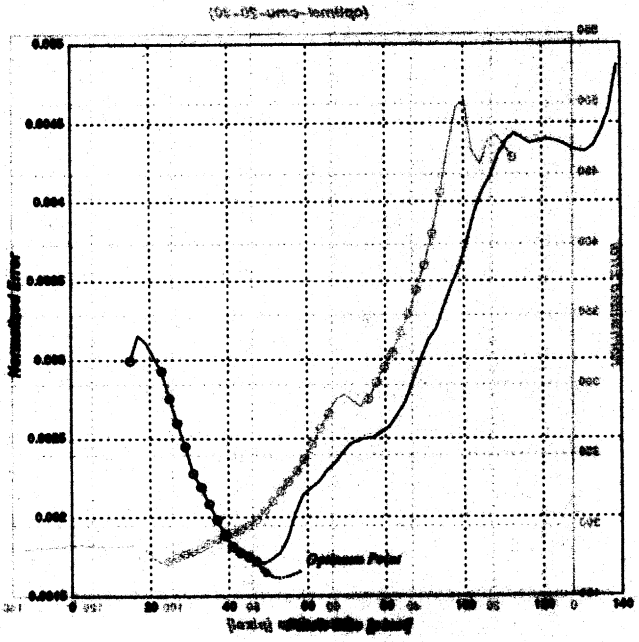

Figure 9-4: Searching process of finding the optimum patch size for the *cup* image sequence. The nominated points are shown by small circles. The last point represents the optimum point which occurs at 47 pixels in this case.

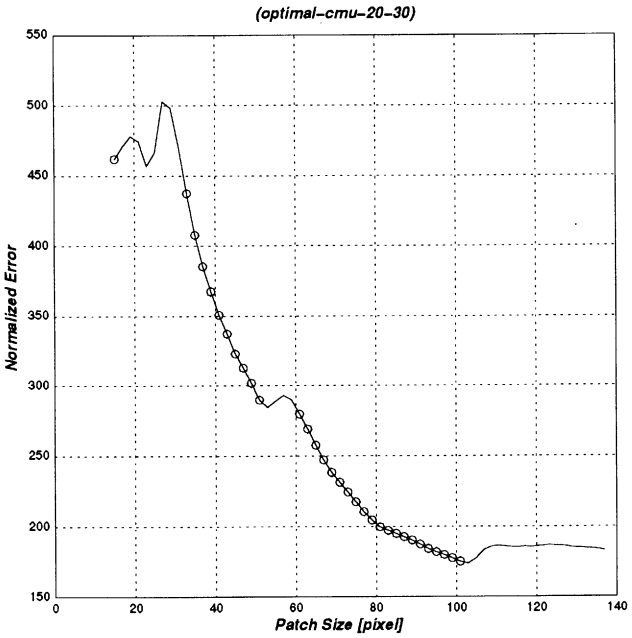

Figure 9-5: Searching process of finding the optimum patch size for the *landscape20-30* image sequence. The motion is two times as large as before (-4 *mm* translation and -0.6 deg rotation). The nominated points are shown by small circles. The last point represents the optimum point which occurs at 101 *pixels* in this case.

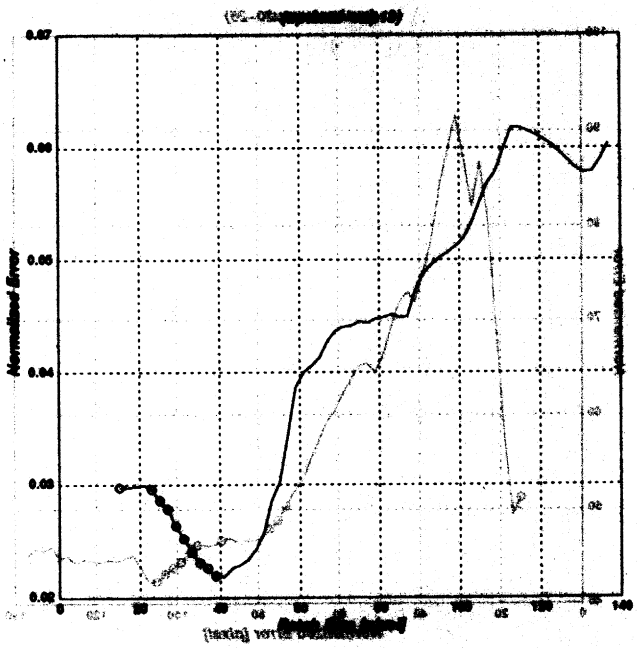

Figure 9.6: Searching process of finding the optimum patch size for the *cup13* image sequence. The motion is two times as large as before (5 mm translation). The nominated points are shown by small circles. The last point represents the optimum point which occurs at 39 *pixels* in this case.

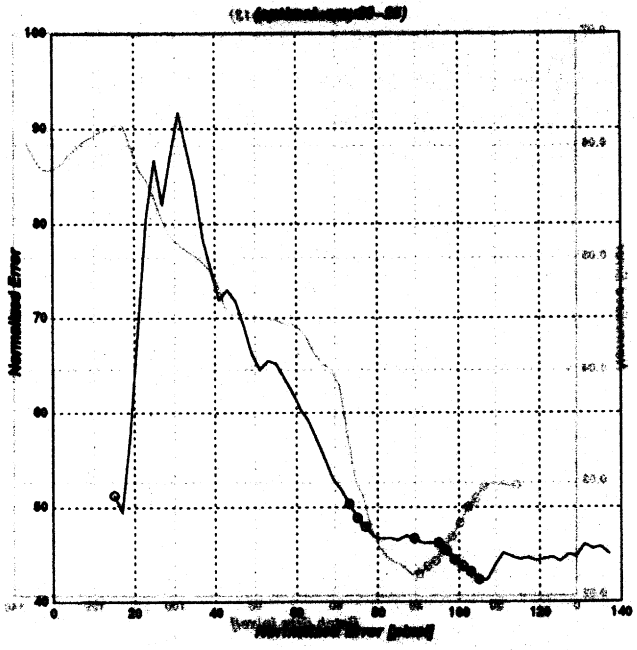

Figure 9-7: Searching process of finding the optimum patch size for the *landscape20-25* image sequence. The motion is −2 *mm* translation and −0.5 deg rotation. The nominated points are shown by small circles. The last point represents the optimum point which occurs at 105 *pixels* in this case.

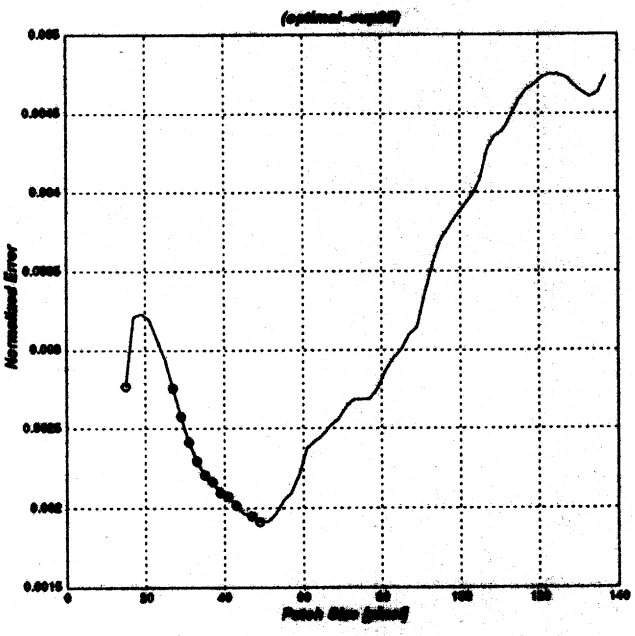

Figure 9-8: Searching process of finding the optimum patch size for the *cup23* image sequence. The motion is -2.5 *mm* translation. The nominated points are shown by small circles. The last point represents the optimum point which occurs at 49 *pixels* in this case.

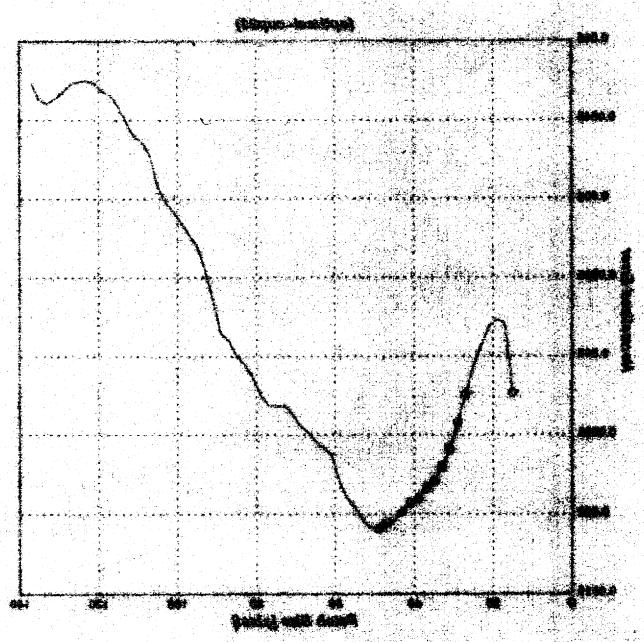

Figure 9-8: Searching process of finding the optimum patch size for the cup23 image sequence. The motion is −2.5 mm translation. The nominated points are shown by small circles. The last point represents the optimum point which occurs at 49 pixels in this case.

Autonomous Choice of an Appropriate Fixation Point

Chapter 10

In general, our fixation algorithms do not place any restrictions on the choice of the fixation point location and virtually any point can be chosen as the fixation point. Among all points, the choice of principal point (image center) makes the formulations simpler. However, in practice, one should take some more considerations into account while choosing an appropriate fixation point. Most significantly, the motion of the chosen fixation point should be detectable using the information from its corresponding patch. To clarify this, we can consider a patch which has a uniform brightness. Choosing the center of such a patch as the fixation point will not be useful, because the motion of such a point is irrecoverable using only the information from that patch. This chapter introduces a technique for autonomous choice of an appropriate fixation point.

10.1 Algorithm

Similar to chapter 4 (when using $\omega_{\mathbf{R}_o} = 0$), the least squares method can be applied to the BCCE terms to obtain the following system of linear equations for the uniform motion field (u, v) on a patch as

$$\begin{bmatrix} \iint_p E_x^2 dx\, dy & \iint_p E_x E_y dx\, dy \\ \iint_p E_x E_y dx\, dy & \iint_p E_y^2 dx\, dy \end{bmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -\iint_p E_t E_x dx\, dy \\ -\iint_p E_t E_y dx\, dy \end{pmatrix}. \tag{10.1}$$

It is obvious that the solution for (u, v) exists (i.e. motion is detectable) if the determinant of the above matrix

$$D = \left(\iint_p E_x^2 dx\, dy\right)\left(\iint_p E_y^2 dx\, dy\right) - \left(\iint_p E_x E_y dx\, dy\right)^2 \tag{10.2}$$

is not zero. However, this is not a reliable criteria for real images because due to noise we may have $D \neq 0$ but it does not guarantee that the patch is an appropriate one.

If we denote the smaller eigenvalue of the coefficient matrix in eqn. 10.1 by λ_s,

$$\lambda_s = \tfrac{1}{2}\left(\iint_p (E_x^2 + E_y^2) dx\, dy - \sqrt{\iint_p (E_x^2 - E_y^2)^2 dx\, dy + 4(\iint_p E_x E_y dx\, dy)^2}\right) \tag{10.3}$$

then we can define a good fixation point as a point whose corresponding patch has the largest λ_s. Using such a patch not only guarantees a solution ($D \neq 0$) but also ensures that our solution (u, v) is not sensitive to noise errors in the coefficient matrix of eqn. 10.1.

The reasoning behind using the largest λ_s is the form of the characteristic polynomial of the coefficient matrix in 10.1,

$$F(\lambda) = \lambda^2 - 2\left(\iint_p (E_x^2 + E_y^2) dx\, dy\right)\lambda + \left(\iint_p E_x^2 dx\, dy\right)\left(\iint_p E_y^2 dx\, dy\right) - \left(\iint_p E_x E_y dx\, dy\right)^2. \tag{10.4}$$

When λ is large, small errors in the coefficients results in negligible error in $F(\lambda)$ compared to the case when λ is small. This implies that in patches with larger λ_s, the apparent motion components (u, v) are less sensitive to small errors in the coefficients which may occur due to noise.

10.2 Discussion

It is easy to implement the λ_s criteria for autonomous choice of a good fixation point. This criteria results in reliable choices for the fixation point even in real noisy images.

For patches with relatively uniform brightness the λ_s is small which means that we should avoid choosing the fixation point in such a patch. We will get larger and larger λ_s's as we choose patches with more features and brightness variations.

We have addressed the question of finding an appropriate fixation point (the center of a fixation patch) among a number of given patches. But which patches should we check in the first place? We can search the whole image for a globally optimum location of a fixation point in the following steps:

- *Step 1*: Divide the whole image into 4 quadrants and find the corresponding λ_s for each quadrant.
- *Step 2*: Use the quadrant with the largest λ_s as a new base image.
- *Step 3*: Repeat steps 1 & 2 until reaching a quadrant with an acceptable size.

However, performing such a comprehensive search may not always be necessary. Instead, we can check a limited number of neighboring patches (near the principal point, for convenience) and choose the center of the one with the largest λ_s as the fixation point.

For patches with relatively uniform brightness the λ_1 is small which means that we should avoid choosing the fixation point in such a patch. We will get larger and larger λ_1's as we choose patches with more features and brightness variations.

We have addressed the question of finding an appropriate fixation point (the center of a fixation patch) among a number of given patches. But which patches should we check in the first place? We can search the whole image for a globally optimum location of a fixation point in the following steps:

- Step 1: Divide the whole image into 4 quadrants and find the corresponding λ_1 for each quadrant.
- Step 2: Use the quadrant with the largest λ_1 as a new base image.
- Step 3: Repeat steps 1 & 2 until reaching a quadrant with an acceptable size.

However, performing such a comprehensive search may not always be necessary. Instead, we can check a limited number of neighboring patches (near the principal point, for convenience) and choose the center of the one with the largest λ_1 as the fixation point.

Tracking without Moving the Camera

Chapter 11

The fixation method requires a sequence of fixated (tracked) images as its input. However, in general the acquired image sequences may not be fixated at any point and even if they are it is not easy to find that fixation point.

Our fixation method does not depend on how the fixated images are obtained. But along the course of this thesis work, we were confronted with the challenge of constructing a sequence of fixated (tracked) images from an arbitrary image sequence.

