RECURSIVE ESTIMATION OF 3-D MOTION AND STRUCTURE IN IMAGE SEQUENCES BASED ON MEASUREMENT TRANSFORMATIONS

by

ROBERT MING-YIU CHANN

A thesis submitted to the
Department of Electrical and Computer Engineering
in conformity with the requirements
for the degree of Master of Science

Queen's University
Kingston, Ontario, Canada
September 1994

Copyright © ROBERT MING-YIU CHANN, 1994

Abstract

This thesis examines the problem of estimating the three-dimensional trajectory and structure of a moving rigid object in an image sequence. We have built on a recursive algorithm previously proposed by Broida *et al.* [15], in which the positions of a set of tracked feature points are used as input to an iterated extended Kalman filter. The main emphasis of this thesis is on the development of a hybrid algorithm that incorporates image plane velocity or optical flow measurements into the same recursive framework. It is shown that the incorporation of image velocity information significantly improves trajectory estimation as long as the relative error in velocity measurements is less than about 100%. It is also shown that using image velocity significantly improves the estimator's tracking performance during filter initialization as well as during abrupt changes in the 3-D motion model.

To overcome some practical difficulties with the algorithm proposed by Broida *et al.* , the concept of measurement transformation is also introduced in this thesis. The mathematical formulation of an algorithm based on this idea is presented. Using the two-view motion and structure estimation algorithm previously proposed by Weng *et al.* [91], feature correspondences are transformed into quantities which are used as input to a recursive motion estimator. A comparison between this algorithm and the one developed by Broida *et al.* is made. The relative merits and shortcomings are discussed.

Acknowledgements

I would like to thank my thesis supervisor Dr. Steven Blostein for his financial support, timely technical advice, constant encouragement, and patience. Many thanks to Professor David Fleet for his helpful suggestions on numerous occasions. I am grateful to Mr. Mark Earnshaw who proofread part of my thesis, taught me a lot about LaTeX, and most importantly, took care of all the photocopying while I was in Los Angeles. Thanks also go to Dr. Arthur Toga for letting me use the equipment at the UCLA Laboratory of NeuroImaging to complete my thesis. Last but not least, I thank my parents for giving me a chance to study in Canada and for always supporting me in my decisions.

Contents

Abstract ii

Acknowledgements iii

List of Tables vii

List of Figures viii

1 INTRODUCTION 1

2 BACKGROUND 4

2.1 Image Formation 4

2.1.1 Perspective Projection 4

2.1.2 Spatial Quantization Errors 5

2.2 Previous Approaches to 3-D Motion Estimation from Digital Image Sequences 6

2.2.1 Feature-Based Methods 8

2.2.2 Optical-Flow-Based Methods 14

2.2.3 Direct Methods 16

2.3 Kalman Filtering 17

2.3.1 State-Space Notation 17

2.3.2 Discrete Linear Kalman Filter 18

2.3.3 Application to Nonlinear Problems 20

2.4 Summary 24

3 RECURSIVE ESTIMATION USING OPTICAL FLOW MEASUREMENTS **28**

3.1 Overview of the Broida-Chandrashekhar-Chellappa Algorithm 29
3.1.1 Two Coordinate Systems 29
3.1.2 Motion Models 30
3.1.3 Choosing the State Variables 31
3.1.4 Position Measurements of Feature Points 33
3.2 Optical Flow Measurement Formulation 34
3.2.1 Modifying the BCC Algorithm 34
3.2.2 Optical Flow from Feature Displacement 36
3.3 Implementation Detail 37
3.3.1 Description of the Synthetic Testbed 37
3.3.2 Evaluating Three Suboptimal Filters 38
3.3.3 Simulating Noisy Measurements 39
3.3.4 Measurement Noise Covariance Matrix 40
3.3.5 Measuring the Noise Level 41
3.3.6 Process Noise Covariance Matrix 42
3.3.7 Simulating Uncertainty in Initial State Estimate 42
3.4 Simulation Results 43
3.4.1 Design of Experiments 43
3.4.2 Experiment 1: Filter Evaluation 45
3.4.3 Experiment 2: Adding Optical Flow 55
3.4.4 Experiment 3: Translational Acceleration 65
3.4.5 Experiment 4: Abrupt Change 67
3.4.6 Experiment 5: Feature Displacements 76
3.5 Discussions 96
3.6 Summary 105

4 MEASUREMENT TRANSFORMATION **106**

4.1 Overview of the Weng-Ahuja-Huang Two-View Motion Estimation Algorithm 107
4.2 Recursive Update Formulae 109

4.2.1 Translation Subsystem 110
4.2.2 Rotation Subsystem 114
4.3 Adaptive Estimation of the Measurement Error Covariance Matrix . . 117
4.3.1 Translation Subsystem 118
4.3.2 Rotation Subsystem 121
4.4 Discussions 123
4.5 Summary 123

5 CONCLUSIONS AND FUTURE WORK **125**

A THE JACOBIAN MATRIX **128**

Bibliography **132**

Vita **143**

List of Tables

2.1 Summary of batch 3-D motion estimation techniques 11
2.2 Summary of recursive 3-D motion estimation techniques 13

3.1 Motion parameters for Experiment 1 45
3.2 Motion parameters for Experiment 3 67
3.3 Motion parameters for Experiment 4 76

List of Figures

2.1 Boundaries of the image plane 5
2.2 Feature-based approach 8
2.3 Optical-flow-based approach 15
2.4 Discrete linear Kalman filter block diagram 21
2.5 Extended Kalman filter flow chart 25
2.6 Iterated extended Kalman filter flow chart 26
2.7 Iterated linear filter-smoother flow chart 27

3.1 Camera- and Object-Centred Coordinate Systems 30
3.2 Feature points in object-centred coordinate system 38
3.3 Experiment 1: Object motion in image plane 46
3.4 Experiment 1: Results of Monte Carlo simulations (bias) 47
3.4 Experiment 1: Results of Monte Carlo simulations (bias) 48
3.4 Experiment 1: Results of Monte Carlo simulations (bias) 49
3.4 Experiment 1: Results of Monte Carlo simulations (bias) 50
3.5 Experiment 1: Results of Monte Carlo simulations (variance) 51
3.5 Experiment 1: Results of Monte Carlo simulations (variance) 52
3.5 Experiment 1: Results of Monte Carlo simulations (variance) 53
3.5 Experiment 1: Results of Monte Carlo simulations (variance) 54
3.6 Experiment 2: Results of Monte Carlo simulations (bias) 56
3.6 Experiment 2: Results of Monte Carlo simulations (bias) 57
3.6 Experiment 2: Results of Monte Carlo simulations (bias) 58
3.6 Experiment 2: Results of Monte Carlo simulations (bias) 59
3.7 Experiment 2: Results of Monte Carlo simulations (variance) 60

3.7 Experiment 2: Results of Monte Carlo simulations (variance) 61
3.7 Experiment 2: Results of Monte Carlo simulations (variance) 62
3.7 Experiment 2: Results of Monte Carlo simulations (variance) 63
3.8 Experiment 2: Increasing the number of Monte Carlo trials (bias) . . 64
3.8 Experiment 2: Increasing the number of Monte Carlo trials (bias) . . 65
3.9 Experiment 3: Object motion in image plane 66
3.10 Experiment 3: Results of Monte Carlo simulations (bias) 68
3.10 Experiment 3: Results of Monte Carlo simulations (bias) 69
3.10 Experiment 3: Results of Monte Carlo simulations (bias) 70
3.10 Experiment 3: Results of Monte Carlo simulations (bias) 71
3.11 Experiment 3: Results of Monte Carlo simulations (variance) 72
3.11 Experiment 3: Results of Monte Carlo simulations (variance) 73
3.11 Experiment 3: Results of Monte Carlo simulations (variance) 74
3.11 Experiment 3: Results of Monte Carlo simulations (variance) 75
3.12 Experiment 4: Object motion in image plane 77
3.13 Experiment 4: Results of Monte Carlo simulations (bias) 78
3.13 Experiment 4: Results of Monte Carlo simulations (bias) 79
3.13 Experiment 4: Results of Monte Carlo simulations (bias) 80
3.13 Experiment 4: Results of Monte Carlo simulations (bias) 81
3.14 Experiment 4: Results of Monte Carlo simulations (variance) 82
3.14 Experiment 4: Results of Monte Carlo simulations (variance) 83
3.14 Experiment 4: Results of Monte Carlo simulations (variance) 84
3.14 Experiment 4: Results of Monte Carlo simulations (variance) 85
3.15 Experiment 5: FDM with correlated errors (bias) 87
3.15 Experiment 5: FDM with correlated errors (bias) 88
3.15 Experiment 5: FDM with correlated errors (bias) 89
3.15 Experiment 5: FDM with correlated errors (bias) 90
3.16 Experiment 5: FDM with correlated errors (variance) 91
3.16 Experiment 5: FDM with correlated errors (variance) 92
3.16 Experiment 5: FDM with correlated errors (variance) 93
3.16 Experiment 5: FDM with correlated errors (variance) 94

3.17 Experiment 5: FDM with uncorrelated errors (bias) 97
3.17 Experiment 5: FDM with uncorrelated errors (bias) 98
3.17 Experiment 5: FDM with uncorrelated errors (bias) 99
3.17 Experiment 5: FDM with uncorrelated errors (bias) 100
3.18 Experiment 5: FDM with uncorrelated errors (variance) 101
3.18 Experiment 5: FDM with uncorrelated errors (variance) 102
3.18 Experiment 5: FDM with uncorrelated errors (variance) 103
3.18 Experiment 5: FDM with uncorrelated errors (variance) 104

4.1 Kalman filter with measurement transformation 107
4.2 Relationship between Z_i's and $\mathcal{K}_k$ 114
4.3 Measurement transformation: overall system diagram 117
4.4 Block diagram of method proposed by Heel 124

Chapter 1

INTRODUCTION

Background

Estimating the three-dimensional (3-D) motion of a moving object from a sequence of images is of great importance in computer vision. This subject has attracted a lot of attention in recent years and is a rapidly changing area of research. There exist a variety of real-world applications that require the estimation of 3-D motion. Automatic detection and diagnosis of heart conditions is of immense interest to the medical community and thus the ability to estimate the 3-D motion of the left ventricle from ultrasound images, for example, is of great importance. Real-time obstacle avoidance, which is achievable through the estimation of 3-D motion, is crucial to the successful development of autonomous vehicles. Moreover, 3-D motion information can be used to guide a robotic arm to intercept and retrieve a damaged satellite in space.

The basic problem that we consider in this thesis is how to obtain reliable information on the 3-D motion of a moving object using a single camera. It is assumed that there is only one object in the scene and that the object is rigid. In other words, the 3-D distance between any pair of points on the object does not change with time. It is also assumed that the camera is stationary and that a priori knowledge about the structure of the object is *not* available. We would like to find out where the object is and how fast it is moving. The quantities of interest are the 3-D position, orientation, translational velocity and rotational velocity. To be more precise, we are interested in learning how an object's 3-D position and orientation vary with time.

This information is extremely valuable for motion prediction. However, because of the so-called "scale factor ambiguity", the 3-D position and translational velocity can only be estimated up to a scale factor (see Chapter 2). Fortunately, the rotational motion parameters do not suffer from this problem.

A complete, practical 3-D motion analysis system often consists of two phases. Depending upon the measurement data used, the first phase, typically, computes the optical flow field or establishes the correspondence of discrete features. The second phase then uses these data to determine the motion parameters. The focus of this thesis is on the recovery of motion parameters; therefore, issues such as feature selection, feature extraction, feature correspondence, and accurate computation of optical flow will not be addressed.

Contributions

The main contribution of this thesis is a hybrid feature/flow-based recursive 3-D trajectory and structure estimation algorithm, which, unlike any of the existing approaches, makes use of *both* image plane coordinates of discrete features and image plane velocity information. Through the results of our simulations, we will show that by adding optical flow measurements to a purely featured-based algorithm one can achieve improved performance. A unique feature of these simulations is that three different approximate non-linear filters are used, allowing a direct comparison for the specific purpose of 3-D motion estimation. Furthermore, to illustrate the robustness and advantages of the proposed hybrid algorithm, we have experimented with three different sets of motion parameters. In particular, we have studied the situation in which there is an abrupt change in the motion model.

Although the addition of optical flow measurements is an improvement over the use of feature correspondence alone, the proposed hybrid algorithm is not without shortcomings. Most notably, feature points need to be tracked over an entire image sequence and the size of the state vector is proportional to the number of feature points. In order to eliminate these problems, we present the mathematical formulation of an approach that combines two-view and recursive motion estimation.

Organization

The organization of this thesis is as follows: Chapter 2 surveys recent work done in the area of 3-D motion analysis and offers a brief overview of the Kalman filter and some of its variations. A detailed discussion of the hybrid feature/flow-based 3-D trajectory estimation algorithm, including the experimental results, is given in Chapter 3. Chapter 4 introduces the concept of measurement transformation and presents the mathematical formulation of an algorithm based on this idea. Conclusions will be made in Chapter 5, where suggestions for future research are also given.