This chapter describes the experimental results and the implementation issues involved in constructing sequences of fixated images from several real images sequences.

11.1 Background

The task of constructing a sequence of fixated images is, in essence, the well known *tracking* problem. People have been working on different aspects of this problem using various techniques for many years [43, 22, 53]. For example, Aloimonos & Tsakiris [5] propose a method for tracking a foveated target of known shape; Bandopadhay et al. [10] use optical flow and feature correspondence for tracking the principal point in order to find the motion in a special case (they assume that there is no rotation along the optical axis) without considering noise; and Sandini & Tistarelli [52] use an optical flow based tracking method for finding the depth in a special case (no rotation along the optical axis). All these methods use optical flow and/or feature

correspondence and address only special cases. There has also been some work on using visual tracking for finding the trajectory of an object moving in an environment [15, 90].

Traditionally, tracking has been associated with mechanically moving the camera to keep the image of a particular point stationary at the image center. Some techniques even rely on such a system. For example, Thompson [74] introduces an optical flow method for recovering the motion in special case where the rotational velocity along the optical axis is zero. His method requires a sequence of tracked images at the principal point but he acknowledges that the actual implementation of such tracking requirement in engineering systems is not possible yet.

Hardware tracking is done by physically moving the camera with respect to the environment. Considering that in general the point of interest has a motion relative to the observer, the 2nd fixated image cannot be obtained in one step. As a result, feedback control loop is required for the camera rotation system to compensate for the errors resulting from the new position of the fixation point [46, 20, 24, 37, 89, 19]. These difficulties and other problems such as expense, real time response, and potential errors involved make mechanical tracking unattractive especially for our vision system.

11.2 Pixel Shifting Process

Here, we use the *pixel shifting process* described in chapter 5 for constructing a sequence of fixed images from an arbitrary image sequence. This method solves the tracking problem in its most challenging case. In other words, it does not require any knowledge about the motion or shape. Furthermore, the fixation point is not restricted to the principal point (image center) and virtually any point can be chosen as the fixation point. The *pixel shifting process* is done purely in software without any need to mechanically move the camera for tracking. It is computationally simple and

uses neither optical flow nor feature correspondence. Instead, brightness gradients of the initial input images are used directly.

11.2.1 Bilinear Interpolation

We showed that constructing a fixated image is the same as finding the brightness E for any pixel (x, y) of such an image, (see chapter 5). We proved that the brightness E at pixel (x, y) of the 2nd fixated image is the same as the brightness at the pixel $(x - Tu, y - Tv)$ of the 2nd initial image where the shifting vector (u, v) is given by eqn. 5.4 and T is the time interval between two initial images.

In practice, the point $(x - Tu, y - Tv)$ does not exactly coincide with any pixel. Instead it is usually surrounded by four pixels whose brightnesses may be denoted by $E_{i,j}$, $E_{i,j+1}$, $E_{i+1,j}$, and $E_{i+1,j+1}$, fig. 11-1. In this figure, p and q are the horizontal

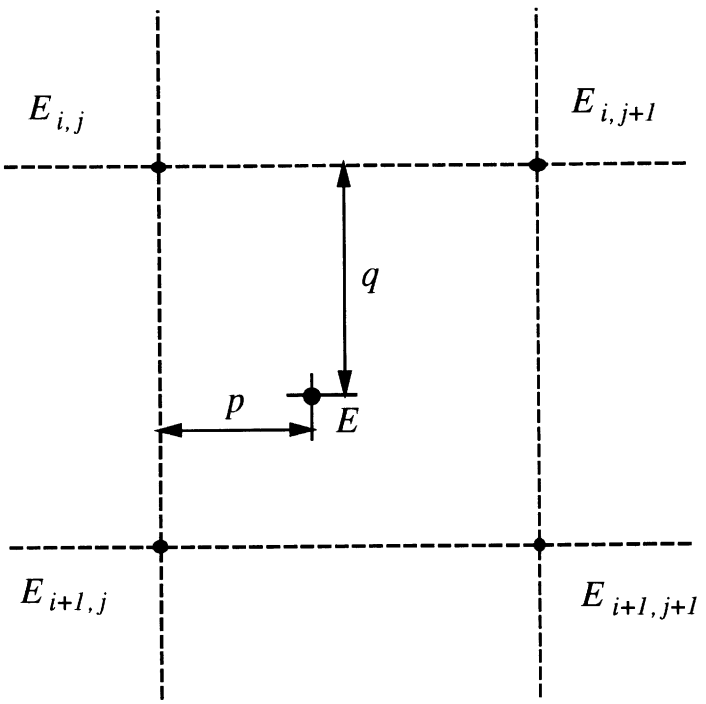

Figure 11-1: The mapped point in the 2nd initial image does not usually coincide with any single pixel. Instead it is usually surrounded by four pixels.

and vertical distances of the mapped point from pixel (i, j). Considering that this can happen for any pixel, the *average* $\frac{1}{4}(E_{i,j} + E_{i,j+1} + E_{i+1,j} + E_{i+1,j+1})$ is not a good estimation for E because it corrupts the constructed image by introducing aliasing.

Bilinear interpolation of the surrounding brightness levels has proven to be a very good estimate for E which is given as,

$$E = (1-p)(1-q)E_{i,j} + p(1-q)E_{i,j+1} + q(1-p)E_{i+1,j} + pqE_{i+1,j+1}. \qquad (11.1)$$

As shown in fig. 11-1, p and q represent the horizontal and vertical distance of the mapped point from pixel (i, j). Such an algorithm gives the largest weight to the pixel closest to the mapped point and results in the exact brightness value when it coincides with any pixel, $p = q = 0$.

All the constructed images in this work are obtained using *bilinear interpolation.* Our experimental results have shown that such interpolation is quite satisfactory. There are some other techniques such as *bicubic interpolation* [1, 13, 32, 49, 50] which are much more expensive, however we did not find that we needed to use them in this work.

11.3 Construction of Fixated Images

The *landscape* and *cup* image sequences in figures 7-1 and 7-4 are used as input (initial) images in the following experiments. As we discussed earlier, the 1st initial images (top images) in those figures are directly used as the 1st fixated images. Then the *pixel shifting process* and the *bilinear interpolation* are applied to the 2nd initial images (bottom images in figures 7-1 and 7-4) to construct the 2nd fixated images, figures 11-2 and 11-3. These constructed images are quite good and look as natural and crisp as the original images do. We will describe the quality of these images further in the following sections.

Depending on the size and direction of the equivalent rotational velocity $\boldsymbol{\Omega}$ (see

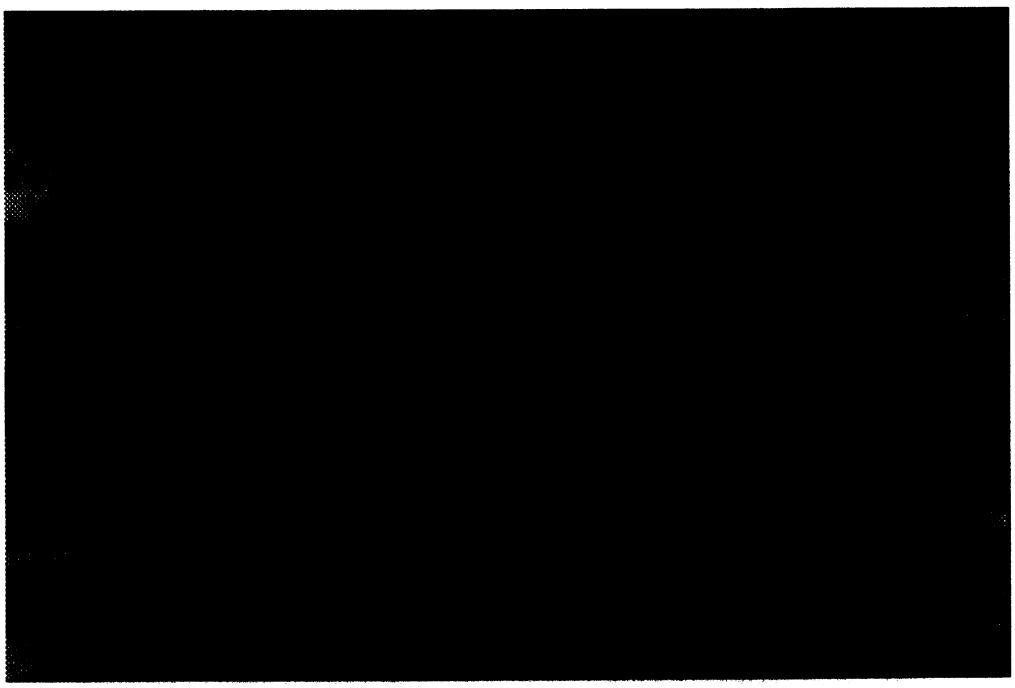

Figure 11-2: The constructed, *2nd fixated*, image for the landscape image sequence.

chapter 5), the brightness E at some border pixels are not computable because they are mapped to points outside the initial images domain. The brightness at such bordering pixels are given an arbitrary value of 0 which causes the appearance of bold black lines at the border of constructed images. This should not concern us because in general the results near the image borders are not considered reliable anyway.

11.4 Spatial and Temporal Gradient Maps

The gradient maps are good measures for studying the quality and characteristics of fixated image sequences. This section examines the gradient maps of two different fixated image sequences that we have constructed from real image sequences.

Figure 11-3: The constructed, *2nd fixated* image, for the cup image sequence.

11.4.1 Landscape fixated image sequence

The combination of the 1st initial image (top image in fig. 7-1) and the 2nd fixated image in fig. 11-2 form the landscape fixated image sequence. The corresponding spatial gradient maps in fig. 11-4 show that these gradients contain valuable information. The vertical and horizontal features of the initial images are indirectly represented in the spatial gradients.

The temporal gradient map of the landscape fixated image sequence is shown in fig. 11-5. This map contains very important information. First of all it clearly shows the characteristic of a fixated image sequence. It is clear that both the horizontal and vertical features of the image sequence become more obvious as their distance from the fixation point location (image center in this case) increases. Secondly, the appearance of the horizontal and vertical lines here provides hints about the existence of a rotational component about the fixation axis. And finally the dominant vertical lines are an indication that the equivalent rotational velocity has a major component

about the vertical axis.

11.4.2 Cup fixated image sequence

The fixated cup image sequence consists of the top image in fig. 7-4 (as the 1st fixated image) and the 2nd fixated image in fig. 11-3. Figure 11-6 shows the spatial gradient maps for this image sequence. The horizontal gradient map (top) identifies the vertical edge-like features and the vertical gradient map (bottom) detects the horizontal edge-like features in the image. We should emphasize here that we neither intended to find edges nor have we used those. However, it is important to observe that spatial gradients (simple horizontal and vertical differences) of fixated images indirectly capture important features of the images.

Figure 11-7 represents the temporal gradient map of the fixated cup image sequence. This map is dominated by vertical lines which indicate that the rotational component about the fixation axis is negligible and the equivalent rotational velocity has only a component about the vertical axis. Furthermore these vertical lines become more evident as their distance from the image center increase which is an indication that the fixation point is located near the image center.

11.5 Summary

The experimental results in this chapter show that the *pixel shifting process* can be easily used for constructing a sequence of images fixated at any arbitrary point. This software based technique is computationally simple and does not require moving the camera for tracking the desired fixation point.

The novel representation of the spatio-temporal gradients by their corresponding maps showed that gradients not only preserve the image features but also capture the motion in a unique way which reflects the characteristics of fixated image sequences.

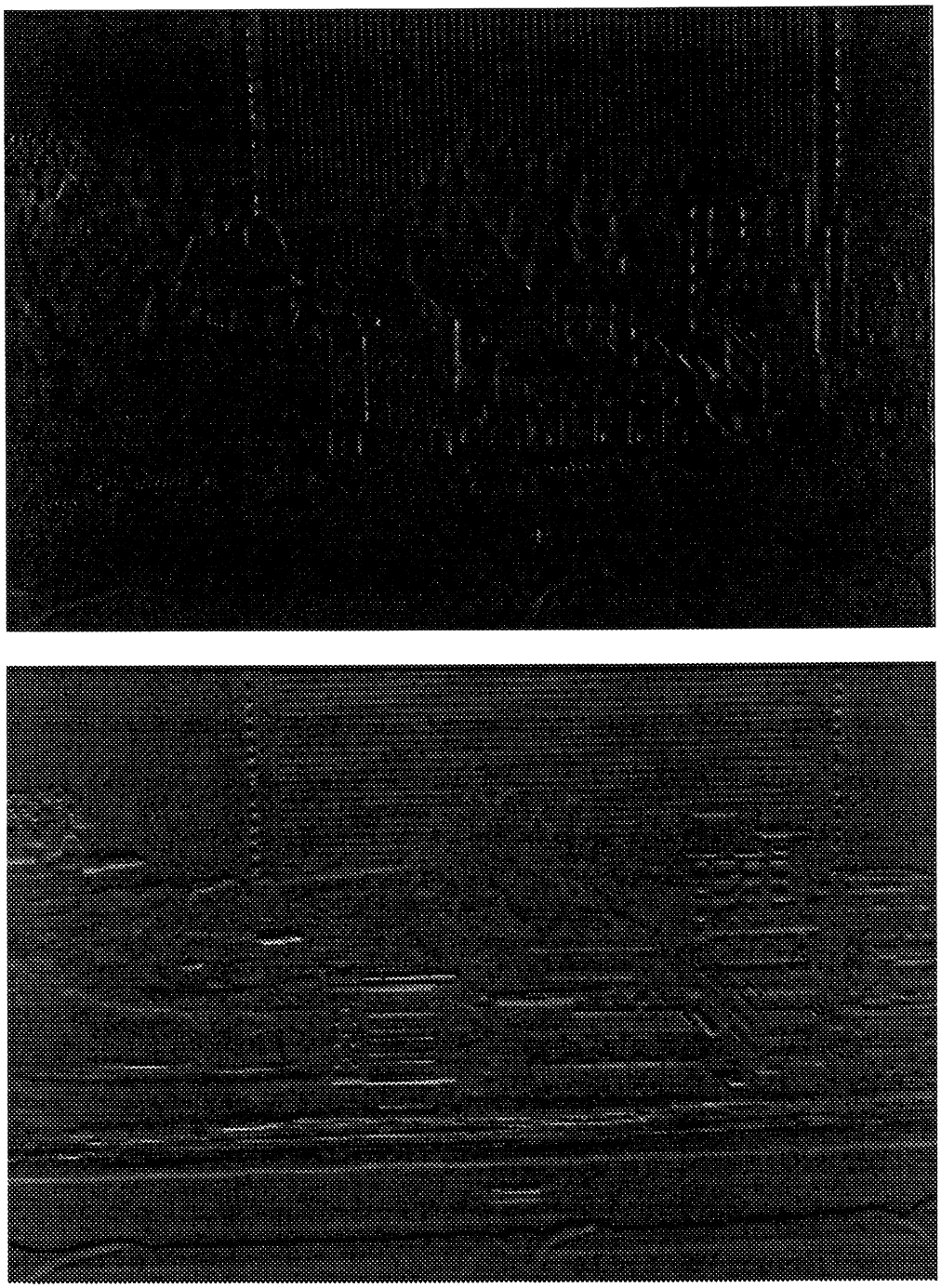

Figure 11-4: The spatial gradient maps of the *fixated* landscape image sequence in *x direction* (top) and *y direction* (bottom).