Chapter 2

BACKGROUND

The purpose of this chapter is to present relevant background material required for the remainder of this thesis. First, a mathematical model of projecting a 3-D scene onto a 2-D image plane will be presented, followed by a brief discussion on spatial quantization errors. Then we will examine some recent techniques for recovering 3-D rigid-body motion from a sequence of digital images. Emphasis will be on monocular image sequences, such as those taken from a single video camera. Finally, the basic concepts behind recursive estimation and Kalman filtering, which will be applied to 3-D motion estimation in Chapters 3 and 4, will be reviewed.

2.1 Image Formation

2.1.1 Perspective Projection

For our purpose, the pin-hole camera model, combined with the perspective projection, is sufficient for modelling the imaging process. Suppose that a coordinate frame $O_cX_cY_cZ_c$ is attached to the camera, with its origin at the focal point and its Z-axis coinciding with the optical axis. The image plane is parallel to the X–Y plane and is located at $Z = f$, where $f > 0$ is the focal length of the camera. Let $\mathbf{P} = (X, Y, Z)$ be a 3-D point and $\mathbf{p} = (x, y)$ be its projection onto the image plane via the perspective projection. The relationship between $\mathbf{P}$ and $\mathbf{p}$ is described by

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix} = \frac{f}{Z} \begin{bmatrix} X \\ Y \end{bmatrix} + \begin{bmatrix} n_x \\ n_y \end{bmatrix} \tag{2.1}$$

where n_x and n_y represent additive noise terms that are included to account for random errors such as those arising from spatial quantization.

Information about the absolute depth is lost during the imaging process if only a single camera is used. Consider two 3-D points, $\mathbf{P_0} = (X_0, Y_0, Z_0)$ and $\mathbf{P_1} = (cX_0, cY_0, cZ_0)$ where c is any real number. It is easy to see, from (2.1), that the projection of $\mathbf{P_0}$ is indistinguishable from that of $\mathbf{P_1}$ in the absence of noise (*i.e.*, $n_x = n_y = 0$). This is known as the "scale factor ambiguity" in monocular vision [90].

2.1.2 Spatial Quantization Errors

The field of view (FOV) of an imaging system is the angle of the cone of directions encompassed by the scene that is being imaged [39]. Given the field of view and the focal length (f) of a camera, and assuming a square image plane, the boundaries of the "viewing area" (Figure 2.1) can be easily computed as

$$x_{lo} = y_{lo} = -\arctan\left(\frac{FOV}{2}\right) \times f \tag{2.2}$$

$$x_{hi} = y_{hi} = +\arctan\left(\frac{FOV}{2}\right) \times f. \tag{2.3}$$

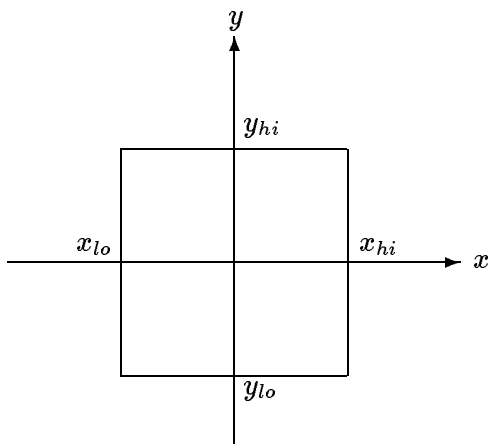


Figure 2.1: Boundaries of the image plane

Due to the finite resolution of the imaging device, it is necessary to convert image plane coordinates (represented by real numbers) to device coordinates (represented by integers). During this conversion, information loss is inevitable. Suppose that

the imaging device has a spatial resolution of $M \times M$ pixels and let $(0,0)$ be the device coordinates of the top-left corner. Spatial quantization occurs when image plane coordinates (x, y) are converted to device coordinates (x_{dev}, y_{dev}) according to

$$x_{dev} = Round\left\{\frac{x - x_{lo}}{x_{hi} - x_{lo}}(M-1)\right\} \tag{2.4}$$

$$y_{dev} = Round\left\{\frac{y_{hi} - y}{y_{hi} - y_{lo}}(M-1)\right\} \tag{2.5}$$

where $Round\{\cdot\}$ denotes the rounding operator. It should be clear that unless $(x_{hi} - x_{lo})$ is an integer multiple of $(x - x_{lo})$, an arbitrary x cannot be recovered from x_{dev}. Though not immediately obvious from (2.4) and (2.5), it is true that the severity of spatial quantization error is inversely proportional to the spatial resolution of the image plane.

2.2 Previous Approaches to 3-D Motion Estimation from Digital Image Sequences

Recovery of 3-D motion parameters from digital image sequences has been an active area of research for the past 15 years. There exist a number of different approaches, depending on the type of sensor technology used: active range sensing, passive range sensing, single video camera, etc.

Both active and passive range sensing systems are capable of producing range images. A range image is a collection of distance measurements from a known reference coordinate system to surface points on object(s) in a scene [10, 82]. An active range sensing system that is based on the principle of active triangulation projects structured light (point, line, grid, etc.) onto an object. An image of the illuminated area is captured using a camera. Since the position of the light source relative to the camera is known, a simple geometrical method called triangulation can be applied to recover the 3-D positions of surface points. Instead of using a light source and a camera, a passive range sensing system relies on two or more cameras to compute 3-D information. Let us consider a binocular (two-camera) configuration. Assuming that a certain number of points in the scene are visible to both cameras and that these

points can be "matched", the same triangulation scheme mentioned above can be used to calculate the 3-D positions of the points. Once the 3-D positions of a set of points are available, from either active or passive range sensing, 3-D rigid-body motion can be recovered from a sequence of range images by existing algorithms [1, 70] provided that point correspondences among consecutive frames have also been established,

Each approach has its own strengths and weaknesses. Absolute depths of points in a scene can be obtained directly from a range image produced by an active range sensor. Unlike a stereo vision system, there is no matching involved. Also, many existing active range sensors can generate dense depth maps by employing scanning devices. However, because of this scanning mechanism, it usually takes a long time to complete a scan of a scene, which results in low temporal resolution or frame rate. It is not uncommon for active range sensors to use lasers which tend to be expensive and consume a lot of power. Furthermore, the use of structured light is often limited to indoors where the illumination in the environment can be controlled.

While the acquisition unit of a passive sensor may cost less than its active counterpart, the processing unit, however, is much more complicated. As previously mentioned, stereo matching must be performed before triangulation can be carried out. This is known to be a difficult problem and almost all of the existing solutions are computationally intensive. In order to operate the system at a reasonable speed, fairly sophisticated processing hardware is required, which will inevitably increase the overall cost. The processing unit in an active range sensing system is less complex in comparison because no processing is required to obtain depth from the range images. Moreover, stereo vision systems, in general, provide depth maps that are less dense than those provided by active range sensing devices. Since range sensing in stereo vision is passive, it does not require structured light and is applicable even in hostile environments.

As one can see, neither active range sensing nor passive range sensing provides perfect data for 3-D motion estimation. Because of this, many researchers have turned to monocular image sequences, those that are taken from a single camera. The major drawback appears to be the scale factor ambiguity that is inherent in monocular vision. However, the 3-D motion estimation algorithm can be designed in such a way

that only a *single* scale factor needs to be estimated over an entire image sequence, as we shall see in Chapter 4. Besides, either an active or a passive range sensing technique may be used for one or two frames at the start of an image sequence to resolve this scale factor ambiguity.

Although a large number of methods have been proposed, all of them can be categorized into one of the three main approaches: feature-based, optical-flow-based, and direct. The following is a brief survey of these main approaches. For an excellent and comprehensive review of this subject area, please refer to [1]. A more recent survey can be found in [35].

2.2.1 Feature-Based Methods

Feature-based methods are based on observing the image plane coordinates of a small number of salient features over two or more image frames. Commonly used features include corners, lines, regions, etc.. It should be noted that the choice of features does affect the formulation of the solution to some extent. A high-level block diagram of an integrated feature-based motion estimation system is depicted in Figure 2.2.

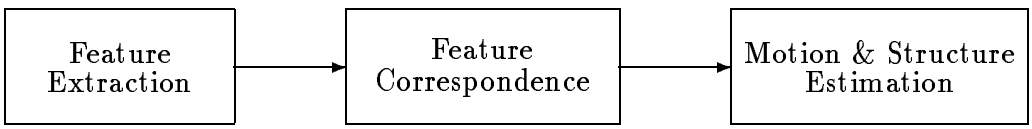


Figure 2.2: Feature-based approach

Given two images of the same scene (taken at two instants in time) and a 2-D feature in the first image, the purpose of the feature correspondence module in Figure 2.2 is to find another 2-D feature in the second image so that both are projections of the same 3-D feature. Obtaining these correspondences accurately is crucial to feature-based approaches but this is a highly non-trivial problem and constitutes an extensive research area on its own. A practical alternative is to use known "markers"

on the object. The problem becomes even more difficult when one tries to establish correspondence among three or more frames. One of the reasons is that features may disappear from one frame to another due to occlusion (*e.g.,* as a result of the object's rotational motion). We will not examine this problem any further in this thesis. The interested reader should consult [5, 11, 22, 66, 71].

Although the feature correspondence problem is far from solved in the general case, many solutions to the 3-D motion estimation problem have been proposed, assuming, of course, that the correspondences of feature points have been established. Feature-based methods can be further divided into two sub-classes: batch and recursive.

Batch Algorithms

Batch algorithms apply one or more constraints (*e.g.,* the object rigidity constraint) in order to arrive at a system of linear or nonlinear equations in terms of the motion parameters. The number of equations must, at least, be equal to the number of motion parameters to be estimated. In the absence of noise, the minimum number of equations suffices but in order to combat noise, the number of equations is usually chosen to be much greater than the minimum. Weng *et al.* [91] proposed an algorithm that requires 8 or more feature points to be tracked over two consecutive image frames. In addition to the 3-D translational direction and rotation matrix, the algorithm also recovers the relative depths of the points being tracked. The number of frames is fixed at 2 and the minimum number of point correspondences is 8. Large feature point displacements are needed for the algorithm to work well. Some argue that the rotation axis used by this algorithm is not "natural" since it must go through the origin of the camera-centred coordinate system or the projection centre, instead of having an arbitrary rotation axis in space. A more detailed description of this algorithm will be given in Chapter 4.

Spetsakis and Aloimonos [78] argued that macrofeatures should be used in favour of microfeatures because macrofeatures are less sensitive to small perturbations resulting from discretization. They proposed a multi-frame approach to motion estimation using line correspondences. They developed a non-iterative, closed-form solution that requires the correspondence of 13 lines in three frames. Essential information about

the motion parameters are captured in three matrices which can be computed directly from the image plane measurements. The elements of these matrices are called the essential parameters. Singular value decomposition (SVD) is used to extract the motion parameters from these matrices. Artificial test data was used in their experiments. The configuration of the lines and the motion parameters were generated randomly for each experiment. They assumed perfect correspondence of all the feature lines but allowed uncertainty in the locations of the lines. The number of frames is fixed at three and the minimum number of line correspondences is 13.

Shariat and Price [73] reported a general motion estimation system that is based on using three or more frames. They assume an unknown natural centre of rotation. They proposed an integrated motion estimation system consisting of a number of components. Observed motion is first classified as one of the following categories: translation dominant, rotation dominant, pure rotation, pure translation and general motion. Errors in the motion parameters can be estimated, together with confidence values associated with the estimated errors. After observing the image plane locations of one feature point over 5 frames, the motion parameters are recovered from feature correspondences by solving a system of mostly second-order polynomials using the iterative Gauss-Newton method. Since this method is highly dependent on the goodness of the initial guess, a module for generating initial guesses automatically has been developed. Different object motions are used to evaluate the performance of the proposed method. Both synthetic and real test data are used. Noisy measurements in the synthetic test sequence were generated by perturbing the true image plane locations with calculated random quantities. Unfortunately, the true motions of the object in their real image sequences are not known. The number of point correspondences is fixed and the minimum number of frames is 5.

Broida and Chellappa [16] and Young and Chellappa [98] have proposed two similar batch solutions to the problem of 3-D motion estimation. They both adopt a nonlinear least-squares formulation that minimizes the distance between the measured and the expected feature point locations. The conjugate gradient descent technique is used to locate the (local) minimum. Young and Chellappa address the problem of occlusion whereas Broida and Chellappa examine the uniqueness of motion parameters

recovered by their method of nonlinear optimization. It is worth pointing out that Broida and Chellappa use both the quaternions and the rotation rates to propagate the rotation matrix in time while Young and Chellappa use only the rotation rates. The significance of this will become clear in Chapter 3. Table 2.1 summarizes the batch algorithms reviewed in this section.

name	method	# frames	# features	type	test data
Broida & Chellappa	nonlinear	≥ 17	≥ 4	point	real
Shariat & Price	nonlinear	5	1	point	synthetic and real
Spetsakis & Aloimonos	linear	3	≥ 13	line	synthetic
Weng *et al.*	linear	2	≥ 8	point	synthetic and real
Young & Chellappa	nonlinear	≥ 3	≥ 4	point	synthetic

Table 2.1: Summary of batch 3-D motion estimation techniques

Recursive Algorithms

In many real-world applications, 3-D motion parameters must be estimated at real-time. Most batch algorithms, especially those that employ iterative search techniques, simply cannot compute estimates fast enough. A recursive algorithm requires only the previous state estimate and the current measurement to produce a current state estimate. Higher computational efficiency (in terms of both execution speed and storage requirements) can be achieved because the amount of data to be processed is reduced significantly. The most commonly used recursive estimator is the Kalman filter (or one of its many variations).

Broida *et al.* [15] developed a recursive formulation using the iterated extended Kalman filter. Specific information that is estimated over time includes the normalized translational velocity, the rotational velocity, and the structure of the object of interest. The unit quaternion is used to represent and propagate the rotation matrix. Broida *et al.* tested their algorithm using both synthetic and real images although it is not clear if the results were obtained from a Monte Carlo analysis. This algorithm

will be examined in more detail in Chapter 3. Wu *et al.* [95] proposed an algorithm similar to the one proposed by Broida *et al.* , using the roll-pitch-yaw expression for rotation matrix representation and propagation. It is not clear from [95], but it appears that a priori knowledge about the configuration of the feature points within the object-centred coordinate system is assumed. The measurement equations are expressed in terms of the object-centred coordinates of the feature points. However, unlike Broida *et al.* , Wu *et al.* do not maintain any structure states and they do not mention how the object-centred coordinates of the feature points are obtained in a general situation. Real imagery is used in their experiment but a pre-determined set of 14 feature points are tracked over only 6 frames. The true motion parameters are known, however, since the motion is generated by a stepper-motor-controlled device. Although it seems that the method proposed by Wu *et al.* uses a state vector smaller than that of Broida *et al.* , it is not at all clear whether a prior knowledge about the feature point configuration is required.

Iu and Wohn [42] proposed a method for estimating the *general* 3-D motion of a rigid body. In their formulation, translation and rotation are modelled as polynomials of arbitrary order (essentially a Taylor series expansion up to the n-th derivative). The problem then becomes one of estimating the polynomial coefficients. As in many other algorithms, the extended Kalman filter is used. Although not clearly stated, it is implicit that an object-centred coordinate system is used in their formulation. One of the advantages of this method is that the measurement function, expressed in terms of the state variables, is very simple. Moreover, this method does not make use of the 3-D orientation of the object, which makes the state vector smaller (compared to [15]). However, the assumption that the centre of rotation is visible does not appear to be realistic. Single-run simulations were performed on synthetic test data, at three different noise levels, with constant translational and rotational accelerations. A Monte Carlo analysis was, apparently, also carried out but the results were not included.

Silvén and Repo [75, 76] have recently developed an integrated monocular visual tracking system, with an emphasis on real-time operation. Their work addressed a number of practical issues, in considerable detail, such as feature extraction and

feature selection. The vision system is divided into four parts: initial modeller, measurement subsystem, feature trackers, and environment model maintainer (the last two are based on the extended Kalman filter). Even though their goal was to estimate the *motion* of a camera, results showing the quality of the velocity estimates, for example, were not included. Most of the experiments were carried out using real image sequence images, in a Monte Carlo fashion. The system is capable of utilizing multiple trackers but in order to achieve real-time performance with available resources, only a single tracker was used. Their preliminary experiments focussed on evaluating the different feature selection schemes and measurement uncertainty modelling methods,

With the help of known navigational landmarks, Chandrashekhar and Chellappa applied recursive state estimation techniques to solve the problem of passive navigation [18]. The EKF is used to estimate the motion parameters and the 3-D locations of feature points, while an "IEKF smoother" was used to provide the necessary initial estimates. Motion estimation and feature correspondence are interleaved, allowing the matching process to take advantage of the Kalman filter's ability to predict feature point locations. Experiments were performed using a 16-frame real outdoor image sequence. Chandrashekhar and Chellappa reported that the EKF converged within 15 frames and that the EKF "seemed to be able to handle model deviations". Table 2.2 summarizes the recursive algorithms reviewed in this section.

name	# frames	# features	type	test data
Broida *et al.*	100	4	point	synthetic and real
Chandrashekhar & Chellappa	16	11	point	real
Iu & Wohn	100	3	point	synthetic
Silvén & Repo	1000	1–6	point	real
Wu *et al.*	6	14	point	real

Table 2.2: Summary of recursive 3-D motion estimation techniques

It is appropriate to mention, as a conclusion to our brief survey of recent feature-based algorithms, that Cui *et al.* [23] have recently proposed a recursive-batch approach to handle long image sequences. A nonlinear optimization procedure is used to estimate the motion parameters of the current frame while estimates of the 3-D structure are maintained and updated through time. One of the advantages is that, unlike other purely batch algorithms, there is no need to store past image frames. Moreover, due to the separation of motion and structure parameters, the size of the search space is reduced. Experimental results based on synthetic data (randomly generated 3-D points) and a real monocular image sequence (consisting of 20 frames) are quite encouraging, indicating substantial improvements over the linear algorithm proposed by Weng *et al.* [91].

2.2.2 Optical-Flow-Based Methods

Optical-flow-based techniques always begin with the computation of an optical flow field $\{\mathbf{u}(x_i, y_i) = [u(x_i, y_i) \quad v(x_i, y_i)]^T\}$. The optical flow at coordinates (x_i, y_i), can be modelled as [39]

$$u(x_i, y_i) = \frac{T_x}{Z_i} - x_i \frac{T_z}{Z_i} - x_i y_i \omega_x + (1 + x_i^2)\omega_y - y_i \omega_z \tag{2.6}$$

$$v(x_i, y_i) = \frac{T_y}{Z_i} - y_i \frac{T_z}{Z_i} - (1 + y_i^2)\omega_x + x_i y_i \omega_y + x_i \omega_z, \tag{2.7}$$

is the instantaneous velocity of the brightness pattern at that point. It is interesting to point out that while feature-based approaches often assume a stationary observer/moving object combination, optical-flow-based approaches tend to deal with situations where an observer is moving within a static environment. A variety of methods exist for computing optical flow. They include differential methods, energy-based methods, and phase-based methods. A review of these methods is out of the scope of this thesis. The interested reader is referred to [6] for a comprehensive survey.

Splitting the optical flow equation algebraically into three separate components is the principal idea behind the subspace methods proposed by Heeger and Jepson [35] recently. To recover the translational direction, the space of all possible translational directions is first sampled and a residual function is computed for each candidate. The one with the minimum value corresponds to the least-squares estimate of the direction

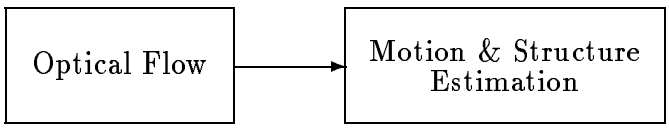


Figure 2.3: Optical-flow-based approach

of translation. Using this estimate, a least-squares estimate of the rotational parameters can be computed quite straightforwardly. Finally, given information about both translation and rotation, the relative depth at each pixel location can be calculated, although they may be quite sensitive to errors in the input optical flow field. Synthetic image sequences and artificial flow fields have been used in their experiments. As an alternative to the above nonlinear method, a linear method for recovering the translational direction have been proposed by Jepson and Heeger [45, 46]. A translation constraint vector, orthogonal to the true translational direction, is computed for each image patch, providing linear constraints which results in a linear equation that can be solved by any standard linear least-squares methods. A significant bias, however, is discovered in the estimate although the source is not clearly identified and the proposed method of removing this bias appears to be suboptimal. Earnshaw and Blostein have recently developed a new technique for removing this bias [27]. Their work also includes an alternative explanation of the bias.

Motivated by Jepson and Heeger, Sundareswaran [83, 84] has developed two independent, global algorithms for computing 3-D motion parameters from optical flow: the flow circulation algorithm for estimating the rotational parameters and the focus-of-expansion (FOE) location algorithm for estimating the direction of translation. Based on the observation that the curl of the flow field is approximately linear, Sundareswaran proposed to recover the rotational velocity by fitting a linear surface to the curl values. A number of regions in the image, within which the curl values are calculated, must be defined. However, no procedure was proposed for selecting the location and size of these regions. Sundareswaran also made the observation that at the focus of expansion the circular component of the flow field is a quadratic polynomial

of a particular form. The algorithm simply checks this condition at every candidate point. One method was proposed to determine whether this condition is satisfied. Two additional methods have been proposed by Hummel and Sundareswaran [40].

2.2.3 Direct Methods

The third category of techniques for 3-D motion estimation attempt to eliminate the need to compute optical flow or establish feature correspondence prior to the actual recovery 3-D motion and structure. Information about 3-D motion is obtained directly from the brightness value at each pixel location.

Taalebinezhaad [85, 87] developed a direct method called fixation which uses brightness gradients directly. Starting with two frames (called the initial 1^{st} image and the initial 2^{nd} image), a fixation point is chosen arbitrarily. Associated with this point are a fixation patch and a fixation velocity. Two small systems of linear equations are solved, giving the necessary parameters for the pixel shifting process that is applied to every pixel in the initial 2^{nd} image. The result of this process is a fixated 2^{nd} image. Finally, a 3×3 matrix is solved for its eigenvalues and eigenvectors. The eigenvector corresponding to the smallest eigenvalue, together with the initial 1^{st} image and the fixated 2^{nd} image, are used to recover the direction of translational velocity, rotational velocity, and relative depths. Selecting a proper fixation point and size of its associated fixation patch appear to be crucial. A detailed discussion of this issue is given in [86]. Experimental results based on real images and a partial implementation have been reported although the motion is restricted to pure translation along the X-axis. More experimentation using general 3-D motion is needed before any conclusive evaluation of this method should be made.

Heel [36] has developed a direct method for estimating the structure of a scene and the motion of an observer relative to the scene from a sequence of images. The Kalman filter is used to incrementally improve the estimates over time. The update stage of the Kalman filter combined the current estimate of depth and the measured depth to provide an updated estimate. The prediction stage involves the geometric transformation (or warping) and resampling of the dense depth map. The approach

taken by Heel is to solve two sub-problems, motion-from-known-depth and depth-from-known-motion, simultaneously in a feedback configuration. Depth values are estimated based solely on image brightness in two frames. His experiment results are based on real images. More results based on real images are given in [37], together with a detailed description of how the error variances are propagated through the Kalman filter structure.

2.3 Kalman Filtering

As we have seen in the previous section, the Kalman filter is used quite extensively in the study of 3-D motion estimation. A detailed treatment of Kalman filtering theory and its extensions to non-linear problems is beyond the scope of this thesis. The reader is referred to [33, 44, 61, 62] as well as [63, 77]. A general overview, however, is necessary as some of the basic principles are used in the remaining chapters of this thesis. Attempts have been made to adopt the notation used in [44] whenever possible.

2.3.1 State-Space Notation

Suppose that we have a physical system which we can describe using a vector differential equation of the following form:

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{f}(\mathbf{x}(t), t) + \mathbf{w}(t) \tag{2.8}$$

where $\mathbf{x}(t)$ is the state of the system and $\mathbf{w}(t)$ represents the process noise. Information about the state of the system is normally obtained by making various measurements and the measurement process can be described, mathematically, by the following equation:

$$\mathbf{y}(t) = \mathbf{h}(\mathbf{x}(t), t) + \mathbf{v}(t) \tag{2.9}$$

where $\mathbf{y}(t)$ denotes the measurement and $\mathbf{v}(t)$ is the measurement noise. The above is commonly known as the state-space representation of a dynamical system in control theory.

Let us first concentrate on the special case where both $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ are linear functions of the state variables. The *plant equation* (2.8) can be rewritten as

$$\frac{d\mathbf{x}(t)}{dt} = F(t)\mathbf{x}(t) + \mathbf{w}(t). \tag{2.10}$$

Before any processing can take place on a digital computer, one needs to discretize (2.10), which leads to

$$\mathbf{x}_{k+1} = \Phi(k+1,k)\,\mathbf{x}_k + \mathbf{w}_{k+1}, \qquad \mathbf{w}_k \sim \mathcal{N}(0, Q_k) \tag{2.11}$$

where $\Phi(k+1,k)$ is the state transition matrix given by

$$\Phi(k+1,k) = \exp[(t_{k+1} - t_k)F_k]. \tag{2.12}$$

In practice, measurements are made at discrete time intervals; therefore, the *measurement equation* (2.9) can also be rewritten as

$$\mathbf{y}_k = H_k\,\mathbf{x}_k + \mathbf{v}_k, \qquad \mathbf{v}_k \sim \mathcal{N}(0, R_k). \tag{2.13}$$

In order to simplify the notation, the explicit dependencies of the various quantities on time are not always shown. For example, $\mathbf{x}(t_k)$ is often simplified to $\mathbf{x}(k)$. Moreover, the subscript and parentheses notations will be used interchangeably, *e.g.,* x_k and $x(k)$ are equivalent.

We now seek a method for estimating the state vector in a recursive manner. Or, more precisely, a scheme that computes $\hat{\mathbf{x}}_k$, an estimate of $\mathbf{x}_k$, based only on (a) the current measurements $\mathbf{y}_k$, (b) the previous state estimate $\hat{\mathbf{x}}_{k-1}$, (c) a priori knowledge about $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$, and (d) the noise statistics of $\mathbf{w}_k$ and $\mathbf{v}_k$. Previous estimates $(\hat{\mathbf{x}}_0, \cdots, \hat{\mathbf{x}}_{k-2})$ and measurements $(\mathbf{y}_0, \cdots, \mathbf{y}_{k-1})$ are not required.