Figure 11-5: The temporal gradient map of the *fixated* landscape image sequence.

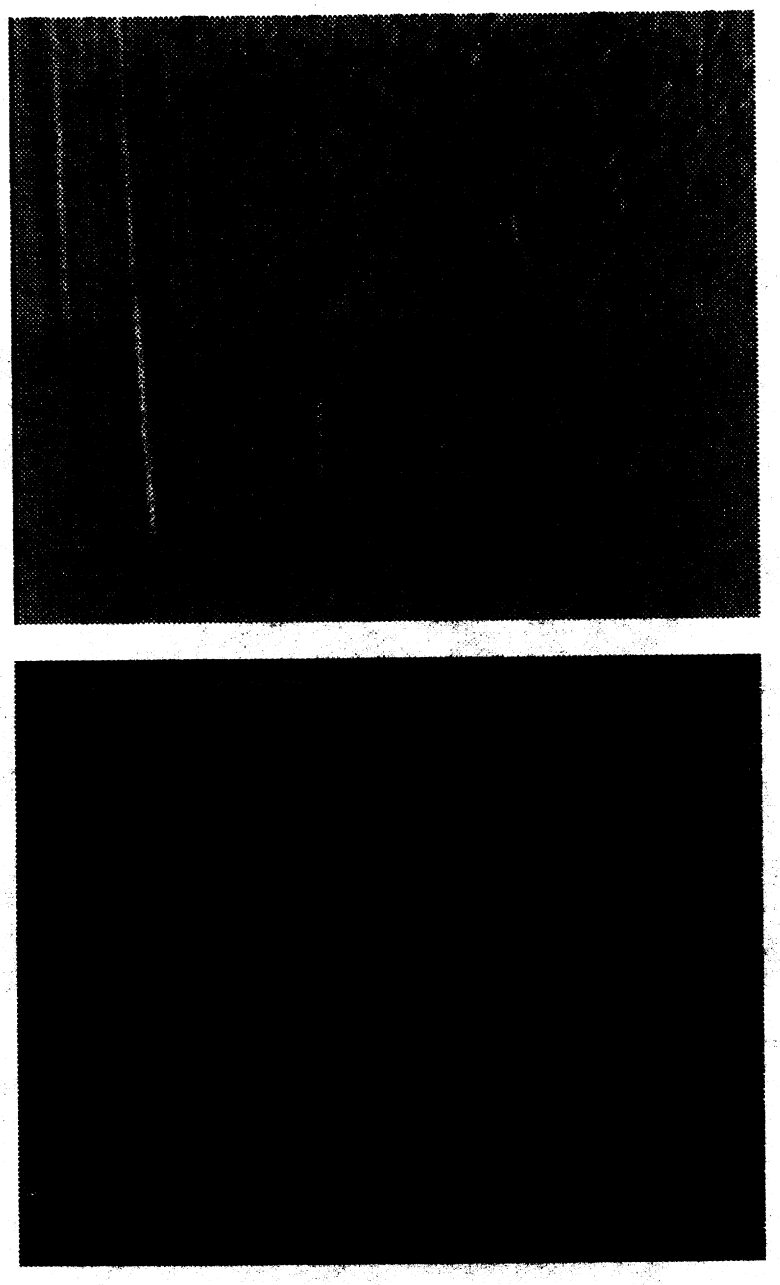

Figure 11-6: The spatial gradient maps of the *fixated* cup image sequence in *x direction* (top) and *y direction* (bottom)

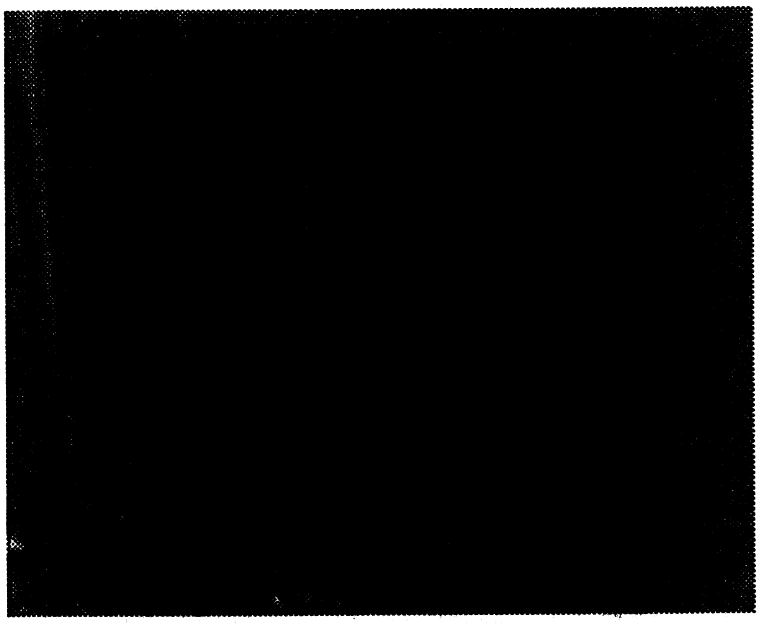

Figure 11-7: The temporal gradient map for the *fixated* cup image sequence.

Figure 11-7: The temporal gradient map for the Reader cup image sequence.

Depth Map Recovery

Chapter 12

This chapter describes how depth maps are recovered from real image sequences. It also describes implementation issues and the techniques used in the recovery of depth maps.

12.1 Introduction

Earlier in chapter 3, we proved that ideally the depth at any point of a fixated image is given by eqn. 3.35,

$$Z = \frac{(\mathbf{s} \cdot \mathbf{t})}{\frac{(\mathbf{v} \times \hat{\mathbf{R}}_o) \cdot \mathbf{t}}{\|\mathbf{R}_o\|} - E_t - \omega_{\mathbf{R}_o} \mathbf{v} \cdot \hat{\mathbf{R}}_o} \tag{12.1}$$

where $\hat{\mathbf{R}}_o$ is the unit vector along the fixation axis and $\mathbf{s}$ and $\mathbf{v}$ are the known vector functions of pixel position (x, y) and spatial gradients (E_x, E_y) as given in equations 2.9 and 2.10.

The translational velocity $\mathbf{t}$ is obtained by finding the eigenvector corresponding to the smallest eigenvalue of matrix M in eqn. 3.31. The optimal patch size found in chapter 9 is used for the estimation of $\mathbf{t}$.

All the computations in this chapter are performed using the data from the fixated image sequences that we constructed in chapter 11.

12.2 Detecting the Depth Flaws

It is well known that depth recovery from real images is not perfect because of noise and other characteristics of real images. This section describes the techniques for detecting pixels where depths are not acceptable.

Using the notations *Num* and *Denom* as,

$$Num = (\mathbf{s} \cdot \mathbf{t})(\mathbf{s} \cdot \mathbf{t}) \tag{12.2}$$

and

$$Denom = \left(\frac{(\mathbf{v} \times \hat{\mathbf{R}}_o) \cdot \mathbf{t}}{\|\mathbf{R}_o\|} - E_t - \omega_{\mathbf{R}_o} \mathbf{v} \cdot \hat{\mathbf{R}}_o \right) (\mathbf{s} \cdot \mathbf{t}). \tag{12.3}$$

equation 12.1 can be written as,

$$Z = \frac{Num}{Denom}. \tag{12.4}$$

Using this equation, we can compute depth Z at any single pixel in the image. However, the recovered depth is not always reliable. We call a depth Z unacceptable if it satisfies any of the following cases.

- *Case 1: Denom is negative.*

This condition results in a negative depth which should not happen in our vision system. This usually happens where the data is noisy.

- *Case 2: Denom is zero.*

This case results in an irrecoverable depth ($Z = \frac{0}{0}$) or wrong depth ($Z = \infty$).

It may occur due to many reasons such as zero translational velocity, in case the pixel is in a patch with uniform brightness (zero gradients), or when the apparent motion is in a direction perpendicular to the spatial gradients.

Figure 12-1 shows the depth flaw map for the fixated cup image sequence obtained by using the above criteria for detecting the points with unacceptable depth. Any black point in this map represents a pixel whose computed depth is not acceptable.

It is quite obvious from this figure that if we compute the depth using only the data

Figure 12-1: The flaws in the depth map for the fixated cup image sequence. The pixels with unacceptable depth are shown in black.

from a single pixel, then we will end up with considerable number of pixels where depths are not acceptable.

12.3 Constructing a Primary Depth Map

Figure 12-2 shows the depth map where each depth value is computed using only the data from its corresponding pixel. Using such a method leave us with many pixels of unacceptable depths which are left blank (white) in this depth map.

This is a primary depth map and obviously is not very informative because depth information is missing in many areas. In the next section the first effort is made for estimating the depth at such points.

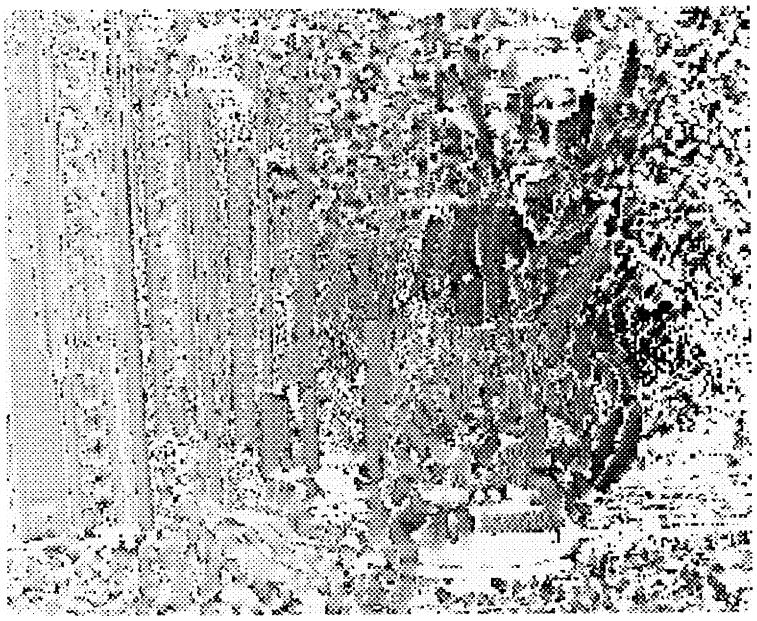

Figure 12-2: The initial depth map for the fixated cup image sequence. The areas close to the viewer are bright and the pixels whose depths are not acceptable are left blank (white).

12.3.1 Filling in the Missing Depths

At any pixel where the depth information is missing (depth is unacceptable), we can find a depth estimate by averaging the reliable depths at its surrounding pixels. The notation r_f is used for the radius of such a patch. This radius is defined in a way that forms a square patch whose side has a length of $(2 \times r_f + 1)$ *pixels*. Figure 12-3 shows the corresponding completed depth map. A maximum patch size of radius $r_f = 6$ *pixels* has been used for finding an estimate for the points where depths were not known in the initial depth map, fig. 12-2. Although this primary depth map is not perfect, it delivers very useful clues about the boundary of objects in the environment (books, cup, and spoon).

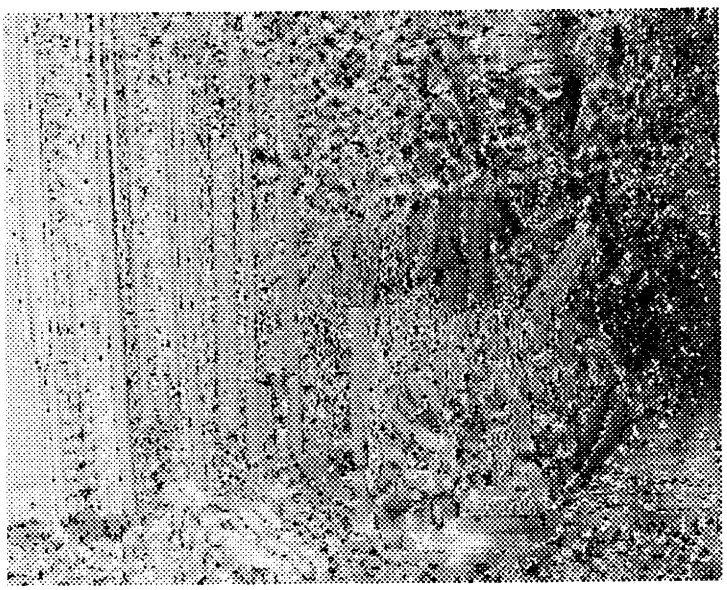

Figure 12-3: The completed depth map for the fixated cup image sequence with $r_f = 6$ *pixels*. The areas close to the viewer look brighter.

12.4 Improving the Depth Map

We can considerably improve the depth map by using the data from a surrounding patch for computing the depth at any pixel point. We denote the radius of such patch with r_p. Similar to r_f, the radius r_p is defined in a way to form a square patch whose side has a length of $(2 \times r_p + 1)$ *pixels*.

Applying such a simple technique decreases the number of depth flaws and increases the quality of depth map considerably. Figure 12-4 shows the results when a patch of 1 *pixel* in radius is used for depth computation at any pixel ($r_p = 1$ *pixel*). Although the depth flaws (in the top of the fig. 12-4) have not disappeared, they have shrunk noticeably when compared to the previous case.

The initial depth map is shown in the middle of fig. 12-4 where the pixels with unreliable depth estimates are left blank (white). The completed depth map is given at the bottom of fig. 12-4 where a patch of maximum 9 *pixels* in radius ($r_f = 9$ *pixels*) is used for finding depth estimates at points where depths were not known in the initial

depth map. The shape of the objects in the image have started to become identifiable in this completed depth map.