2.3.2 Discrete Linear Kalman Filter

The discrete linear Kalman filter offers an elegant solution to the above problem. Quoting Chui[21], "the Kalman filter is a linear, unbiased, and minimum error variance recursive algorithm that estimates, optimally, the unknown state of a dynamical system from noisy data taken at discrete time intervals".

If we define the error in the state estimate as

$$\delta_{\hat{\mathbf{x}}_k} \quad = \quad \hat{\mathbf{x}}_k - \mathbf{x}_k \tag{2.14}$$

where $\mathbf{x}_k$ is the true system state at time t_k, then the error covariance matrix is simply

$$P(k) \quad = \quad E\{\delta_{\hat{\mathbf{x}}_k} \delta_{\hat{\mathbf{x}}_k}^T\}. \tag{2.15}$$

Since the true state of the system is never known (after all, it is what we are trying to estimate), the Kalman filter can only provide an estimate of this error covariance matrix. However, such an estimate is still very useful for one can utilize it to speculate about the quality of the state estimate.

Each cycle of the linear Kalman filter consists of a time update, followed by a measurement update. The time update phase (or sometimes called the prediction phase) is quite straightforward. One only needs to propagate the state estimate and the error covariance matrix forward in time using the system dynamics, *i.e.,*

$$\hat{\mathbf{x}}(k+1|k) \quad = \quad \Phi(k+1,k)\,\hat{\mathbf{x}}(k|k) \tag{2.16}$$

$$\hat{P}(k+1|k) \quad = \quad \Phi(k+1,k)\,\hat{P}(k|k)\,\Phi^T(k+1,k) + Q_{k+1}. \tag{2.17}$$

On the other hand, the measurement update phase (or simply called the update phase) is slightly more complicated. Let us first try to understand it at an intuitive level. In (2.16), we have already predicted what the state estimate should be at the current time step. Using this information, we can predict the current measurement as well. The difference between the actual measurement $\mathbf{y}_{k+1}$ and the predicted measurement $H_{k+1}\,\hat{\mathbf{x}}(k+1|k)$ is called the residual error or innovation:

$$\mathbf{r}_{k+1} \quad = \quad \mathbf{y}_{k+1} - H_{k+1}\,\hat{\mathbf{x}}(k+1|k). \tag{2.18}$$

The current state estimate is simply a combination of the previous state estimate and the residual error:

$$\hat{\mathbf{x}}(k+1|k+1) \quad = \quad \hat{\mathbf{x}}(k+1|k) + K_{k+1}\mathbf{r}_{k+1}. \tag{2.19}$$

where K_{k+1}, the Kalman gain, controls how much the residual error contributes to the state estimate. This is a good place to remind the reader that $\hat{\mathbf{x}}(k|k)$ is equivalent

to $\hat{\mathbf{x}}(t_k|t_k)$. More formally, the measurement update equations can be summarized as follows:

$$K_{k+1} = \hat{P}(k+1|k)H_{k+1}^T[H_{k+1}\hat{P}(k+1|k)H_{k+1}^T + R_{k+1}]^{-1} \tag{2.20}$$

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + K_{k+1}[\mathbf{y}_{k+1} - H_{k+1}\,\hat{\mathbf{x}}(k+1|k)] \tag{2.21}$$

$$\begin{aligned}\hat{P}(k+1|k+1) &= [I - K_{k+1}H_{k+1}]\hat{P}(k+1|k) \\ &= [I - K_{k+1}H_{k+1}]\hat{P}(k+1|k)[I - K_{k+1}H_{k+1}]^T \\ &\quad + K_{k+1}R_{k+1}K_{k+1}^T.\end{aligned} \tag{2.22}$$

Although the two forms in (2.22) are equivalent, we elect to use the latter because it ensures the positive definiteness and symmetry of the error covariance matrix and it is insensitive to errors in the filter gain [44]. A high-level block diagram of the discrete linear Kalman filter is shown in Figure 2.4.

In the formulation of the linear Kalman filter, the process noise and the measurement noise are assumed to be temporally white, that is,

$$E\{\mathbf{w}_i\mathbf{w}_j^T\} = \begin{cases} Q_k, & \text{i} = \text{j} \\ 0, & \text{i} \neq \text{j} \end{cases} \tag{2.23}$$

$$E\{\mathbf{v}_i\mathbf{v}_j^T\} = \begin{cases} R_k, & \text{i} = \text{j} \\ 0, & \text{i} \neq \text{j}. \end{cases} \tag{2.24}$$

Without loss of generality, $\mathbf{w}$ and $\mathbf{v}$ are often assumed to be mutually uncorrelated as well:

$$E\{\mathbf{w}_i\mathbf{v}_j^T\} = 0, \quad \forall \text{ i,j}. \tag{2.25}$$

2.3.3 Application to Nonlinear Problems

The equations outlined in the previous section only apply when both the plant and measurement functions are linear. When the plant and/or measurement functions are

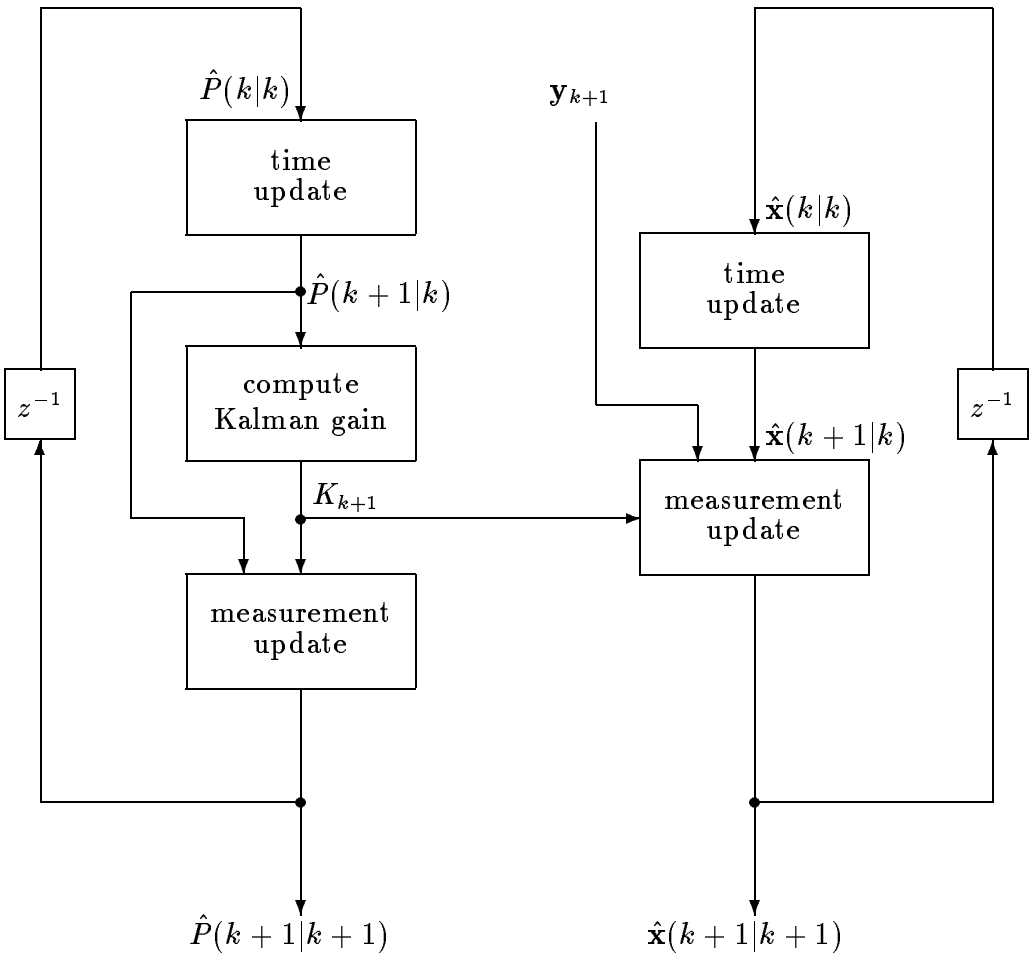


Figure 2.4: Discrete linear Kalman filter block diagram

.

non-linear functions of the state variables, however, one cannot use the linear Kalman filter. Fortunately, as long as we are willing to sacrifice the optimality achieved by the Kalman filter, we are able to retain the recursive structure. The key idea is to linearize the dynamics and/or the measurements so that linear Kalman filter equations can be applied but the resulting filter will become sub-optimal. We will begin by reviewing the extended Kalman filter (EKF).

Extended Kalman Filter

Linearized plant and measurement equations are obtained by applying the Taylor series expansion and truncating after the first two terms. It is important to point out the difference between the linearized Kalman filter and the extended Kalman filter. The former is linearized about some predetermined *reference trajectory*, independent of the measurement data, whereas the latter is linearized about the latest state estimate. Figure 2.5 shows a flow chart of the EKF algorithm.

The extended Kalman filter equations are essentially the same as those of the linear Kalman filter except for a few minor changes. The equation for the forward propagation of the state estimate, (2.16), will be replaced by

$$\hat{\mathbf{x}}(k+1|k) = \hat{\mathbf{x}}(k|k) + \int_{t_k}^{t_{k+1}} \mathbf{f}(\mathbf{x}, \tau) d\tau \tag{2.26}$$

and the measurement update equation for the state estimate, (2.21), will be replaced by

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + K_{k+1}\{\mathbf{y}_{k+1} - \mathbf{h}[\hat{\mathbf{x}}(k+1|k)]\}. \tag{2.27}$$

Furthermore, the Jacobian of $\mathbf{f}(\cdot)$ with respect to the state vector will take the place of F_k in (2.12):

$$F_k = \left.\frac{d\mathbf{f}(\mathbf{x}_k, k)}{d\mathbf{x}_k}\right|_{\mathbf{x}_k = \hat{\mathbf{x}}_k} \tag{2.28}$$

and, finally, in (2.20), (2.21), and (2.22) H_k will be replaced by the Jacobian of $\mathbf{h}(\cdot)$ with respect to the state vector:

$$H_k = \left.\frac{d\mathbf{h}[\mathbf{x}_k, k]}{d\mathbf{x}_k}\right|_{\mathbf{x}_k = \hat{\mathbf{x}}_k}. \tag{2.29}$$

Iterated Extended Kalman Filter

As we have seen, only one linearization is performed per observation in the EKF. This may not be sufficient if significant non-linearities exist in the measurement function. Assuming that the filtered estimates, $\hat{\mathbf{x}}(k+1|k+1)$, are, in general, better than the predicted estimates, $\hat{\mathbf{x}}(k+1|k)$, one should be able to achieve better results by

repeatedly re-linearizing the measurement function about the latest estimate. This iteration continues until a predefined condition is met. This is the basic idea behind the iterated extended Kalman filter (IEKF), which is obtained by simply replacing (2.27) in the EKF with the following iterator [44]:

$$\eta_{i+1} = \hat{\mathbf{x}}(k+1|k) + K_{k+1}\{\mathbf{y}_{k+1} - \mathbf{h}(\eta_i) - H_{k+1}[\hat{\mathbf{x}}(k+1|k) - \eta_i]\} \quad i = 0, \ldots, l. \quad (2.30)$$

The initial and final values are $\eta_0 = \hat{\mathbf{x}}(k+1|k)$ and $\hat{\mathbf{x}}(k+1|k+1) = \eta_l$, respectively. Figure 2.6 shows a flow chart of the IEKF algorithm. As we can see, the EKF is simply a special case of the IEKF. When $l = 1$ the IEKF "reduces" to the EKF.

Iterated Linear Filter-Smoother

Having seen the IEKF, the logical extension is to include the linearization of the plant equation in the iteration. When non-linearities exist in the system dynamics forward propagation of the error covariance matrix (*i.e.,* $\hat{P}(k|k) \rightarrow \hat{P}(k+1|k)$) may be unreliable. The basic idea is to include a backward prediction or smoothing step in the iteration. With information up to time t_{k+1}, we ought to be able to smooth back to time t_k and obtain a "better" estimate, *i.e.,* $\hat{\mathbf{x}}(k|k+1)$. Using this estimate we continue with the iteration until the termination condition is met. We now have the so-called iterated linear filter-smoother (ILFS). Figure 2.7 shows a flow chart of the ILFS algorithm and the ILFS iterator equations are given below [44]:

$$\hat{\mathbf{x}}(k+1|k) = \xi_i + \int_{t_k}^{t_{k+1}} f(\overline{x}, \tau)d\tau + \Phi(k+1,k)(\hat{\mathbf{x}}(k|k) - \xi_i) \quad (2.31)$$

$$\hat{P}(k+1|k) = \Phi(k+1,k)\hat{P}(k|k)\Phi^T(k+1,k) + Q_{k+1} \quad (2.32)$$

$$K_{k+1} = \hat{P}(k+1|k)H_{k+1}^T[H_{k+1}\hat{P}(k+1|k)H_{k+1}^T + R_{k+1}]^{-1} \quad (2.33)$$

$$\eta_{i+1} = \hat{\mathbf{x}}(k+1|k) + K_{k+1}[\mathbf{y}_{k+1} - \mathbf{h}(\eta_i) - H_{k+1}(\hat{\mathbf{x}}(k+1|k) - \eta_i)] \quad i = 0, \ldots, l \quad (2.34)$$

$$\xi_{i+1} = \hat{\mathbf{x}}(k|k) + S(k)[\eta_{i+1} - \hat{\mathbf{x}}(k+1|k)], \qquad i = 0, \ldots, l \quad (2.35)$$

$$S(k) \quad = \quad P(k|k)\Phi^T(k+1,k)\hat{P}^{-1}(k+1|k). \tag{2.36}$$

The reference trajectory $\overline{x}$, in this case, is determined by the state estimate as a result of the latest iteration. Again, we have opted for notational simplicity and omitted the explicit dependencies of the various quantities on ξ_i and/or η_i. The initial values are $\eta_0 = \hat{\mathbf{x}}(k+1|k)$ and $\xi_0 = \hat{\mathbf{x}}(k|k)$ and the final values are

$$\hat{\mathbf{x}}(k+1|k+1) \quad = \quad \eta_l \tag{2.37}$$

$$\begin{aligned} P(k+1|k+1) = [I - K_{k+1}H_{k+1}]P(k+1|k)[I - K_{k+1}H_{k+1}]^T \\ + K_{k+1}R_{k+1}K_{k+1}^T. \end{aligned} \tag{2.38}$$

2.4 Summary

In this chapter, we looked at a mathematical model of projecting a 3-D scene onto a 2-D image plane. Then we examined some recent techniques for recovering 3-D motion from an image sequence. The basic concepts behind recursive estimation and Kalman filtering were also discussed.

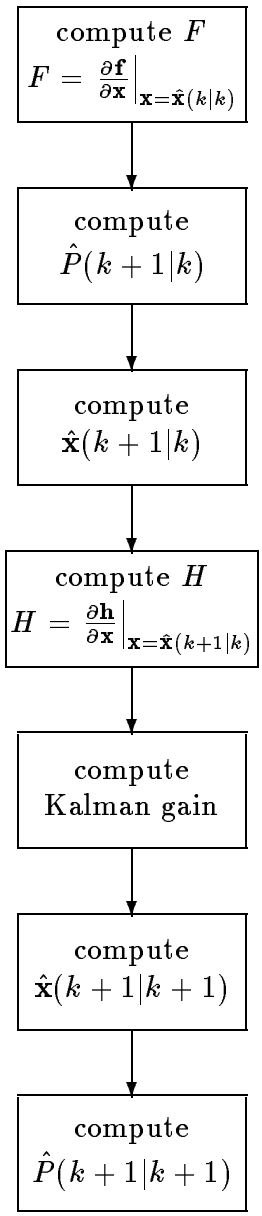


Figure 2.5: Extended Kalman filter flow chart

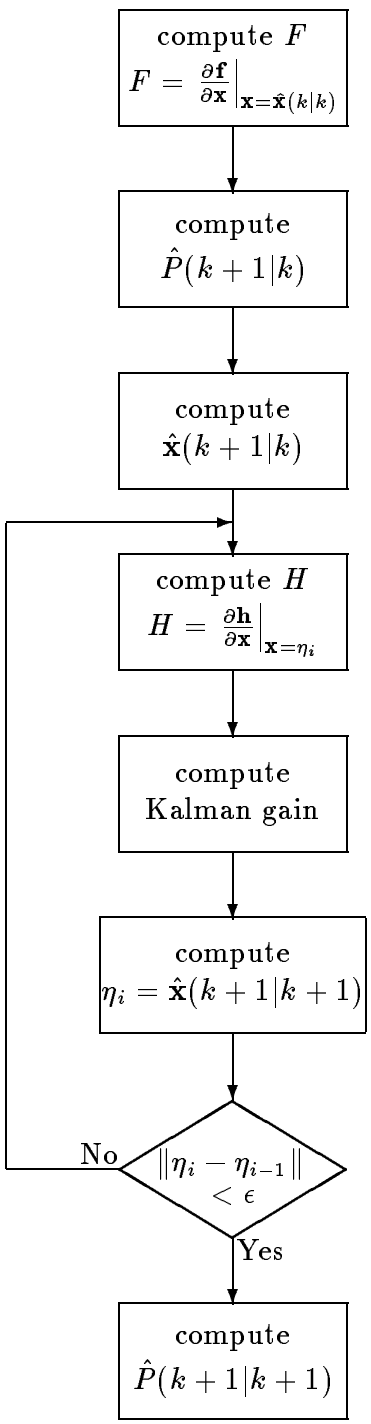


Figure 2.6: Iterated extended Kalman filter flow chart

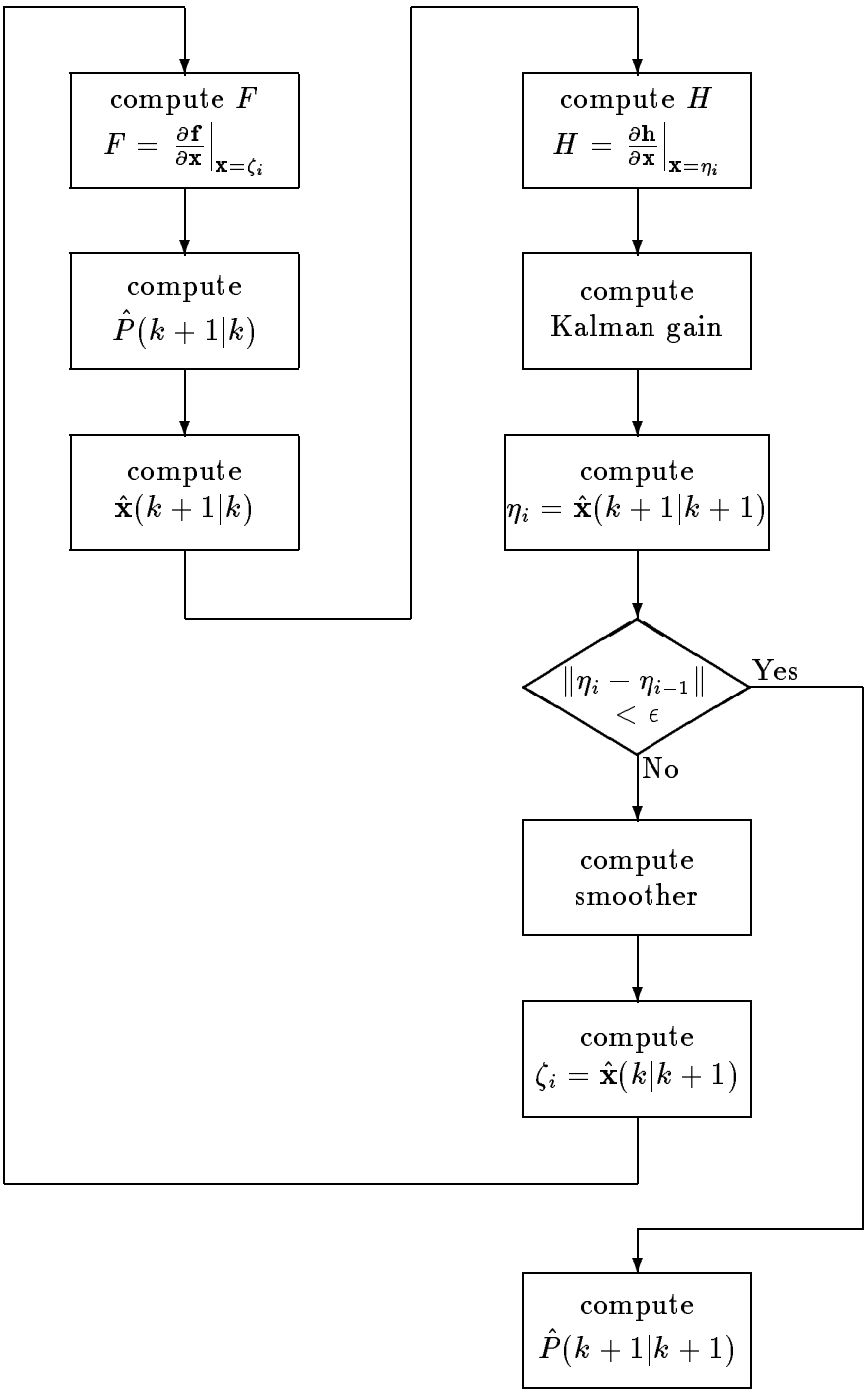


Figure 2.7: Iterated linear filter-smoother flow chart

Chapter 3

RECURSIVE ESTIMATION USING OPTICAL FLOW MEASUREMENTS

As we have seen in the previous chapter, a great deal of effort has been spent on solving the problem of 3-D motion estimation from monocular image sequences in recent years. Among the existing approaches, feature-based and optical-flow-based algorithms are the most popular. While each has its strengths and weaknesses, the most important difference between these two approaches is in the type of measurements. In one approach, discrete features are extracted and tracked over time. In the other approach, 2-D velocities of brightness patterns are computed for every frame in an image sequence. Is it possible to design an algorithm that makes use of *both* types of measurements? What are the potential benefits? We shall attempt to answer these questions in this chapter by incorporating optical flow measurements into a purely featured-based algorithm.

We have selected the recursive algorithm proposed by Broida *et al.* [15] (which we shall call the BCC algorithm hereafter) as our basic feature-based algorithm. After an overview of the BCC algorithm, the new hybrid feature/flow-based algorithm will be described in detail. A few implementation issues that are of practical importance will be discussed, followed by a presentation of the simulation results.

3.1 Overview of the Broida-Chandrashekhar-Chellappa Algorithm

The problem under investigation involves a) a rigid object translating and rotating continuously in time and b) observations that are made at regular time intervals. In control theory terminology, we have a system with a continuous plant and discrete measurements. With this in mind, recovering 3-D motion and structure parameters can be viewed as being equivalent to estimating the unknown parameters of a dynamical system. In [15], Broida *et al.* propose to estimate the 3-D motion and structure of a rigid object recursively using an iterated extended Kalman filter. Before formulating an estimation problem in terms of a Kalman filter, the system dynamics, the state vector, the measurement vector, and the measurement process must be identified. Let us examine them in order.

3.1.1 Two Coordinate Systems

The choice of coordinate system(s) is very important. It has direct influence on the formulation of the solution. Broida *et al.* make use of two coordinate systems (Figure 3.1): the camera-centred coordinate system (CCCS), a stationary reference frame "attached" to the camera, and the object-centred coordinate system (OCCS), a moving frame of reference "attached" to the object.

If we let the position vector of O_o in the CCCS be

$$\mathbf{s}_{\mathrm{R}}(t) = [X_{\mathrm{R}}(t) \quad Y_{\mathrm{R}}(t) \quad Z_{\mathrm{R}}(t)]^T \tag{3.1}$$

and the position vector of the i-th feature point in the OCCS be

$$\mathbf{s}_{\mathrm{oi}} = [X_{\mathrm{oi}} \quad Y_{\mathrm{oi}} \quad Z_{\mathrm{oi}}]^T, \tag{3.2}$$

then the position vector of the i-th feature point in the CCCS is

$$\mathbf{s}_{\mathrm{i}}(t) = \mathbf{s}_{\mathrm{R}}(t) + R(t)\mathbf{s}_{\mathrm{oi}} \tag{3.3}$$

where $R(t)$ is the rotation matrix that aligns the OCCS with the CCCS. Note that due to the rigidity assumption, $\mathbf{s}_{\mathrm{oi}}$ is not dependent on time.