12.5 Even Better Depth Maps

The depth maps can be further improved by using larger patches for depth estimation at any single pixel. Figures 12-5 through 12-7 show the depth flaw, initial depth, and completed depth maps for cases with patch sizes of radius $r_p = 2,\ 3,\ \&\ 4$ *pixels*. The maximum radial patch size for completing the depth map have been $r_f = 11,\ 15,$, and 17 *pixels* respectively. These maps show that the environment objects (books, spoon, cup, and even the background poster) become more identifiable and smoother.

The experimental results show that if a relatively large initial patch size r_p is used then depth map may loose some of its fine details.

12.6 Subsampling the Fixated Images

In this section, we have subsampled each of the fixated images by a factor of 2 before using them for depth recovery. This is done by substituting a patch of 2×2 neighboring pixels with a new pixel whose brightness is an average of 4 initial pixels. This is the smallest symmetric subsampling which can be done on an image. We expect to gain a better depth map because subsampling usually leads to a decrease in noise.

The depth flaw (top), initial depth (middle), and complete depth (bottom) maps for the subsampled image sequence with $r_p = 0$ are shown in fig. 12-8. These maps indicate that some improvements are made by subsampling. This becomes clear if we notice that in the depth flaw map (top of fig. 12-8) there are less regions with unacceptable depths than in the corresponding depth map obtained from images which were not subsampled (fig. 12-1). The initial depth map (middle) is not very informative here. As before, the pixels with unacceptable depths are left blank (white) in the initial depth map. A patch of maximum 4 *pixels* in radius ($r_f = 4$ *pixels*) is

used for completing the initial depth map. Even this completed depth map (bottom of fig. 12-8) offers only a very vague intuition about the boundaries of the objects in the image.

In the next step, we have used a patch of 1 *pixel* in radius ($r_p = 1$) for the depth estimation at any single pixel. The results are shown in fig. 12-9. As expected, the depth flaws have not fully disappeared (top). These points are left blank (white) in the initial depth map (middle). For obtaining the complete depth map (bottom), a patch of maximum 6 *pixels* in radius ($r_f = 6$) is used in this case. Considering the subsampling size of 2×2 *pixels*, these results are located somewhere between the results of nonsampled images with $r_p = 2$, and $r_p = 3$ (figures 12-5, and 12-6).

Figure 12-10 shows the results for the subsampled images for the case with $r_p = 2$ *pixels*, and $r_f = 9$ *pixels*.

A careful observation shows that there are not many differences between sampled and nonsampled results from the point of view of identifying different objects in the environment. However, the depth maps of subsampled images have much better quality and are relatively free from the systematic noise. This is quite clear if we notice that the vertical black lines between the books which were seen in previous depth maps are absent here. These lines represent narrow but deep vertical gaps between the books which did not actually exist in the environment.

Furthermore, due to the printer grey level limitation, quality depth maps cannot be printed out. The computed depth maps are much better than what are shown here. For example each book has its relatively uniform depth which clearly distinguishes it from its neighboring books when there is a depth change in the real environment.

12.7 Summary

This chapter combined the individual results that we had obtained in previous chapters and used them in the recovery of depth maps. The recovered depth maps are

quite good considering that the input to the system was only two unrestricted frames. These images were real and noisy. Furthermore, the motion was not known in advance, and the recovered motion was used in the computations. It is also important to notice that simple computations have been involved in all the steps.

The experimental results show that by subsampling the initial images, much better depth maps are obtained. This is due to the fact that subsampling acts as a low pass filter and eliminates the high frequency noise which is inherent in real images.

An overall study of the experimental results in this chapter shows that depth maps obtained by using an $r_p = 2$ or 3 pixels seem to be a good choice. This is probably because of the fact that a mask of 2×2 pixels is used for the computation of gradients. As a result, using smaller r_p will not give a good depth map. On the other hand, using larger r_p's may result in the elimination of some fine details of the depth map and does not improve the overall quality of the depth map.

It should also be pointed out that we do not have any control over choosing r_f. The algorithm automatically chooses an r_f large enough to include pixels with reliable depths in order to find estimates for depths at pixels where depths were missing in the initial depth map.

All the results in this chapter were constructed by using a single r_f for obtaining depth estimate at any pixel point with an unacceptable depth value. An adaptive approach which chooses r_p appropriately at any desired pixel point will result in smother depth maps.

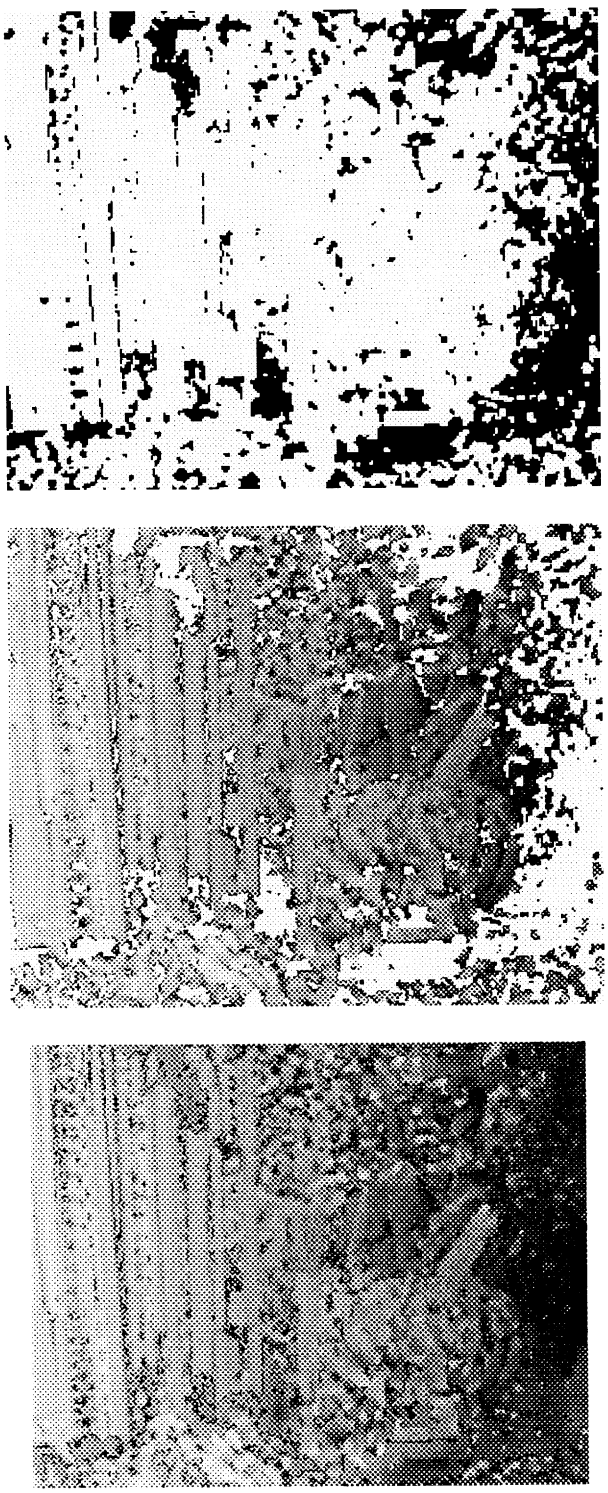

Figure 12-4: The depth flaw (top), initial depth (middle), and completed depth (bottom) maps for the fixated cup image sequence with $r_p = 1$ *pixel*, and $r_f = 9$ *pixels*. The areas close to the viewer look brighter.

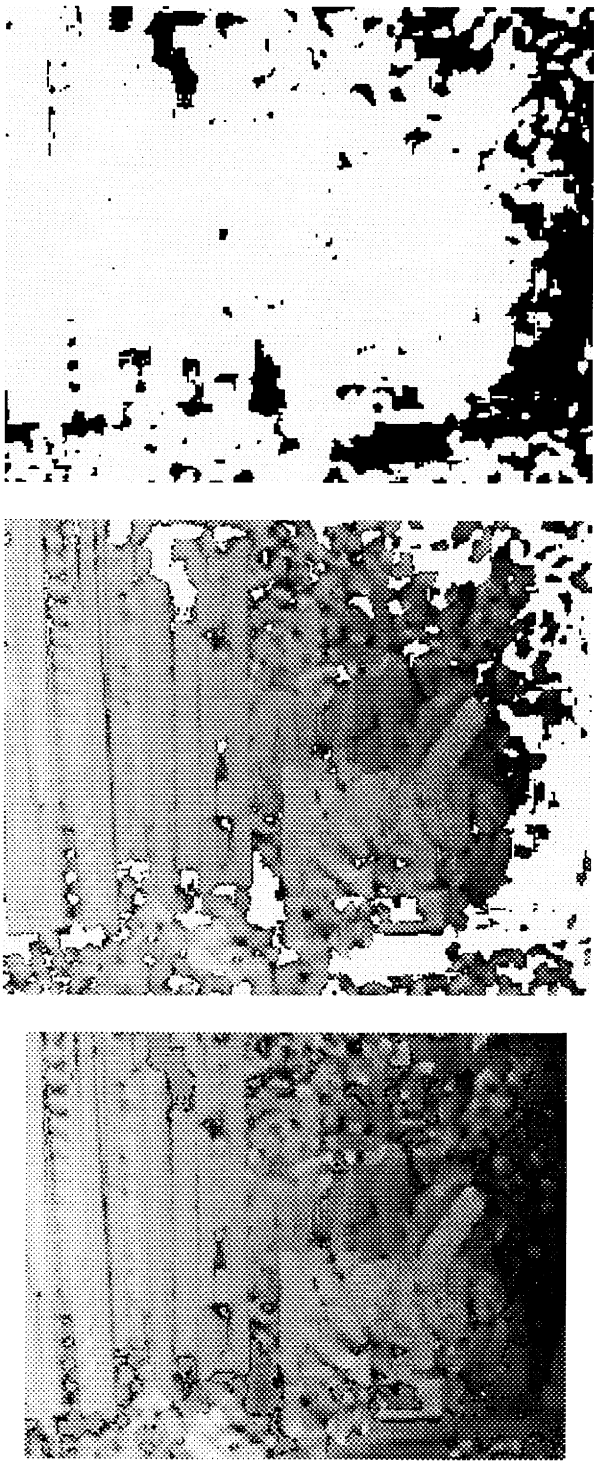

Figure 12-5: The depth flaw (top), initial depth (middle), and completed depth (bottom) maps for the fixated cup image sequence with $r_p = 2$ *pixels*, and $r_f = 11$ *pixels*. The areas close to the viewer are shown brighter.

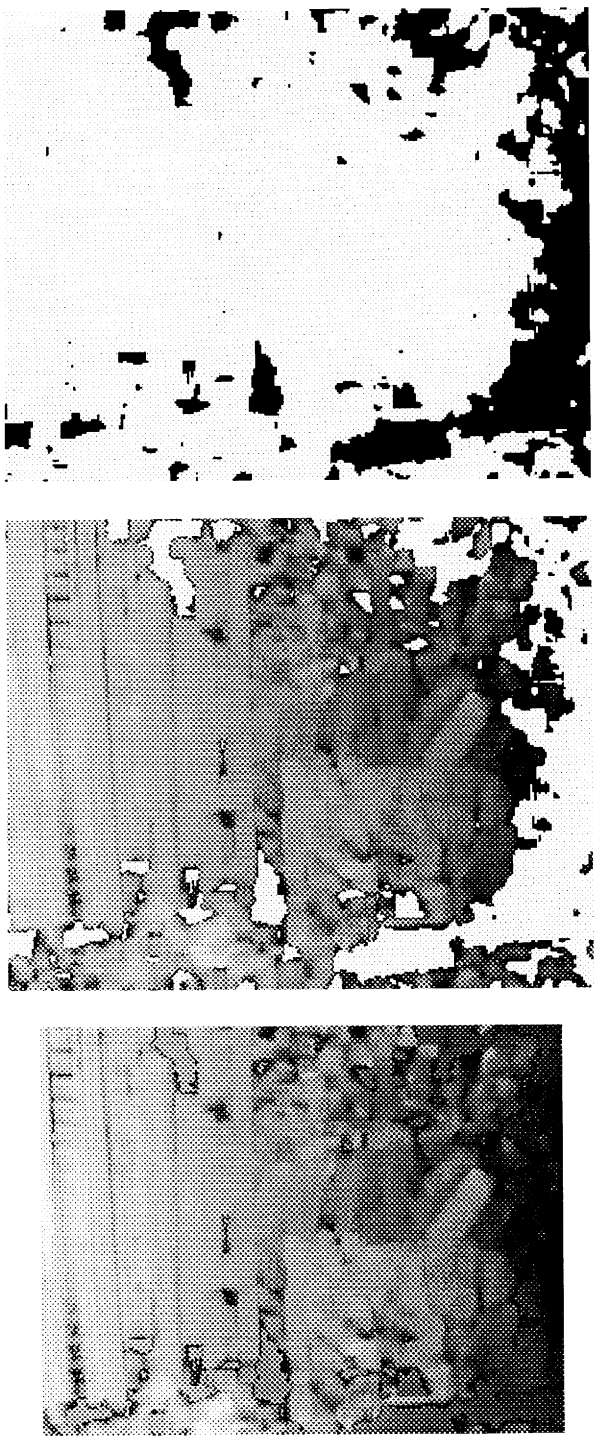

Figure 12-6: The depth flaw (top), initial depth (middle), and completed depth (bottom) maps for the fixated cup image sequence with $r_p = 3$ *pixels*, and $r_f = 15$ *pixels*. The areas close to the viewer look brighter.