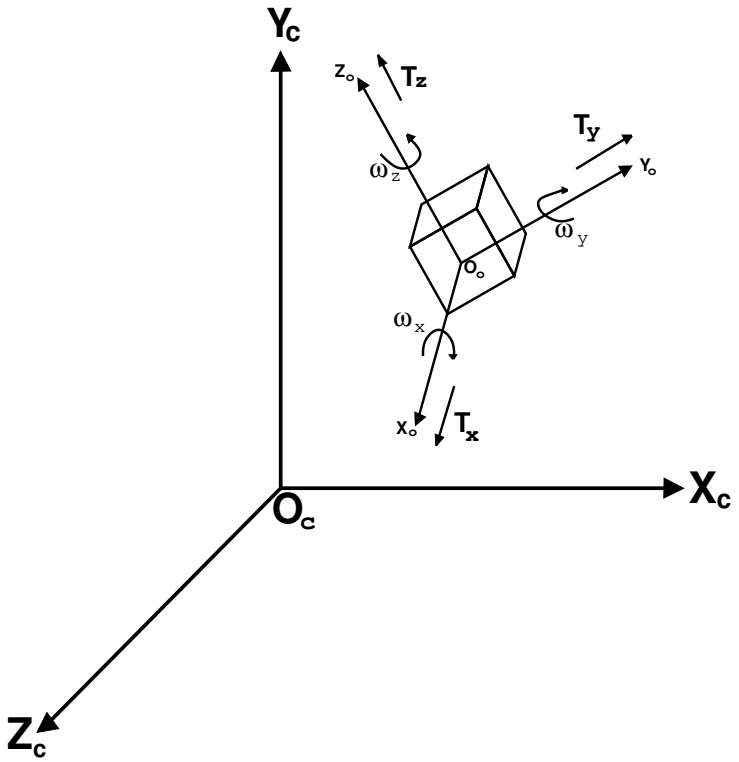


Figure 3.1: Camera- and Object-Centred Coordinate Systems

3.1.2 Motion Models

Broida *et al.* assume that the object is translating at a constant translational velocity $\mathbf{T} = [T_x\ T_y\ T_z]^T$ and rotating about an axis through O_o at a constant rotational velocity $\boldsymbol{\omega} = [\omega_x\ \omega_y\ \omega_z]^T$. The centre of rotation, O_o, is unaffected by the rotational motion of the object; therefore, under the rigidity assumption, the velocity of this point represents the translational velocity of the entire object. Its position at any time t is described by

$$\mathbf{s}_R(t) \quad = \quad \mathbf{s}_R(t_0) + (t - t_0)\mathbf{T}. \tag{3.4}$$

The orientation of the OCCS with respect to the CCCS can be defined as the 3-D rotation required to align the OCCS with the CCCS. Many methods exist for representing rotation [70, 79]: the Euler angles, the axis/angle representation, the roll-pitch-yaw angles, the exponential representation, etc. For instance, the axis/angle representation is used by Young and Chellappa [98], the roll-pitch-yaw approach is used by Wu *et al.* [95], and the exponential representation is used by Silvén and Repo

[76]. However, as pointed out by Sabata and Aggarwal [70], all of these representations have singularity points (*i.e.,* certain angle(s) cannot be represented). The unit quaternion, on the other hand, does not suffer from this problem, although it does require 4 parameters to represent 3 degrees of freedom [32, 39, 57]:

$$\mathbf{q}(t) \;=\; [q_1(t)\;\; q_2(t)\;\; q_3(t)\;\; q_4(t)]^T, \qquad \|\mathbf{q}(t)\| = 1. \tag{3.5}$$

The rotation matrix R that aligns the OCCS with the CCCS can be expressed in terms of $\mathbf{q}(t)$ as

$$R = \begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 - q_3q_4) & 2(q_1q_3 + q_2q_4) \\ 2(q_1q_2 + q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 - q_1q_4) \\ 2(q_1q_3 - q_2q_4) & 2(q_2q_3 + q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix}. \tag{3.6}$$

Note that time dependencies have been omitted for convenience. Two other advantages offered by the unit quaternion representation are smooth propagation of the rotation matrix through time and easy composition of multiple rotations. By assuming constant rotational velocity, the relative orientation can be propagated in time using [31]

$$\mathbf{q}(t) \;=\; \exp[(t - t_0) \times \Omega(\boldsymbol{\omega})]\mathbf{q}(t_0) \tag{3.7}$$

where

$$\Omega(\boldsymbol{\omega}) \;=\; \frac{1}{2}\begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix} \tag{3.8}$$

which is a 4×4 skew symmetric matrix.

3.1.3 Choosing the State Variables

Typically, a state vector contains information sufficient to characterize the condition of a system. The state vector proposed by Broida *et al.* consists of the image plane coordinates of the origin of the OCCS, the translational velocity, the unit quaternion

representing the orientation of the object, the rotational velocity, and the coordinates of the feature points in the OCCS:

$$\mathbf{x}(t) = \begin{bmatrix} X_R(t)/Z_R(t) \\ Y_R(t)/Z_R(t) \\ \dot{X}_R/Z_R(t) \\ \dot{Y}_R/Z_R(t) \\ \dot{Z}_R/Z_R(t) \\ q_1 \\ q_2 \\ q_3 \\ q_4 \\ \omega_x \\ \omega_y \\ \omega_z \\ X_{o1}/Z_R(t) \\ Y_{o1}/Z_R(t) \\ Z_{o1}/Z_R(t) \\ \vdots \\ X_{oM}/Z_R(t) \\ Y_{oM}/Z_R(t) \\ Z_{oM}/Z_R(t) \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \\ s_{10} \\ s_{11} \\ s_{12} \\ s_{13} \\ s_{14} \\ s_{15} \\ \vdots \\ s_{10+3M} \\ s_{11+3M} \\ s_{12+3M} \end{bmatrix}. \tag{3.9}$$

In this thesis, states s_1–s_5 will be referred to as the translational states, states s_6–s_{12} will be called the rotational states, and states s_{13}–s_{12+3M} will be termed the structural states. Due to the fact that only one camera is used, the translational and structural parameters can only be determined up to a scale factor. Broida *et al.* normalize the translational and structural states by $Z_R(t)$ to eliminate this unknown scale factor.

Having chosen the state variables, obtaining $\mathbf{f}(\cdot)$ is simply a matter of differentiating (3.9) with respect to time, *i.e.*,

$$\mathbf{f}(\mathbf{x}(t), t) = \frac{d\mathbf{x}(t)}{dt}. \tag{3.10}$$

Given $\mathbf{f}(\cdot)$, it is straightforward to compute the F matrix which is required by the iterated extended Kalman filter and the iterated linear filter-smoother (see (2.28)).

3.1.4 Position Measurements of Feature Points

The structure of the measurement vector is very simple. In a feature-based algorithm, the measurements are normally the image plane coordinates of the feature points. Using the perspective projection, the measurement vector is given by

$$\mathbf{y}_k = \begin{bmatrix} x_1 \\ y_1 \\ \vdots \\ x_M \\ y_M \end{bmatrix} = \begin{bmatrix} X_1/Z_1 \\ Y_1/Z_1 \\ \vdots \\ X_M/Z_M \\ Y_M/Z_M \end{bmatrix}. \tag{3.11}$$

The measurement function is highly non-linear in the state variables. To see this, let us examine the image plane coordinates of the i-th feature point:

$$x_i = \frac{X_i}{Z_i} = \frac{X_R + \mathbf{R}_1 \mathbf{s}_{oi}}{Z_R + \mathbf{R}_3 \mathbf{s}_{oi}} \tag{3.12}$$

$$y_i = \frac{Y_i}{Z_i} = \frac{Y_R + \mathbf{R}_2 \mathbf{s}_{oi}}{Z_R + \mathbf{R}_3 \mathbf{s}_{oi}} \tag{3.13}$$

where $\mathbf{R}_1$, $\mathbf{R}_2$, and $\mathbf{R}_3$ are row vectors defined as follows:

$$\mathbf{R}_1 = [(q_1^2 - q_2^2 - q_3^2 + q_4^2) \quad 2(q_1 q_2 - q_3 q_4) \quad 2(q_1 q_3 + q_2 q_4)] \tag{3.14}$$

$$\mathbf{R}_2 = [2(q_1 q_2 + q_3 q_4) \quad (-q_1^2 + q_2^2 - q_3^2 + q_4^2) \quad 2(q_2 q_3 - q_1 q_4)] \tag{3.15}$$

$$\mathbf{R}_3 = [2(q_1 q_3 - q_2 q_4) \quad 2(q_2 q_3 + q_1 q_4) \quad (-q_1^2 - q_2^2 + q_3^2 + q_4^2)]. \tag{3.16}$$

Dividing the right hand side of (3.12) and (3.13) by Z_R, we get

$$x_i = \frac{X_R/Z_R + \mathbf{R}_1 \mathbf{s}_{oi}/Z_R}{1 + \mathbf{R}_3 \mathbf{s}_{oi}/Z_R} \tag{3.17}$$

$$y_i = \frac{Y_R/Z_R + \mathbf{R}_2 \mathbf{s}_{oi}/Z_R}{1 + \mathbf{R}_3 \mathbf{s}_{oi}/Z_R}. \tag{3.18}$$

Finally, we can express (3.17) and (3.18) in terms of the state variables,

$$x_i = \frac{s_1 + a_i}{1 + c_i} \tag{3.19}$$

$$y_i = \frac{s_2 + b_i}{1 + c_i} \tag{3.20}$$

where

$$\begin{aligned} a_i &= (q_1^2 - q_2^2 - q_3^2 + q_4^2)\frac{X_{oi}}{Z_R} + 2(q_6q_7 - q_8q_9)\frac{Y_{oi}}{Z_R} + 2(q_6q_8 + q_7q_9)\frac{Z_{oi}}{Z_R} \\ &= (s_6^2 - s_7^2 - s_8^2 + s_9^2)s_{10+3i} + 2(s_6s_7 - s_8s_9)s_{11+3i} + \\ &\quad 2(s_6s_8 + s_7s_9)s_{12+3i} \end{aligned} \tag{3.21}$$

$$\begin{aligned} b_i &= 2(q_1q_2 + q_3q_4)\frac{X_{oi}}{Z_R} + (-q_1^2 + q_2^2 - q_3^2 + q_4^2)\frac{Y_{oi}}{Z_R} + 2(q_2q_3 - q_1q_4)\frac{Z_{oi}}{Z_R} \\ &= 2(s_6s_7 + s_8s_9)s_{10+3i} + (-s_6^2 + s_7^2 - s_8^2 + s_9^2)s_{11+3i} + \\ &\quad 2(s_7s_8 - s_6s_9)s_{12+3i} \end{aligned} \tag{3.22}$$

$$\begin{aligned} c_i &= 2(q_1q_3 - q_2q_4)\frac{X_{oi}}{Z_R} + 2(q_2q_3 + q_1q_4)\frac{Y_{oi}}{Z_R} + (-q_1^2 - q_2^2 + q_3^2 + q_4^2)\frac{Z_{oi}}{Z_R} \\ &= 2(s_6s_8 - s_7s_9)s_{10+3i} + 2(s_7s_8 + s_6s_9)s_{11+3i} + \\ &\quad (-s_6^2 - s_7^2 + s_8^2 + s_9^2)s_{12+3i} \end{aligned} \tag{3.23}$$

3.2 Optical Flow Measurement Formulation

Suppose that, in addition to the image plane coordinates of the feature points, we also have available to us the optical flow vector at each one of these locations. In other words, we have added optical flow measurements to our observations. By measuring the instantaneous 2-D velocities of the feature points, thus supplying the Kalman filter with more information, we hope to improve the accuracy of the state estimates.

3.2.1 Modifying the BCC Algorithm

Assuming perspective projection and differentiating the image coordinates (x_i, y_i) with respect to time, we get the optical flow equations:

$$u_i = \frac{dx_i}{dt} = \frac{d}{dt}\left(\frac{X_i}{Z_i}\right) = \frac{\dot{X}_i}{Z_i} - \frac{X_i\dot{Z}_i}{Z_i^2} \tag{3.24}$$

$$v_i = \frac{dy_i}{dt} = \frac{d}{dt}\left(\frac{Y_i}{Z_i}\right) = \frac{\dot{Y}_i}{Z_i} - \frac{Y_i\dot{Z}_i}{Z_i^2} \tag{3.25}$$

where, without loss of generality, the focal length is assumed to be unity.

In order to incorporate optical flow data into the recursive filtering framework proposed by Broida *et al.* , (3.24) and (3.25) must be expressed in terms of the existing state variables. First we re-arrange (3.24) and (3.25) to get

$$\begin{aligned} u_i &= \frac{\dot{X}_{\mathrm{i}}}{Z_{\mathrm{i}}} - \frac{X_{\mathrm{i}}}{Z_{\mathrm{i}}}\frac{\dot{Z}_{\mathrm{i}}}{Z_{\mathrm{i}}} \\ &= \frac{\dot{X}_{\mathrm{i}}}{Z_{\mathrm{R}}}\frac{Z_{\mathrm{R}}}{Z_{\mathrm{i}}} - \frac{X_{\mathrm{i}}}{Z_{\mathrm{R}}}\frac{Z_{\mathrm{R}}}{Z_{\mathrm{i}}}\frac{\dot{Z}_{\mathrm{i}}}{Z_{\mathrm{R}}}\frac{Z_{\mathrm{R}}}{Z_{\mathrm{i}}} \end{aligned} \tag{3.26}$$

$$\begin{aligned} v_i &= \frac{\dot{Y}_{\mathrm{i}}}{Z_{\mathrm{i}}} - \frac{Y_{\mathrm{i}}}{Z_{\mathrm{i}}}\frac{\dot{Z}_{\mathrm{i}}}{Z_{\mathrm{i}}} \\ &= \frac{\dot{Y}_{\mathrm{i}}}{Z_{\mathrm{R}}}\frac{Z_{\mathrm{R}}}{Z_{\mathrm{i}}} - \frac{Y_{\mathrm{i}}}{Z_{\mathrm{R}}}\frac{Z_{\mathrm{R}}}{Z_{\mathrm{i}}}\frac{\dot{Z}_{\mathrm{i}}}{Z_{\mathrm{R}}}\frac{Z_{\mathrm{R}}}{Z_{\mathrm{i}}}. \end{aligned} \tag{3.27}$$

Clearly, we need the velocity of the i-th feature point with respect to the camera, *i.e.*, $\dot{\mathbf{s}}_i(t) = [\dot{X}_i\ \dot{Y}_i\ \dot{Z}_i]^T$. Differentiating (3.3) with respect to time, we get

$$\frac{d}{dt}\mathbf{s}_{\mathrm{i}}(t) = \frac{d}{dt}\mathbf{s}_{\mathrm{R}}(t) + \frac{dR(t)}{dt}\mathbf{s}_{\mathrm{oi}} \tag{3.28}$$

where the time derivative of the rotation matrix is [2, 79]

$$\frac{d}{dt}R(t) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} R(t). \tag{3.29}$$

Then, we establish the following relationship

$$\frac{Z_{\mathrm{R}}}{Z_{\mathrm{i}}} = \frac{Z_{\mathrm{R}}}{Z_{\mathrm{R}} + \mathbf{R}_3\,\mathbf{s}_{\mathrm{io}}} = \frac{1}{1 + \mathbf{R}_3\,\frac{\mathbf{s}_{\mathrm{io}}}{Z_{\mathrm{R}}}}. \tag{3.30}$$

Now, we are finally ready to express (3.24) and (3.25) in terms of the state variables:

$$u_i = \frac{s_3 + c_i\,s_{11} - b_i\,s_{12}}{1 + c_i} - \frac{(s_5 + b_i\,s_{10} - a_i\,s_{11})(s_1 + a_i)}{(1 + c_i)^2} \tag{3.31}$$

$$v_i = \frac{s_4 - c_i\,s_{10} + a_i\,s_{12}}{1 + c_i} - \frac{(s_5 + b_i\,s_{10} - a_i\,s_{11})(s_2 + b_i)}{(1 + c_i)^2} \tag{3.32}$$

where a_i, b_i, and c_i are as defined in (3.21), (3.22), and (3.23), respectively.

The measurement vector $\mathbf{y}_k$ described in the previous section will have to be expanded as there are now two extra observations at each feature point location:

$$\mathbf{y}_k = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_M \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ \vdots \\ x_M \\ y_M \\ u_1 \\ v_1 \\ \vdots \\ u_M \\ v_M \end{bmatrix}. \tag{3.33}$$

The individual elements of the Jacobian matrix H_k are obtained by differentiating (3.19), (3.20), (3.31), and (3.32) with respect to each state variable (see Appendix A):

$$H_k = \frac{d\mathbf{h}_k}{d\mathbf{x}} = \begin{bmatrix} \frac{\partial x_1}{\partial s_1} & \cdots & \frac{\partial x_1}{\partial s_{12+3M}} \\ \frac{\partial y_1}{\partial s_1} & \cdots & \frac{\partial y_1}{\partial s_{12+3M}} \\ \vdots & \cdots & \vdots \\ \frac{\partial u_M}{\partial s_1} & \cdots & \frac{\partial u_M}{\partial s_{12+3M}} \\ \frac{\partial v_M}{\partial s_1} & \cdots & \frac{\partial v_M}{\partial s_{12+3M}} \end{bmatrix}. \tag{3.34}$$

3.2.2 Optical Flow from Feature Displacement

A logical question at this point is, "How does one obtain the optical flow measurements?". Although many algorithms have been proposed over the years, most of them are computationally expensive. For certain applications, such as real-time visual tracking, it is simply not practical to compute optical flow using one of these time-consuming techniques.

Let us think of optical flow as a time derivative. It is well known that a derivative can be approximated by a finite difference. By measuring the displacement of a feature point from time t_k to time t_{k+1} and dividing it by the time interval $\Delta t =$

$t_{k+1} - t_k$, we can compute the average image plane velocity of that feature point, or the instantaneous image plane velocity if Δt is small enough. (See [65] for a related discussion.) In other words, this forward difference method of approximating a derivative will allow us to obtain optical flow measurements:

$$u_k(\epsilon) = \frac{x_{k+1}(\epsilon) - x_k(\epsilon)}{\Delta t} \tag{3.35}$$

$$v_k(\epsilon) = \frac{y_{k+1}(\epsilon) - y_k(\epsilon)}{\Delta t}. \tag{3.36}$$

The additional cost of computing optical flow using this approximation technique is negligible since the optical flow measurements are derived directly from measurements that are already available, namely, the position measurements. We will call this the feature displacement method (FDM).

Other approximation schemes, such as the backward difference method and the central difference method, are essentially the same as the forward difference method. However, the use of the central difference method would require, for every frame, backward interpolation by half a sampling period. This is undesirable because the extra interpolation step introduces unnecessary overhead and affects the computational efficiency of the filter.

3.3 Implementation Detail

As we have seen in Section 2.3, the use of the Kalman filter requires knowledge of the process noise covariance matrix and the measurement noise covariance matrix. We also need to supply a reasonably accurate estimate of the initial state vector as well as an initial estimate for the error covariance matrix. These issues will be addressed in the following sections.

3.3.1 Description of the Synthetic Testbed

A flexible synthetic testbed has been developed to evaluate our new hybrid feature/flow-based 3-D trajectory estimation algorithm. The entire system is implemented in software using MATLAB [58], a numeric computation package. The vector differentiation

in (3.34) is carried out by hand and verified using Maple [20], a software package for symbolic mathematical calculations. Easy modification of simulation parameters is made possible by the use of a global configuration file. This is important because to ensure the robustness of an algorithm, it must be tested under a wide range of conditions.

Our "synthetic world" consists of a stationary "pin-hole" camera and a transparent cube of size $3 \times 3 \times 3$. Unless otherwise stated, all distances are measured in terms of focal lengths (abbreviated as FL). Four non-coplanar vertices are chosen, mostly arbitrarily, as the feature points ($M = 4$, see Figure 3.2). Certainly, more feature points may be used but every additional feature point increases the size of the state vector by three elements. The trajectory of the object is generated using the kinematic equations associated with the motion model. while the state vector is propagated forward in time by means of numerical integration (using the second- and third-order Runge-Kutta forumulae [58]), due to the non-linearity of the plant.

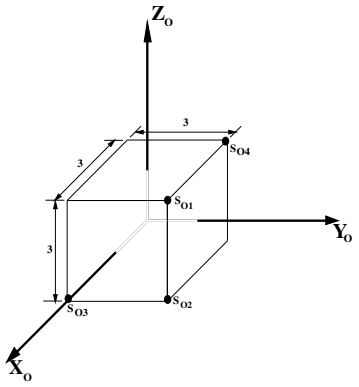


Figure 3.2: Feature points in object-centred coordinate system

3.3.2 Evaluating Three Suboptimal Filters

It is commonly believed that if the measurement function is highly non-linear, then a more "sophisticated" non-linear filter, such as the IEKF, would perform better than the EKF. However, as pointed out by Wishner *et al.* , the relative performance of non-linear filters is highly dependent on the particular problem at hand [93]. Though important, comparison of non-linear filters has not received very much attention in

the motion estimation literature. Koller *et al.* report that the IEKF is superior to the EKF based on their application of estimating the 2-D motion of vehicles [51]. No experimental data is given to support their claim, although there is reference to an unpublished technical report. Maybank compares the optimal filter, a second-order approximation to the optimal filter, and the EKF [60]. The basic problem involves estimating the "depth" of a point moving in 2-D space. The author concluded that the second-order approximation is closer to the optimal filter than the EKF.

The relative performance of *three* non-linear filters (namely, the EKF, IEKF, and ILFS) will be studied in this thesis for the purpose of 3-D motion estimation from monocular image sequences. The algorithms outlined in [44] suggest that the IEKF/ILFS iteration should be terminated when the difference between two consecutive estimates falls below a pre-defined threshold. Our current implementation uses a slightly different termination criterion. The iteration is stopped when the difference between two consecutive estimates no longer decreases in magnitude, which means that there is no need to select a suitable threshold. Furthermore, a limit is set on the maximum number of iterations (arbitrarily chosen to be 5 for both the IEKF and ILFS).

3.3.3 Simulating Noisy Measurements

In this research, we only consider two sources of errors: spatial quantization and feature extraction. In order to simplify the analysis, feature correspondences are assumed to be perfect. Since actual measurements are not available, they must be *simulated*. One may accomplish this by perturbing the noise-free or error-free values with random quantities. If we denote the error-free position measurement of the i-th feature point by $\mathbf{p}_i$, then the corresponding noisy measurement $\mathbf{p}_i(\epsilon)$ will be calculated as

$$\mathbf{p}_i(\epsilon) \;=\; \mathbf{p}_i + \mathbf{r}_p, \qquad \mathbf{r}_p \sim \mathcal{N}(\mathbf{0}, \sigma_p^2 I) \tag{3.37}$$

where I is a 2×2 identity matrix. Note that the errors in the individual components are assumed to be uncorrelated. Similarly, if the noise-free optical flow measurement for the i-th feature point is denoted by $\mathbf{u}_i$, then the noisy optical flow measurement

$\mathbf{u}_i(\epsilon)$ will be computed as

$$\mathbf{u}_i(\epsilon) \quad = \quad \mathbf{u}_i + \mathbf{r}_u, \qquad \mathbf{r}_u \sim \mathcal{N}(\mathbf{0}, \sigma_u^2 I) \tag{3.38}$$

where I is a 2×2 identity matrix. Again, the errors in the individual components are assumed to be uncorrelated.

All of the noisy position and optical flow measurements used in this thesis are computed using (3.37) and (3.38), respectively. The image plane is chosen to be a unit square (measured in terms of focal lengths) whose the origin is located in the centre. Because the focal length is assumed to be unity, this configuration is equivalent to a camera system with a 53° field of view. If we further assume that the image resolution is 512×512, then the inter-pixel distance, in terms of focal lengths, is $1/512 \approx 0.002$. The standard deviation of the position measurement noise is chosen to be one pixel or $\sigma_p = 0.002$. In other words, 99.7% of the measurements lie within 3 pixels of the true values. As a comparison, if one considered only the noise introduced by spatial quantization, then one would use $\sigma_p = 0.0006$. The value of σ_u is chosen to produce average relative errors for optical flow measurements that are consistent with what other researchers have used [9, 45, 46, 83].

3.3.4 Measurement Noise Covariance Matrix

As mentioned in the previous section, the only two sources of error under consideration are spatial quantization and feature extraction. Assuming that reasonably accurate knowledge about the camera is available, one should be able to estimate the variance of the spatial quantization noise. Furthermore, given a particular feature detector, the uncertainty of its output can be obtained through a statistical error analysis. Therefore, it appears that it is feasible to obtain an accurate estimate of the measurement noise covariance matrix R_k. In general, R_k is time-varying but, for

simplicity, it is implemented as a constant diagonal matrix in the simulations:

$$R_k = \begin{bmatrix} \sigma_p^2 & & & & & \\ & \ddots & & & \mathbf{0} & \\ & & \sigma_p^2 & & & \\ & & & \sigma_u^2 & & \\ & \mathbf{0} & & & \ddots & \\ & & & & & \sigma_u^2 \end{bmatrix}. \tag{3.39}$$

3.3.5 Measuring the Noise Level

Broida *et al.* define the position measurement noise level as "the ratio of the standard deviation of the quantization error to that of the signal (expressed as a percentage)" [15]. The signal standard deviation is, in turn, defined as "the root-mean-square distance of the feature points from their centroid, computed during the frame when the object is farthest from the camera" [15]. However, a simpler and more intuitive measure would be preferable. The average absolute error in position measurements is defined as

$$\epsilon_{\text{pm}} = \sum_R \sum_N \sum_M (|p_x(\epsilon) - p_x| + |p_y(\epsilon) - p_y|) \tag{3.40}$$

where R represents the total number of trials in a set of Monte Carlo simulations, N denotes the total number of frames in an image sequence, and M is the total number of feature points. This quantity is expressed in terms of pixels. While the average absolute error is suitable for position measurements, the average relative error is more appropriate for optical flow measurements. It is defined as

$$\epsilon_{\text{ofm}} = \sum_R \sum_N \sum_M \left(\frac{\| \mathbf{v}(\epsilon) - \mathbf{v} \|}{\| \mathbf{v} \|} \right) \times 100\%. \tag{3.41}$$

The main disadvantage of the relative error, which should be evident from (3.41), is that it is undefined if the norm of the "true" vector is zero. It is worth mentioning that the average relative error, when used for position measurements, has the undesirable effect of being dependent on the choice of origin in the image plane.

3.3.6 Process Noise Covariance Matrix

The process noise covariance matrix plays an important role in the overall performance of a Kalman filter. From (2.19), one can see that if the Kalman gain is too small, then contributions of the current measurement will be ignored. On the other hand, if the Kalman gain is too large, then small residual errors will be magnified, thus affecting the accuracy of the updated state estimates. However, one needs to realize that the Kalman gain is directly proportional to the process noise covariance matrix and inversely proportional to the measurement noise covariance matrix (see (2.17) and (2.20)). This makes sense because if we do not trust our motion model, then Q_k should be large. A large Q_k will increase the Kalman gain, which will force the filter to weigh the measurements more. Conversely, if the measurements are extremely noisy, then R_k should be large. A large R_k will decrease the Kalman gain, which will allow the filter to weigh the measurements less. In the extreme case, a zero Kalman gain will lead to a filter that ignores even perfect measurements.