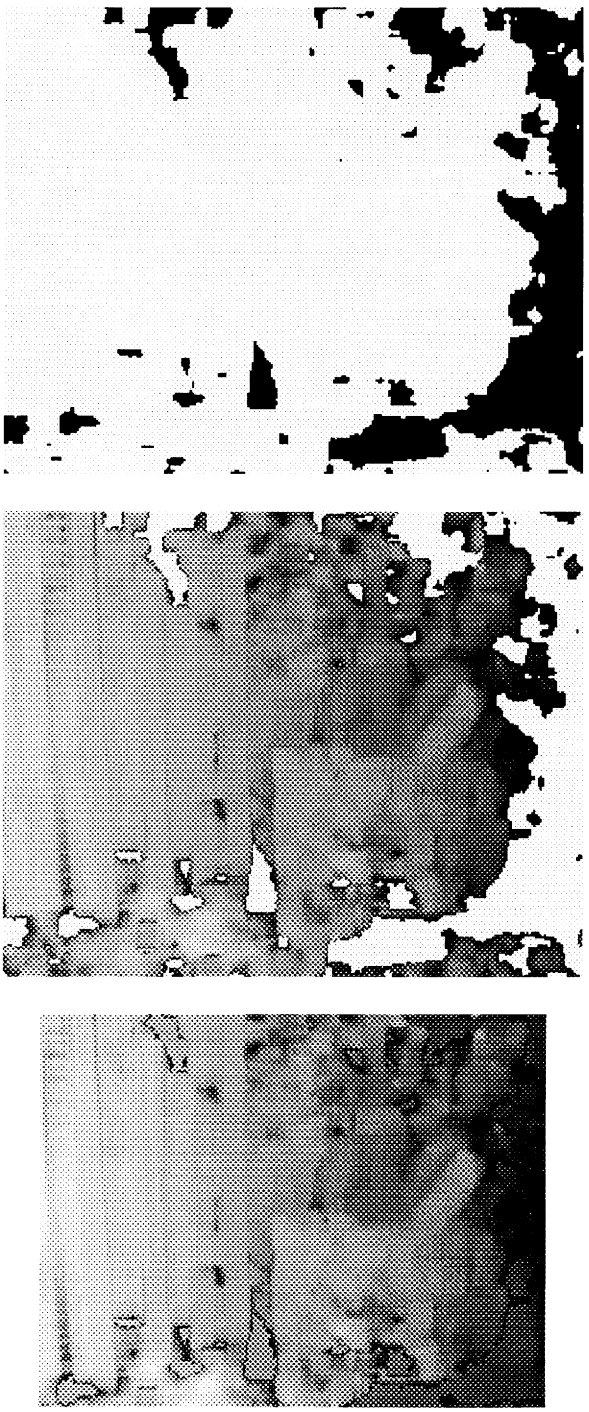

Figure 12-7: The depth flaw (top), initial depth (middle), and completed depth (bottom) maps for the fixated cup image sequence with $r_p = 4$ *pixels*, and $r_f = 17$ *pixels*. The areas close to the viewer look brighter.

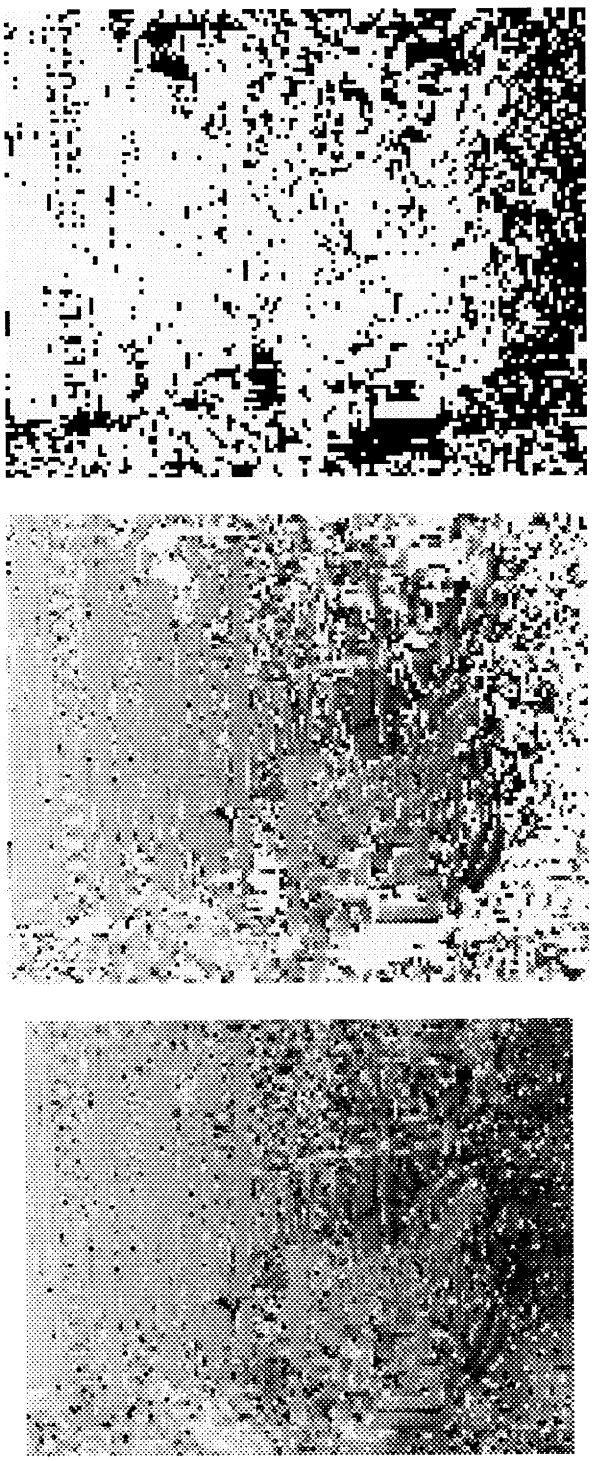

Figure 12-8: The depth flaw (top), initial depth (middle), and completed depth (bottom) maps for the subsampled (by 2) fixated cup image sequence with $r_p = 0$, and $r_f = 4$ *pixels*. The areas close to the viewer look brighter.

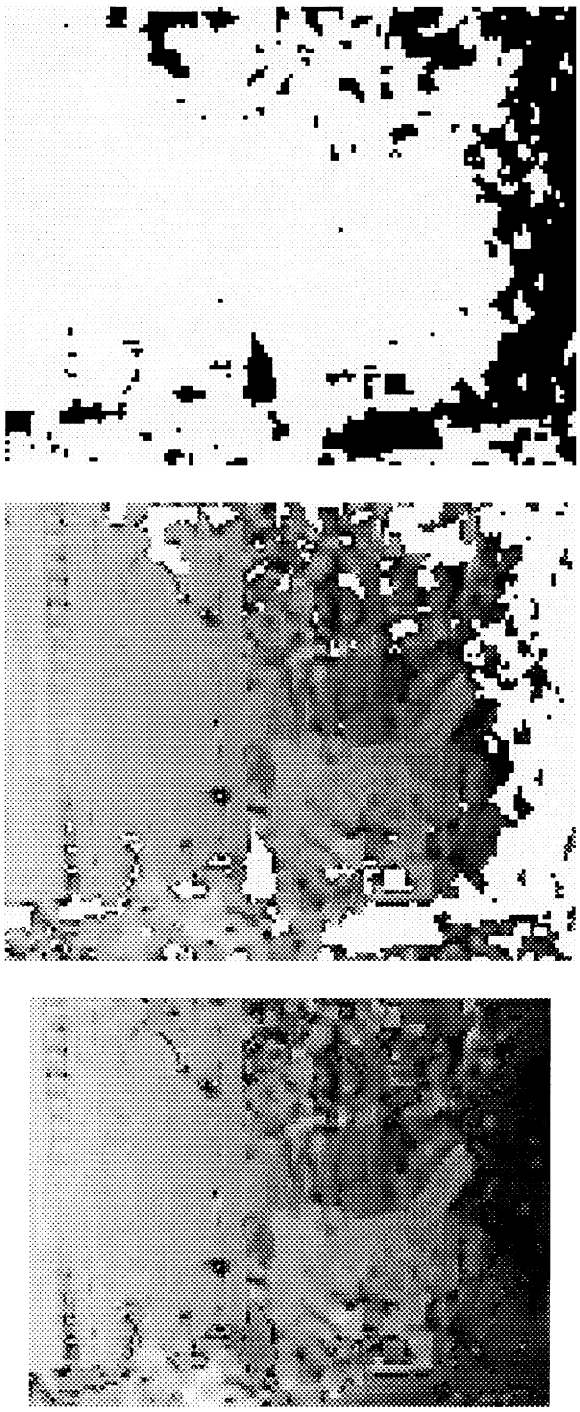

Figure 12-9: The depth flaw (top), initial depth (middle), and completed depth (bottom) maps for the subsampled (by 2) fixated cup image sequence with $r_p = 1$, and $r_f = 6$ *pixels*. The areas close to the viewer look brighter.

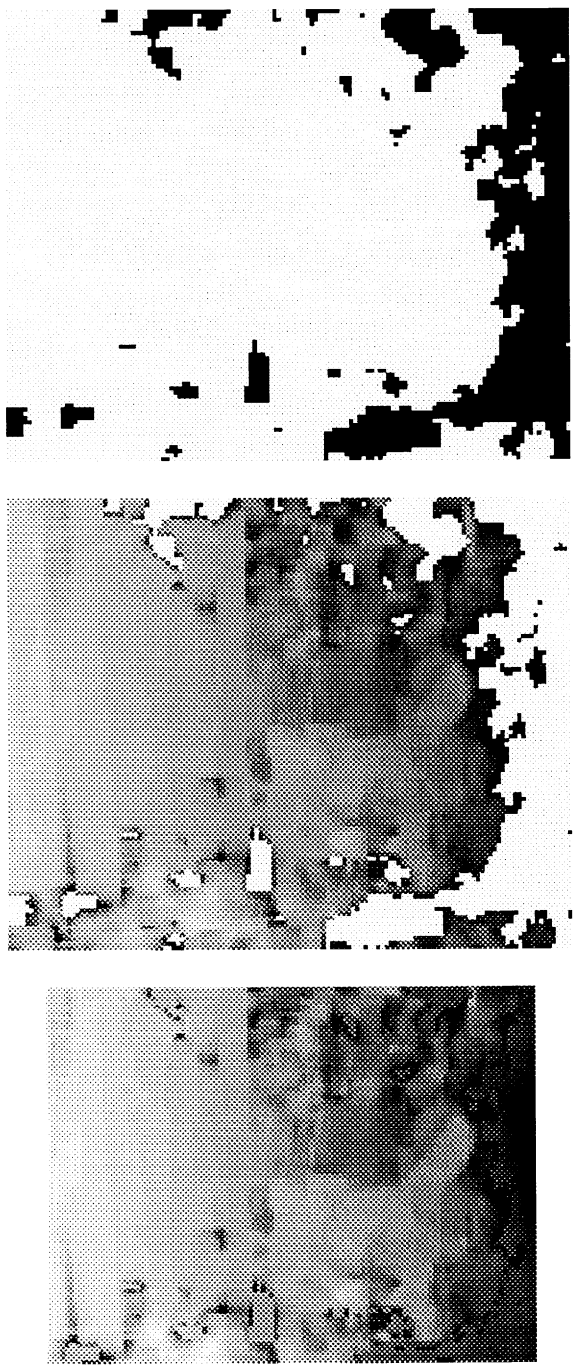

Figure 12-10: The depth flaw (top), initial depth (middle), and completed depth (bottom) maps for the subsampled (by 2) fixated cup image sequence with $r_p = 2$, and $r_f = 9$ *pixels*. The areas close to the viewer look brighter.

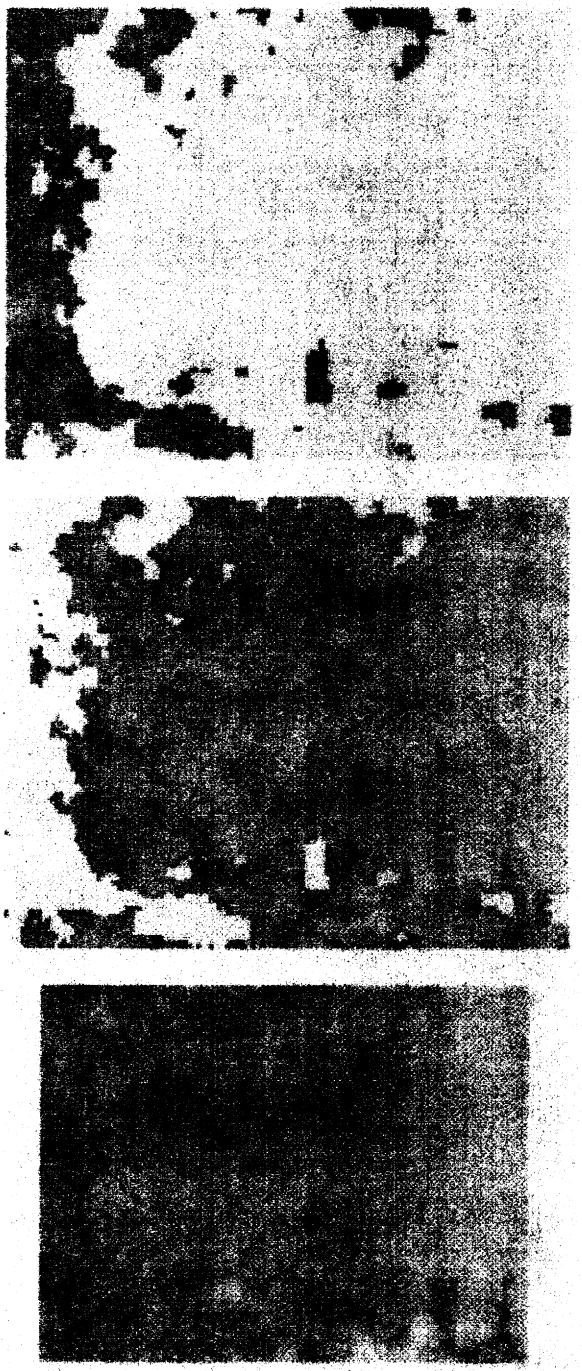

Figure 12-10: The depth flaw (top), initial depth (middle), and completed depth (bottom) maps for the subsampled (by 2) fixated cup image sequence with $r_y = 2$ and $r_y = 9$ pixels. The areas close to the viewer look brighter.

Calibration Issues

Chapter 13

Camera calibration is an important area of research involving the study of techniques for obtaining reliable estimates for the required internal and external parameters of a camera in a vision system.

For many years, computer vision scientists have been working on different aspects of camera calibration problems such as *focal length* (principal distance) [77, 86, 87], *principal point* (image center) [33, 86], *scale factor* (difference between the scanning frequency of the camera sensor plane and the scanning frequency of the image capturing board frame buffer) [33, 47], *intrinsic parameters* (camera internal geometric and optical characteristics) [77], *extrinsic parameters* (the 3D position and orientation of the camera coordinate relative to a certain world coordinate system) [77, 85, 87, 18, 16, 86], and the *hand-eye* transform system (the 3D position and orientation of a camera relative to the last joint of a robot manipulator in an eye-on-hand configuration) [78, 79, 12].

In the previous chapters we saw that some parameters such as *focal length* and *principal point* have important role in the formulations. Manufacturers usually give a nominal value for the focal length but this nominal value is not always sufficiently accurate to be used in the computations. Some other important parameters such as the true principal point are not given at all.

In this chapter, some of the calibration techniques used in this work will be described.

13.1 Principal Point Calibration

The *principal point* is where the optical axis intersects the image plane; see fig. 2-1. Ideally, the principal point is located at the center of the image plane. However, in off-the-shelf cameras the principal point is not necessarily located at the center of the image plane. Finding the true location of the principal point is important because those values appear in our algorithms.

For the cup images the nominal image center was used as the principal point because the camera was not accessible to be calibrated. On the other hand, in the case of the landscape images the true principal point was obtained using a *direct optical method* [33].

The experimental results showed that the true principal point was considerably off from the nominal image center. It was located at about 13 pixels to the left and 13 pixels below the nominal image center.

13.1.1 Direct optical method

The *direct optical method* is a very simple and accurate calibration technique for finding the principal point. This method requires only a laser. The lens assembly is used as a reflecting surface and therefore, the lens can remain mounted on the camera.

When a laser beam is pointed at a lens assembly, part of the light is reflected when the beam enters the glass and also when it leaves it. Multiple reflections occur when the beam is reflected within the lens and can be observed on a piece of paper attached to the front of the laser with a small hole for the primary beam. With some experimental skill the laser can be adjusted relative to the lens so that all reflections coincide with the primary beam, indicating that it is aligned with the optical axis. Once aligned, an attenuation filter is placed in the optical path, the camera is turned on and the center of the light spot observed can be used as the image center.

This method is commonly used in experimental optics to align lens assemblies and

gives reproducible results. If the lens is removed, the reflection from the surface of the image sensor will also give an indication of its perpendicularity with respect to the optical axis. When a low power laser ($< 10mW$) is used, no harm is done to a discrete array camera sensor (CCD). However, vidicon tubes might be damaged by burning in.

13.2 Calibration of the Rotation Axis

In the landscape experiments, we did not explicitly apply any vertical translation (along Y axis). However, fig. 8-2 show a considerable vertical translation of about $-0.9\ mm$. This is mainly because the real rotation axis does not pass through the center of projection[1].

To clarify this, we should mention that in motion vision, it is assumed that the rotation axis passes through the origin of the viewer centered coordinate system, i.e the center of projection. But at the *CMU Imaging Laboratory*, the rotation mechanism was not set up to align the Z axis of rotation with the optical axis. The CMU vision system was equipped with several cameras and evidently the camera used for taking the landscape images was set off center. However, for obtaining the experimental results, we have employed algorithms which erroneously assume that the rotation axis passes through the center of projection.

According to the basic kinematics, the compensating translation which results from shifting the rotation axis is given by

$$\mathbf{V}_o = -\omega \times \mathbf{B} \tag{13.1}$$

where B is a vector extending from a point on the real (desired) rotation axis to a point

[1]If the CCD edges are not accurately aligned with the horizontal and vertical axes of the camera frame, i.e. the CCD is mounted at an angle with respect to the camera coordinate system, such kind of errors happen in both vertical and horizontal directions. But it is not the case here because the inaccuracy of motion estimation has occurred only in the vertical direction.

on the assumed rotation axis; see fig. 13-1. In our experiment, $\mathbf{V}_o = -(\omega\hat{z}) \times (b\hat{x})$

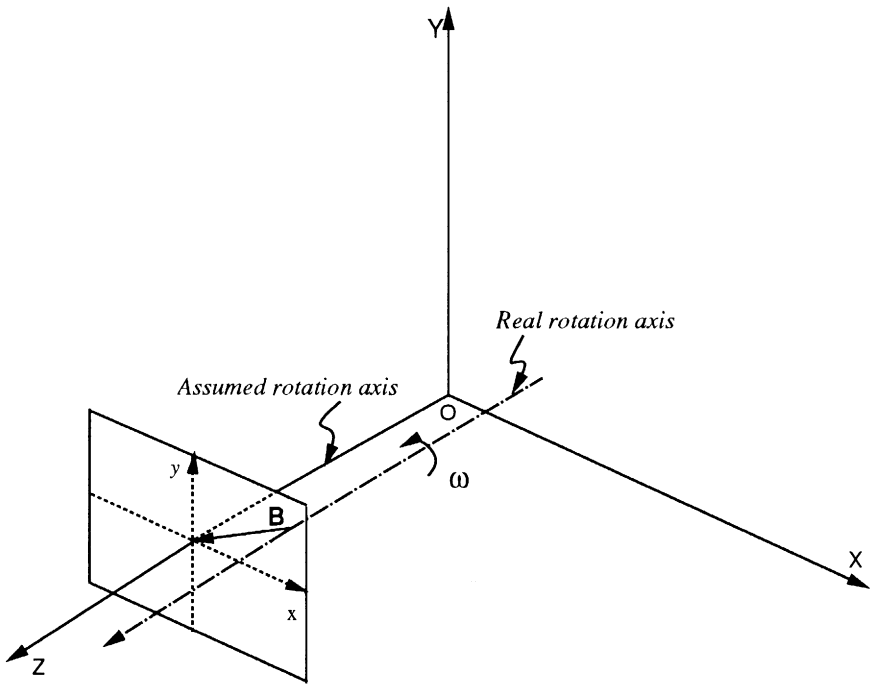

Figure 13-1: In motion vision the assumption is that the rotation axis passes through the center of projection (origin). In the landscape image sequence, the true rotation is parallel to the optical axis but does not pass trough the origin. This will result in a translation which should be compensated for.

where $\mathbf{V}_o = -0.9\hat{y}\ mm$, and $\omega = -0.3\ degree$. As a result, the real rotation axis was located at about $b = -(-0.9)/((-0.3 \times \pi)/180) = -172\ mm$ perpendicular distance from the optical axis in the horizontal plane.

13.2.1 Generalization

A similar method can be used for the calibration of the rotation axis which is parallel to the optical axis in a camera system arrangement in the general case.

In order to find the real location of the rotation axis, the following steps should be taken:

- *Step 1*: Apply a pure rotation about the axis which is supposed to be the optical

axis.

• *Step 2*: If rotation $\omega_{\mathbf{R}_o}$ is not accurately known, compute it by applying eqn. 4.4 to a relatively large patch around the principal point.

• *Step 3*: Estimate the apparent motion (u_o, v_o) at the principal point using the eqn. 8.4 or 4.4.

• *Step 4*: The real location of the rotation axis is given by,

$$\begin{cases} b_x = -\frac{v_o f}{Z_o \omega_{\mathbf{R}_o}} \\ b_y = +\frac{u_o f}{Z_o \omega_{\mathbf{R}_o}} \end{cases} \tag{13.2}$$

where Z_o is depth at the principal point, and f is the focal length of the camera.

Point (b_x, b_y) represents the location where the real rotation axis (which is parallel to the optical axis) intersects the image plane.

13.3 Summary

Focal length, principal point, and the *rotation axis position* are the three most important factors which can effect the computations in our motion vision algorithms.

The experimental results show that we may be able to get away with using the nominal focal length as the focal length, and using the image center as the principal point. However, we have to calibrate the system for finding the real rotation axis and compensate for the resultant translation if the rotation axis does not pass through the projection center. The calibration technique introduced in this chapter offers an easy and reliable solution to this important problem.

axis.

- Step 2: If rotation ω_y is not accurately known, compute it by applying eqn. 4.4 to a relatively large patch around the principal point.
- Step 3: Estimate the apparent motion (v_o, v_o) at the principal point using the eqn. 8.4 or 4.4.
- Step 4: The real location of the rotation axis is given by,

$$\left.\begin{aligned} b_x &= -\frac{v_o Z_o}{\omega_y f} \\ b_y &= +\frac{u_o Z_o}{\omega_y f} \end{aligned}\right\} \tag{13.2}$$

where Z_o is depth at the principal point, and f is the focal length of the camera.

Point (b_x, b_y) represents the location where the real rotation axis (which is parallel to the optical axis) intersects the image plane.

13.3 Summary

Focal length, principal point, and the rotation axis position are the three most important factors which can affect the computations in our motion vision algorithms.

The experimental results show that we may be able to get away with using the nominal focal length as the focal length, and using the image center as the principal point. However, we have to calibrate the system for finding the real rotation axis and compensate for the resultant translation if the rotation axis does not pass through the projection center. The calibration technique introduced in this chapter offers an easy and reliable solution to this important problem.

Conclusions

Chapter 14

This thesis introduced a general motion vision system which takes any sequence of images as its input and recovers the motion and shape without any need to check, choose, and adjust parameters. A complete implementation of this motion vision system has been tested on real images and the critical issues involved in the its autonomous implementation have been studied. This chapter makes some concluding remarks about this fixation based motion vision system.

14.1 Features

• In contrast to previous work done in the area of motion vision, our solutions are general and do not impose any severe restrictions on the motion or the structure of the environment.

• The fixation method uses neither *optical flow* nor *feature correspondence.* Instead, it directly employs the image brightness gradients.

• Our motion vision system neither requires tracked images as input nor uses hardware tracking for obtaining fixated images. Instead, it introduces a *pixel shifting process* for constructing fixated image sequences at any *arbitrary fixation point.* This process is done entirely in software without moving the camera for tracking.

• The fixation method does not restrict the fixation point and virtually any point can be chosen as the fixation point.

• The algorithms and formulations presented in the fixation method are simple

and have been successfully implemented on real images.

14.2 Results

• Good estimations for motion parameters can be obtained using optimum patch sizes (see chapter 8).

• The novel introduction and use of *normalized error* has enabled us to find *optimum patch sizes* which result in good estimates for motion parameters. This technique has been implemented on many real image sequences (see chapter 9).

• The novel *pixel shifting process* for constructing fixated (tracked) images has been successfully tested on several real image sequences (see chapter 11).

• The experimental results in chapter 12 show that good depth maps can be obtained using only two monocular real images. If we use the data from a single pixel for recovering the corresponding depth, the reliable depth map will be sparse. Using the information from several pixels in a surrounding patch for finding the depth at its central point results in a relatively dense map of reliable depths. We can obtain even better results by subsampling the initial images. Subsampling acts as a low pass filter and overcomes some of inherent high frequency noise in real images.

• We may get away with using the nominal focal length and principal point in the fixation formulations, but we have to make sure to calibrate the imaging system for the real rotation axis. The method described in chapter 13 offers a simple solution to this important practical problem which can result in considerable motion estimation errors if it is not detected and compensated for.

• The implementations were done on a Sun SPARCstation IPX using C codes. Despite not using either parallel or optimized programs, the actual run-time for finding the motion parameters and the depth map for an image of 227×280 *pixels* was about a fraction of second and a few seconds respectively.

14.3 Assumptions

• In the process of solving the general motion vision problem and writing the eqn. 3.27, we assumed that motion parameters can be obtained using a small patch around the fixation point. This is a pure geometric assumption and does not place any restrictions on the depth topology. Numerous experimental results in chapter 9 show that *optimum patch* sizes are small enough to justify our assumption.

• This work assumes that there is one rigid motion between the environment and the observer. However, small deviations from rigidity is tolerated by the system because it is treated as noise and the least squares methods finds the best solution which fits the whole data.

14.4 Shortcomings

• The fixation method fails if the fixation point is located at the center of a uniform brightness patch because in such a case, motion will be undetectable. However, we have presented a mechanism for preventing this from happening by introducing an autonomous technique which chooses an appropriate location for the fixation point (see chapter 10).

14.5 Relation to Other Works

• As oppose to other work done in area of direct methods, our fixation technique estimates both the motion and shape for the general case [69, 60].

• In recent years, many *Kalman filter* based techniques have tried to improve the depth estimations over time by using more than two frames [38, 39, 40, 56, 57, 58, 59, 25]. These techniques not only need to know the motion in advance but also require a good initial guess for the depth map in order to converge to a solution. Despite these major advantages of Kalman filter methods, the depth maps recovered by our

fixation method are far more superior compared to those obtained by the Kalman filtering methods even after several iterations [26, 27].

• Recently, Tomasi and Kanade [76, 75] introduced a *feature* based technique for recovering the motion and shape from a sequence of images. Their method is different from our work in the following sense:

- It assumes orthographic projection which handicaps the system when dealing with close by objects.

- It uses *feature correspondence.*

- It requires choosing and tracking many feature points.

- Depth is obtained only at the feature points.

- It is computationally very expensive.

14.6 Future Extensions

• The motion estimates obtained from *fixation method* are quite satisfactory. However, the depth maps may be improved by using more than two image frames in a Kalman filter based system as follows:

- Converting the input images to a sequence of fixated images at a desired fixation point using the *pixel shifting process.*

- Obtaining the motion estimates from the fixation method if it is not known.

- Using the depth map estimates from the fixation method as the initial guess for the Kalman filter system.

Employing such a hybrid system can potentially improve the depth map and accelerate the convergence rate of the Kalman filter.

• The algorithms and formulations in the fixation method are very well suited to parallel implementation. Such an approach overwhelmingly improves the system performance because most of the operations are simple additions and subtractions which are done independently but all over the image.

- Due to their parallel nature, the fixation algorithms can be implemented on a single chip using analog VLSI techniques such as the one by Mead [41]. This seems to be an attractive approach for task specific applications.
- By using segmentation, this work can be extended to multiple motion case.

- Due to their parallel nature, the fixation algorithms can be implemented on a single chip using analog VLSI techniques such as the one by Mead [41]. This seems to be an attractive approach for task specific applications.
- By using segmentation, this work can be extended to multiple motion case.

Derivation of Brightness Change Constraint Equation

Appendix A

The *brightness change constraint equation* (BCCE) relates the change in the image brightness at a point (x, y) to the apparent velocity (u, v) of the brightness pattern at that point in the image. This appendix describes in detail the steps involved in the derivation of the BCCE [30, 54, 29].

Let $E(x, y, t)$ denote the image brightness at time t at the image point (x, y). Then, if $u(x, y)$ and $v(x, y)$ are the x and y components of the apparent velocity at the point, we expect that the brightness will be the same at time $t + \delta t$ at the point $(x + \delta x, y + \delta y)$, where $\delta x = u\delta t$ and $\delta y = v\delta t$. In other words,

$$E(x, y, t) = E(x + u\delta t, y + v\delta t, t + \delta t) \tag{A.1}$$

for small time interval δt. The underlying assumption in writing the eqn. A.1 is slow spatio-temporal variations in lighting which is true for many practical applications.

If brightness varies smoothly with x, y, and t, we can expand the right hand side of the above equation in a Taylor series to obtain

$$E(x, y, t) = E(x, y, t) + \delta x \frac{\partial E}{\partial x} + \delta y \frac{\partial E}{\partial y} + \delta t \frac{\partial E}{\partial t} + \epsilon \tag{A.2}$$

where ϵ includes second- and higher-order terms in δx, δy, and δt. Canceling $E(x, y, t)$,

dividing through by δt, and taking the limit as $\delta t \rightarrow 0$, we obtain

$$\frac{dx}{dt}\frac{\partial E}{\partial x} + \frac{dy}{dt}\frac{\partial E}{\partial y} + \frac{\partial E}{\partial t} = 0, \tag{A.3}$$

which is actually just the expansion of the total derivative of E with respect to time into its partial derivatives, in other words

$$\frac{dE}{dt} = 0. \tag{A.4}$$

Using the abbreviations

$$\begin{cases} x_t &= \frac{dx}{dt} \\ y_t &= \frac{dy}{dt} \end{cases} \tag{A.5}$$

and

$$\begin{cases} E_x &= \frac{\partial E}{\partial x} \\ E_y &= \frac{\partial E}{\partial y} \\ E_t &= \frac{\partial E}{\partial t} \end{cases} \tag{A.6}$$

equation A.3 can be written as

$$E_t + x_t E_x + y_t E_y = 0. \tag{A.7}$$

The above equation is called the *brightness change constraint equation* because it expresses a constraint on the components x_t and y_t of the *apparent velocity* at a point (x, y) in the image.

In appendix B, we will show how the derivatives E_x, E_y, and E_t are estimated at any image point.

Computation of Brightness Gradients

Appendix B

The spatial and temporal derivatives of the image brightnesses are the basic data blocks in the direct methods. This appendix describes the formulations behind the estimation of the brightness gradients in images [30, 29].

The spatial brightness gradients E_x, E_y, and temporal brightness gradient E_t are computed simply by using the first differences of image brightness values on a cubic grid; see fig. B-1.

Using the indices i, j, and k to represent x, y, and time t respectively, the estimates of spatial gradients E_x and E_y are give by:

$$E_x \approx \frac{1}{4\delta x}((E_{i+1,j,k} + E_{i+1,j,k+1} + E_{i+1,j+1,k} + E_{i+1,j+1,k+1}) \\ -(E_{i,j,k} + E_{i,j,k+1} + E_{i,j+1,k} + E_{i,j+1,k+1})), \tag{B.1}$$

and

$$E_y \approx \frac{1}{4\delta y}((E_{i,j+1,k} + E_{i,j+1,k+1} + E_{i+1,j+1,k} + E_{i+1,j+1,k+1}) \\ -(E_{i,j,k} + E_{i,j,k+1} + E_{i+1,j,k} + E_{i+1,j,k+1})), \tag{B.2}$$

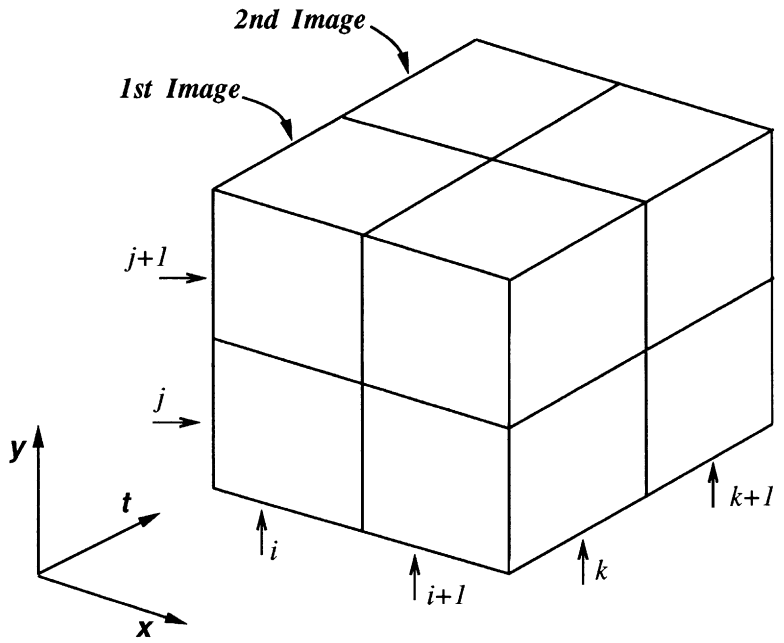

Figure B-1: The first brightness derivatives required in the direct methods can be estimated using first differences in a $2 \times 2 \times 2$ cube of brightness values. The estimates apply to the point where four neighboring pixels in an image meet, and at a time halfway between two successive images.

and the temporal gradient E_t is

$$E_t \approx \frac{1}{4\delta t}((E_{i,j,k+1} + E_{i,j+1,k+1} + E_{i+1,j,k+1} + E_{i+1,j+1,k+1}) - (E_{i,j,k} + E_{i,j+1,k} + E_{i+1,j,k} + E_{i+1,j+1,k})). \quad \text{(B.3)}$$

These formulations give the brightness gradients at a point lying between four neighboring pixels, and between successive images.

Considering the fact that we perform spatial tessellation by using pixels and temporal tessellation by employing individual time varying frames, the above algorithms compensate for part of the tessellation errors involved in discrete digitized images.

Depth at Fixation Point

Appendix C

The results in chapter 3 show that after obtaining the translation $\mathbf{t}$, we need to find Z_o (depth at the fixation point) in order to estimate a depth Z at any point (x, y) in the image plane. This appendix introduces an algorithm for finding the depth Z_o.

At the fixation point, eqn. 3.26 is exactly expanded to

$$E_t + \omega_{\mathbf{R}_o} \mathbf{v}_o \cdot \hat{\mathbf{R}}_o + (\frac{1}{Z_o} - \frac{1}{Z_o})(\mathbf{s}_o \cdot \mathbf{t}) = 0 \tag{C.1}$$

which is similar to eqn. 3.27. Theoretically, all terms of the eqn. C.1 vanish because E_t is zero at the fixation point, and $\mathbf{v} \cdot \mathbf{r} = 0$ applies to all points including the fixation point which means $\mathbf{v}_o \cdot \hat{\mathbf{R}}_o = \frac{\mathbf{v}_o \cdot \mathbf{r}_o}{\|\mathbf{r}_o\|} = 0$. As a result, we cannot directly obtain the depth Z_o from eqn. 3.26. However, at any point i around the fixation point, depth Z_{oi} can be obtained from eqn. 3.26 as

$$Z_{oi} = \frac{1}{E'_{ti}} \left(\frac{\mathbf{v}_i \times \mathbf{r}_o}{\|\mathbf{r}_o\|^2} - \mathbf{s}_i \right) \cdot \mathbf{t}. \tag{C.2}$$

By averaging N of such neighboring depths, we can estimate the depth Z_o as

$$Z_o = \frac{1}{N\|\mathbf{r}_o\|} \mathbf{t} \cdot \sum_{i=1}^{i=N} \left(\frac{\mathbf{v}_i \times \mathbf{r}_o - \|\mathbf{r}_o\|^2 \mathbf{s}_i}{E_{ti}\|\mathbf{r}_o\| + \omega_{\mathbf{R}_o}(\mathbf{v}_i \cdot \mathbf{r}_o)} \right) \tag{C.3}$$

where $\mathbf{s}_i$, $\mathbf{v}_i$, and E_{ti} are computed for N points around the fixation point. In eqn. C.2, it is assumed that $Z_{oi} \approx Z_o$ which is valid considering the averaging in eqn. C.3.

Depth at Fixation Point

Appendix C

The results in chapter 3 show that after obtaining the translation $\mathbf{t}$, we need to find Z_o (depth at the fixation point) in order to estimate a depth Z at any point (x, y) in the image plane. This appendix introduces an algorithm for finding the depth Z_o.

At the fixation point, eqn. 3.26 is exactly expanded to

$$E_o + \omega_{\mathbf{R}} \mathbf{v}_o \cdot \hat{\mathbf{R}}_o + (\frac{1}{Z_o} - \frac{1}{Z_o})(\mathbf{a}_o \cdot \mathbf{t}) = 0 \tag{C.1}$$

which is similar to eqn. 3.27. Theoretically, all terms of the eqn. C.1 vanish because E_i is zero at the fixation point, and $\mathbf{v} = \mathbf{r}$ applies to all points including the fixation point which means $\mathbf{v}_o \cdot \hat{\mathbf{R}}_o = \frac{\mathbf{r}_o^T \hat{\mathbf{r}}_o}{\|\mathbf{r}_o\|} = 0$. As a result, we cannot directly obtain the depth Z_o from eqn. 3.26. However, at any point i around the fixation point, depth Z_{o_i} can be obtained from eqn. 3.26 as

$$Z_{o_i} = \frac{1}{E_i} \left(\frac{\mathbf{v}_i \times \mathbf{r}_o}{\|\mathbf{r}_o\|^2} - s_i \right) \cdot \mathbf{t}. \tag{C.2}$$

By averaging N of such neighboring depths, we can estimate the depth Z_o as

$$Z_o = \frac{1}{N\|\mathbf{r}_o\|} \mathbf{t} \cdot \left(\sum_{i=1}^{i=N} \frac{\mathbf{v}_i \times \mathbf{r}_o - \|\mathbf{r}_o\| s_i}{E_i \|\mathbf{r}_o\| + \omega_{\mathbf{R}} (\mathbf{v}_i \cdot \mathbf{r}_o)} \right) \tag{C.3}$$

where s_i, $\mathbf{v}_i$, and E_i are computed for N points around the fixation point. In eqn. C.2, it is assumed that $Z_{o_i} \approx Z_o$ which is valid considering the averaging in eqn. C.3.

Bibliography

[1] I.E. Abdou and K.Y. Wong. Analysis of linear interpolation schemes for bi-level image applications. *IBM Jour. of Research and Develop.*, 26(6):667–686, Nov. 1982.

[2] G. Adiv. Determining 3-d motion and structure from optical flow generated by several moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(4):384–401, 1985.

[3] J. Aloimonos and A. Basu. Determining the translation of a rigidly moving surface, without correspondence. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, Miami, FL, June 1986.

[4] J. Aloimonos and D. Shulman. *Integration of Visual Modules: An extension of the Marr paradigm.* Academic Press, Boston, 1989.

[5] J. Aloimonos and D.P. Tsakiris. On the mathematics of visual tracking. Technical Report CAR-TR-390, Computer Vision Laboratory, University of Maryland, MD, Sep. 1988.

[6] A.T. Bahil and T. LaRitz. Why can't batters keep their eyes on the ball. *American Scientist*, 72:219–253, 1984.

[7] A.T. Bahil and J.D. McDonald. Model emulates human smooth pursuit system producing zero-latency target tracking. *Biological Cybernetics*, 48:213–222, 1983.

[8] D.H. Ballard and O.A. Kimball. Rigid body motion from depth and optical flow. *Computer Vision, Graphics, and Image Processing*, 22(1), Apr. 1983.

[9] A. Bandopadhay. *A Computational Study of Rigid Motion Perception.* PhD thesis, Dept. of Computer Science, Univ. of Rochester, 1986.

[10] A. Bandopadhay, B. Chandra, and D.H. Ballard. Active navigation: Tracking an environmental point considered beneficial. In *Proc. of IEEE Workshop on*

Motion: Representation and Analysis, pages 23–29, Kiawash Island, May 7-9 1986.

[11] J. Barron. A survey of approaches for determining optical flow, environmental layout and egomotion. Technical Report RBCV-TR-84-5, University of Toronto, Ontario, Canada, Nov. 1984.

[12] D.J. Bennett, D. Geiger, and J.M. Hollerbach. Autonomous robot calibration for hand-eye coordination. *International Journal of Robotics Research*, 10(5):550–559, Oct. 1991.

[13] R. Bernstein. Digital image processing of earth observation sensor data. *IBM Journal of Research and Development*, pages 40–57, Jan. 1976.

[14] A.R. Bruss and B.K.P. Horn. Passive navigation. *Computer Vision, Graphics, and Image Processing*, 21(1):3–20, Jan. 1983.

[15] P.J. Burt, J.R. Bergen, R. Hingorani, R. Kolczynski, W.A. Lee, A. Leung, J. Lubin, and H. Shvaytser. Object tracking with a moving camera. In *IEEE Workshop on Visual Motion*, Mar. 1989.

[16] S.Y. Chen and W.H. Tsai. A systematic approach to analytic determination of camera parameters by line features. *Pattern Recognition*, 23(8):859–877, 1990.

[17] N. Cornelius and T. Kanade. Adapting optical-flow to measure object motion in reflectance and x-ray image sequences. Technical Report CMU-CS-83-119, Carnegie-Mellon University, Pittsburgh, PA, Jan. 1983.

[18] T. Echigo. A camera calibration technique using three sets of parallel lines. *Machine Vision and Applications*, 3:159–167, 1990.

[19] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–326, June 1992.

[20] J.T. Feddema, C.S.G. Lee, and O.R. Mitchell. Automatic selection of image features for visual servoing of a robot manipulator. In *Proceedings of IEEE Int'l Conf. on Robotics and Automation*, pages 832–837, May 1989.

[21] M.A. Gennert and S. Negahdaripour. Relaxing the brightness constancy assumption in computing optical flow. Memo AIM 975, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, June 1987.

[22] A.L. Gilbert, M.K. Giles, G.M. Flachs, R.B. Rogers, and Y.H. U. A real-time video tracking system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(1):47–56, Jan. 1980.

[23] B. Hallert. *Photogrammetry.* McGraw Hill, New York, NY, 1960.

[24] R.C. Harrell, D.C. Slaughter, and P.D. Adsit. A fruit-tracking system for robotic harvesting. *Machine Vision and Applications*, 2:69–80, 1989.

[25] J. Heel. Temporally integrated surface reconstruction. In *Proceedings of the International Conference on Computer Vision*, pages 292–295, Osaka, Japan, Dec. 1990.

[26] J. Heel. Temporal surface reconstruction. Technical Report AI-TR 1296, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Mass., 1991.

[27] J. Heel. Temporal surface reconstruction. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 607–612, Maui, Hawaii, June 1991.

[28] E.C. Hildreth. *The Measurement of Visual Motion.* MIT Press, Cambridge, MA, 1984.

[29] B.K.P. Horn. *Robot Vision.* McGraw Hill and MIT Press, Cambridge, MA, 1986.

[30] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[31] B.K.P. Horn and E.J. Weldon Jr. Direct methods for recovering motion. *Int'l Journal of Computer Vision*, 2:51–76, 1988.

[32] R.G. Keys. Cubic convolution interpolation for digital image processing. *IEEE Trans. Acoustics, Speech and Signal Processing*, 29(6):1153–1160, Dec. 1981.

[33] R.K. Lenz and R.Y. Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3d machine vision metrology. In *Proceedings of IEEE Int'l Conf. on Robotics and Automation*, Raleigh, North Carolina, Mar. 1987.

[34] J.J. Little and A. Verri. Analysis of differential and matching methods for optical flow. Memo AIM 1066, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, Aug. 1988.

[35] H.C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[36] H.C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. In *Proceedings of the Royal Society of London B*, pages 385–397, 1980.

[37] R.C. Luo, R.E. Mullen Jr., and D.E. Wessel. An adaptive robotic tracking system using optical flow. In *Proceedings of IEEE Int'l Conf. on Robotics and Automation*, pages 568–573, 1988.

[38] L. Matthies, R. Szeliski, and T. Kanade. Kalman filer-based algorithms for estimating depth from image sequences. Technical Report CMU-CS-87-185, Computer Science Department, Carnegie Mellon University, Dec. 1987.

[39] L. Matthies, R. Szeliski, and T. Kanade. Incremental estimation of dense depth maps from image sequences. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, Ann Arbor, MI, June 1988.

[40] L. Matthies, R. Szeliski, and T. Kanade. Kalman filer-based algorithms for estimating depth from image sequences. *Int'l Journal of Computer Vision*, 1989.

[41] C. Mead. *Analog VLSI and Neural Systems.* Addison Wesley, 1989.

[42] F. Moffit and E.M. Mikhail. *Photogrammetry.* Harper & Row, New York, NY, 3rd edition, 1980.

[43] S. Moskowitz. Terminal guidance by pattern recognition: A new approach. *IEEE Trans. on Aeros. Navig. Elecron.*, pages 254–265, Dec. 1964.

[44] S. Negahdaripour and B.K.P. Horn. Direct passive navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):168–176, Jan. 1987.

[45] S. Negahdaripour, A. Shokrollahi, and M. Gennert. Relaxing the brightness constancy assumption in computing optical flow. In *Proc. of Int'l Conf. on Image Processing*, pages 806–810, Singapore, Sep. 1989.

[46] N. Papanikolopoulos, P.K. Khosla, and T. Kanade. Vision and control techniques for robotic visual tracking. In *Proceedings of IEEE Int'l Conf. on Robotics and Automation*, pages 857–864, Apr. 1991.

[47] M.A. Penna. Camera calibration: A quick and easy way to determine the scale factor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(12):1240–1245, Dec. 1991.

[48] K. Prazdny. Motion and structure from optical flow. In *Proc. of the Sixth Int'l Joint Conf. on Artificial Intelligence*, Tokyo, Japan, Aug. 1979.

[49] S.S. Rifman and D.M. McKinnon. Evaluation of digital correction techniques for ERTS images. Technical Report TRW 20634-6003-TU-00, NASA Goddard Space Flight Center.

[50] S.S. Rifman and D.M. McKinnon. Evaluation of digital correction techniques for ERTS images. Technical Report E74-10792, TRW Systems Group, July 1974.

[51] A. Rosenfeld. Survey: Image analysis and computer vision: 1991. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 55(3):349–380, May 1992.

[52] G. Sandini and M. Tistarelli. Active tracking strategy for monocular depth inference over multiple frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):13–27, Jan. 1990.

[53] R.J. Schalkoff and E.S. McVey. A model and tracking algorithm for a class of video targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(1):2–10, Jan. 1982.

[54] B.G. Schunck and B.K.P. Horn. Constraints on optical flow computation. In *IEEE Computer Society Conference on Pattern Recognition and Image Processing*, pages 205–210, Aug. 1981.

[55] A. Singh. *Optical Flow Computation: A unified perspective.* IEEE Computer Society Press, Los Alamitos, CA, 1992.

[56] R. Szeliski. Bayesian modeling of uncertainty in low-level vision. Technical Report CMU-CS-88-169, Carnegie Mellon University, 1988.

[57] R. Szeliski. *Bayesian Modeling of Uncertainty in Low-level Vision.* Kluwer Academic Publishers, 1989.

[58] R. Szeliski. Real-time octree generation from rotating objects. Technical Report CRL 90/13, DEC Cambridge Research Laboratory, DEC Cambridge Research Laboratory, Mass., Dec. 1990.

[59] R. Szeliski. Shape from rotation. Technical Report CRL 90/12, DEC Cambridge Research Laboratory, DEC Cambridge Research Laboratory, Mass., Dec. 1990.

[60] M.A. Taalebinezhaad. Direct recovery of motion and shape in the general case by fixation. In *Proc. of IEEE International Conf. on Computer Vision*, Osaka, Japan, Dec. 1990.

[61] M.A. Taalebinezhaad. Direct recovery of motion and shape in the general case by fixation. Memo AIM 1187, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, Mar. 1990.

[62] M.A. Taalebinezhaad. Fixation: A direct method for recovery of motion and shape in the general case. In *Proc. of Image Understanding Workshop*, Pittsburgh, PA, Sep. 1990.

[63] M.A. Taalebinezhaad. Using fixation for direct recovery of motion and shape in the general case. In *Proc. of SPIE Conf. on Mobile Robots V*, volume 1388, Boston, MA, Nov. 1990.

[64] M.A. Taalebinezhaad. Direct robot motion vision by fixation. In *Proc. of IEEE Conf. on Robotics and Automation*, Sacramento, CA, Apr. 1991.

[65] M.A. Taalebinezhaad. Partial implementation of fixation method on real images. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, Maui, Hawaii, June 1991.

[66] M.A. Taalebinezhaad. Autonomous fixation. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 744–747, Champaign, Illinois, Nov. 1992.

[67] M.A. Taalebinezhaad. Autonomous motion vision. In *Proc. of Int'l Conf. on Pattern Recognition*, Hague, Netherlands, Aug. 1992.

[68] M.A. Taalebinezhaad. Autonomous robot motion vision. In *Proc. of Int'l Conf. on Intelligent Robots and Systems*, pages 2161–2166, Raleigh, North Carolina, July 1992. IEEE and RSJ.

[69] M.A. Taalebinezhaad. Direct recovery of motion and shape in the general case by fixation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):847–853, Aug. 1992.

[70] M.A. Taalebinezhaad. *Robot Motion Vision by Fixation.* PhD thesis, Massachusetts Institute of Technology, Cambridge, Mass., Sep. 1992.

[71] M.A. Taalebinezhaad. Shape from 2 monocular real images. Memo AIM 1383, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Mass., Oct. 1992.

[72] M.A. Taalebinezhaad. Towards autonomous motion vision. Memo AIM 1334, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, Apr. 1992.

[73] M.A. Taalebinezhaad. Tracking without moving the camera. Memo AIM 1382, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Mass., Oct. 1992.

[74] W.B. Thompson. Structure-from-motion by tracking occlusion boundaries. *Biological Cybernetics*, 62:113–116, 1989.

[75] C. Tomasi and T. Kanade. Factoring image sequences into shape and motion. In *IEEE Workshop on Visual Motion*, pages 21–28, Princeton, New Jersey, Oct. 1991.

[76] C. Tomasi and T. Kanade. Shape and motion from image streams: A factorization method. Technical Report CMU-CS-91-105, Carnegie Mellon University, Jan. 1991.

[77] R.Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 364–374, Miami Beach, Florida, June 1986.

[78] R.Y. Tsai and R.K. Lenz. A new technique for fully autonomous and efficient 3d robotics hand-eye calibration. In Robert C. Bolles and Bernard Roth, editors, *Robotics Research, The Forth International Symposium*, pages 287–297, Cambridge, Massachusetts, 1988. The MIT Press. (Held in Aug. 1987 at University of California, Santa Cruz).

[79] R.Y. Tsai and R.K. Lenz. Real time versatile robotics hand-eye calibration using 3d machine vision. In *Proceedings of IEEE Int'l Conf. on Robotics and Automation*, pages 554–561, Philadelphia, Pennsylvania, Apr. 1988.

[80] S. Ullman. *The Interpretation of Visual Motion.* MIT Press, Cambridge, MA, 1979.

[81] S. Ullman. The effect of similarity between line segments on the correspondence strength in apparent motion. *Perception*, 9:617–626, 1980.

[82] United Technologies, Adaptive Optics Associates, Cambridge, MA. *Data Sheets for MC4256 and MC6464 Fast Framing Camera*, 1992.

[83] A. Verri and T. Poggio. Motion field and optical flow: Qualitative properties. Memo AIM 917, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, Dec. 1986.

[84] A. Verri and T. Poggio. Against quantitative optical flow. In *Proc. of the First IEEE Int'l Conf. on Computer Vision*, pages 171–180, Washington, DC, 1987.

[85] C.C. Wang. Extrinsic calibration of a vision sensor mounted on a robot. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(2):161–175, Apr. 1992.

[86] L.L. Wang and W.H. Tsai. Computing camera parameters using vanishing-line information from a rectangular parallelepiped. *Machine Vision and Applications*, 3:129–141, 1990.

[87] L.L. Wang and W.H. Tsai. Camera calibration by vanishing line for 3-d computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):370–376, Apr. 1991.

[88] A.M. Waxman and K. Wohn. *Image Flow Theory: A Framework for 3-D Inference from Time-Varying Imagery*, volume 1 of *Advances in Computer Vision*, chapter 3, pages 165–224. Lawrence Erlbaum Assoc., 1988. (Editor: C. Brown).

[89] L.E. Weiss, A.C. Sanderson, and C.P. Neuman. Dynamic sensor-based control of robots with visual feedback. *IEEE Transactions on Robotics and Automation*, 3(5):404–417, Oct. 1987.

[90] Y.H. Yang and M.D. Levine. The background primal sketch: An approach for tracking moving objects. *Machine Vision and Applications*, 5:17–34, 1992.

CS-TR Scanning Project
Document Control Form

Date : 5/5/95

Report # AI-TR-1384

Each of the following should be identified by a checkmark:
Originating Department:

☒ Artificial Intellegence Laboratory (AI)
☐ Laboratory for Computer Science (LCS)

Document Type:

☒ Technical Report (TR) ☐ Technical Memo (TM)
☐ Other:_____

Document Information

Number of pages: 148 (155-images)
Not to include DOD forms, printer instructions, etc... original pages only.

Originals are:
☐ Single-sided or
☒ Double-sided

Intended to be printed as :
☐ Single-sided or
☒ Double-sided

Print type:
☐ Typewriter ☐ Offset Press ☒ Laser Print
☐ InkJet Printer ☐ Unknown ☐ Other:_____

Check each if included with document:

☒ DOD Form (2) ☐ Funding Agent Form ☒ Cover Page
☐ Spine ☐ Printers Notes ☐ Photo negatives
☐ Other: _____

Page Data:

Blank Pages (by page number): 10

Photographs/Tonal Material (by page number): 12, 15, 60-65, 99, 100, 102-105, 109-111, 115-121

Other (note description/page number):

Description : Page Number:

(A) IMAGE MAP (1-6) UN#'ED TITLE, BLANK, ABST, BLANK, ACK, DEDICATION PAGES (7-148) PAGES #'ED 7-148 (149-152) SCANCONTROL, COVER, DOD(2) (153-155) TARGETS (0)

(B) XEROX MARKS ACROSS PAGES 123-END

Scanning Agent Signoff:

Date Received: 5/5/95 Date Scanned: 5/17/95 Date Returned: 5/18/95

Scanning Agent Signature: Michael W. Cook

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1992	technical report

4. TITLE AND SUBTITLE
Robot Motion Vision by Fixation

5. FUNDING NUMBERS
N00014-85-K-0124
N00014-91-J-4038

6. AUTHOR(S)
M. Ali Taalebinezhaad

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Artificial Intelligence Laboratory
545 Technology Square
Cambridge, Massachusetts 02139

8. PERFORMING ORGANIZATION REPORT NUMBER
AI-TR 1384

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
Information Systems
Arlington, Virginia 22217

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AD-A259456

11. SUPPLEMENTARY NOTES
None

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution of this document is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In numerous current and future applications ranging from *autonomous navigation of mobile robots* to *collision avoidance systems for cars*, an imaging system (installed on a moving vehicle) takes $2D$ images of an environment with the *aim* of finding the *motion of the vehicle* (translational and rotational velocities) as well as the *structure of the environment* (shape). In machine vision, this problem is referred to as the *general motion vision problem.*

This thesis introduces a *direct method* called *fixation* for solving this general motion vision problem, arbitrary motion relative to an arbitrary environment. Avoiding *feature correspondence* and *optical flow* has been the motivation behind this *direct method* which uses the spatio-temporal brightness gradients of the images directly.

(continued on back)

14. SUBJECT TERMS (key words)		15. NUMBER OF PAGES
direct motion vision	fixation	148
motion	camera calibration	16. PRICE CODE
normalized error	shape	

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

Block 13 continued:

The *fixation* method results in a linear constraint equation (*Fixation Constraint Equation*) which explicitly expresses the rotational velocity in terms of the translational velocity. The combination of this constraint equation with the *Brightness-Change Constraint Equation* (a fundamental equation which relates the motion to the brightness gradients at any image point) solves the general motion vision problem.

In contrast to previous direct methods, the fixation method does not impose any severe restrictions on the motion or the environment. Moreover, the fixation method neither requires *tracked images* as its input nor uses *tracking* for obtaining *fixated images*. Instead, it introduces a novel technique called the *pixel shifting process* to construct *fixated images* for any arbitrary *fixation point*. This is done entirely in software without any need to move the imaging system for *tracking*.

This fixation method has been successfully tested in the real world environment for the recovery of the motion and shape in the general case. The experimental results are presented and the implementation issues and techniques are discussed.

Scanning Agent Identification Target

Scanning of this document was supported in part by the **Corporation for National Research Initiatives,** using funds from the **Advanced Research Projects Agency** of the **United states Government** under Grant: **MDA972-92-J1029.**

The scanning agent for this project was the **Document Services** department of the **M.I.T Libraries.** Technical support for this project was also provided by the **M.I.T. Laboratory for Computer Sciences.**

Scanned

Date: 5/18/1995

M.I.T. Libraries
Document Services