Clearly, both the measurement noise covariance matrix and the process noise covariance matrix contribute to the overall behaviour of the Kalman gain; therefore, it is important to be able to estimate both $Q(k+1)$ and $R(k+1)$ accurately. Unfortunately, the "proper amount" of process noise is application-dependent and no systematic procedure exists for determining it. As in the case of the measurement noise covariance matrix, the process noise covariance matrix is implemented as a constant diagonal matrix for simplicity. Unless otherwise stated, all of the diagonal entries have the same value of 5×10^{-6} (selected by varying the value by factors of 10).

3.3.7 Simulating Uncertainty in Initial State Estimate

Like many other recursive procedures, a Kalman filter needs to be initialized. Specifically, initial values for the estimated state vector and error covariance matrix must be given. The errors in the initial state estimates are simulated in a way similar to the noisy measurements, *i.e.,* the true initial states are perturbed by random quantities. However, simply adding a random error vector to the true vector would not be appropriate here. We need more control over the amount of error added to each

individual element in the vector. This is because in practice, one would use, for example, a ranging device and/or a batch algorithm [16] to obtain fairly reliable initial values. The following formula allows uncertainty in the initial state estimates to be simulated:

$$\hat{\mathbf{x}}_0(i) \;=\; [1 \pm (1+r)e]\mathbf{x}_0(i), \qquad i = 1, \ldots, 8, 13, \ldots, 12+3M \tag{3.42}$$

where r is a random number uniformly distributed in the interval $(0, 1)$ and e is used to control the magnitude of the errors. No noise is added to the initial estimates for the quaternion states (s_6–s_9) because the OCCS is always assumed to be aligned with the CCCS initially. By setting $e = 0.2$, for example, one can introduce errors in the range 20%–40% and unless otherwise stated, $e = 0.2$ will be used for all the experiments.

Now, the remaining task is to devise a scheme for initializing the error covariance matrix. The individual entries in this matrix reflect our confidence in the initial values. In practice, this would not be a difficult task since the method for obtaining the initial values is always known. In a simulation, however, it would be interesting to see how a filter performs when the accuracy of the initial values is under-estimated. Therefore, given $\hat{\mathbf{x}}_0$ and $\mathbf{x}_0$, the initial error covariance matrix is computed as $P_0 = d^2 I$ where I is a $(12+3M) \times (12+3M)$ identity matrix and d is the largest component in the vector $|\hat{\mathbf{x}}_0 - \mathbf{x}_0|$.

3.4 Simulation Results

3.4.1 Design of Experiments

We have designed a more comprehensive collection of experiments than those outlined in [15] by Broida *et al.* . First of all, all the results presented in this thesis are obtained from *Monte Carlo* simulations. Broida *et al.* presented their results based on single-trial experiments, instead of multiple-trial experiments. One must realize, however, that they introduce "noise" to the image plane coordinates of the feature points by quantizing them to the desired resolution. In other words, for a given set of initial conditions, motion parameters, and camera specifications, the "noisy" image plane

coordinates can be computed deterministically and it would be meaningless to conduct Monte Carlo simulations. Unless otherwise stated, each experiment discussed in this chapter is made up of $R = 30$ trials and each trial consists of $N = 100$ image frames, excluding the initial frame.

Secondly, a total of five different experiments have been conducted. The aim of Experiment 1 is to study, specifically for the purpose of 3-D trajectory estimation, the relative performance of the three suboptimal non-linear filters: the EKF, IEKF, and ILFS. Then, using simulated noisy optical flow measurements, the new hybrid feature/flow-based algorithm is compared to the BCC algorithm (which serves as a "baseline") in three different situations. In Experiment 2, the object travels at constant translational and rotational velocities, *i.e.,* the motion model is accurate. In Experiment 3, slight deviation from the motion model is introduced in the form of translational acceleration. In Experiment 4, the object is allowed to change its motion parameters abruptly after a fixed number of frames. Finally, in Experiment 5, we examine how the proposed algorithm performs when optical flow measurements are obtained by the feature displacement method.

The results of the above mentioned experiments are presented in this section. Since the main objective is 3-D trajectory estimation, certain states are of special importance, namely, the position states (s_1 and s_2), translatoinal velocity states (s_3–s_5), and rotational velocity states (s_{10}–s_{12}). The bias and variance of only these states will be included. It is important to point out that all the results corresponding to the position and translational velocity states have been scaled by $Z_R(t)$, the true depth of the object's rotation centre. Consider the estimated value of one particular state variable $\hat{x}(t)$ at time t. The bias is calculated as

$$b = \frac{1}{R}\sum_{i=1}^{R}\hat{x}_i(t) - x(t) \tag{3.43}$$

and the variance is calculated as

$$v = \frac{1}{R}\sum_{i=1}^{R}\left(\hat{x}_i(t) - x(t)\right)^2 \tag{3.44}$$

where N is the total number of Monte Carlo trials, $x(t)$ is the true value at time t, and the subscript i denotes the trial number.

3.4.2 Experiment 1: Filter Evaluation

The goal of this experiment is to study the relative performance of three suboptimal non-linear filters: the EKF, IEKF, and ILFS. The original BCC algorithm is used in this experiment, *i.e.,* only position measurements are used. The object, which is initially far away from the camera, approaches the camera at constant translational and rotational velocities. The motion of the object, as seen in the image plane, is shown in Figure 3.3 where the object is represented as a wireframe model. Table 3.1 summarizes the motion parameters used. The average absolute error in position measurements is 1.6 pixels.

initial position $\mathbf{s}_R(t_0)$ (FL)	$[-6 \quad 10 \quad 28]^T$
translational velocity $\mathbf{T}$ (FL/frame)	$[0.05 \quad -0.1 \quad -0.2]^T$
rotational velocity $\boldsymbol{\omega}$ (radians/frame)	$[0.03 \quad 0.04 \quad 0.05]^T$

Table 3.1: Motion parameters for Experiment 1

The simulation results are shown in Figures 3.4 (bias) and 3.5 (variance). The EKF performs slightly better than the other two filters in the position states (s_1 and s_2). In the translational velocity states (s_3–s_5), all three filters exhibit similar steady-state performance but the IEKF is clearly the best during the first 30 frames. Finally, in the rotational velocity states (s_{10}–s_{12}), the behaviours of the three filters are very similar. Based on the simulation results, there is no reason to believe that the IEKF or ILFS is superior to the EKF. In fact, due to their iterative nature, the IEKF and ILFS are inherent less efficient than the EKF, in terms of execution speed.

However, before a conclusive statement can be made about the relative performance of the three filters, a more thorough investigation involving many different sets of motion parameters will be necessary. Two important properties that should be studied are numerical stability and rate of convergence. The trade-off between computational efficiency and accuracy cannot be overlooked either.

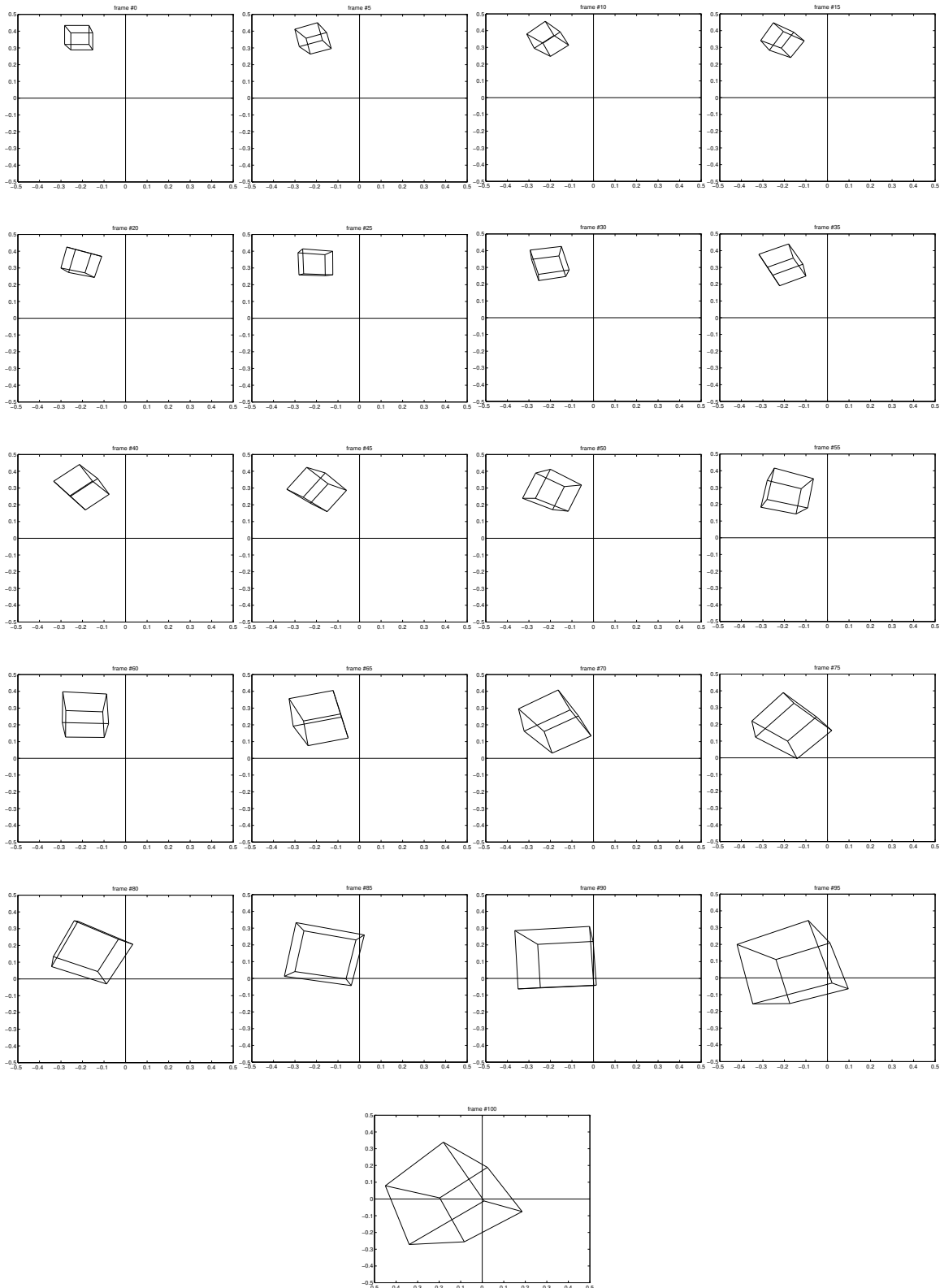
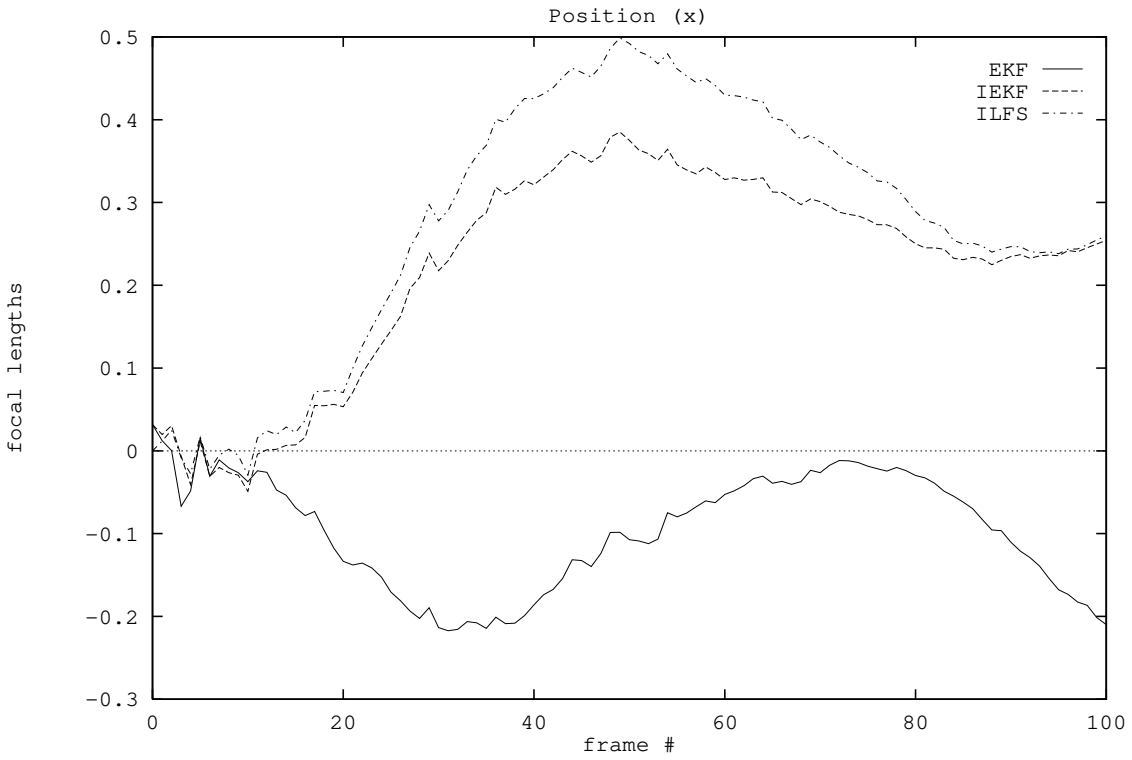
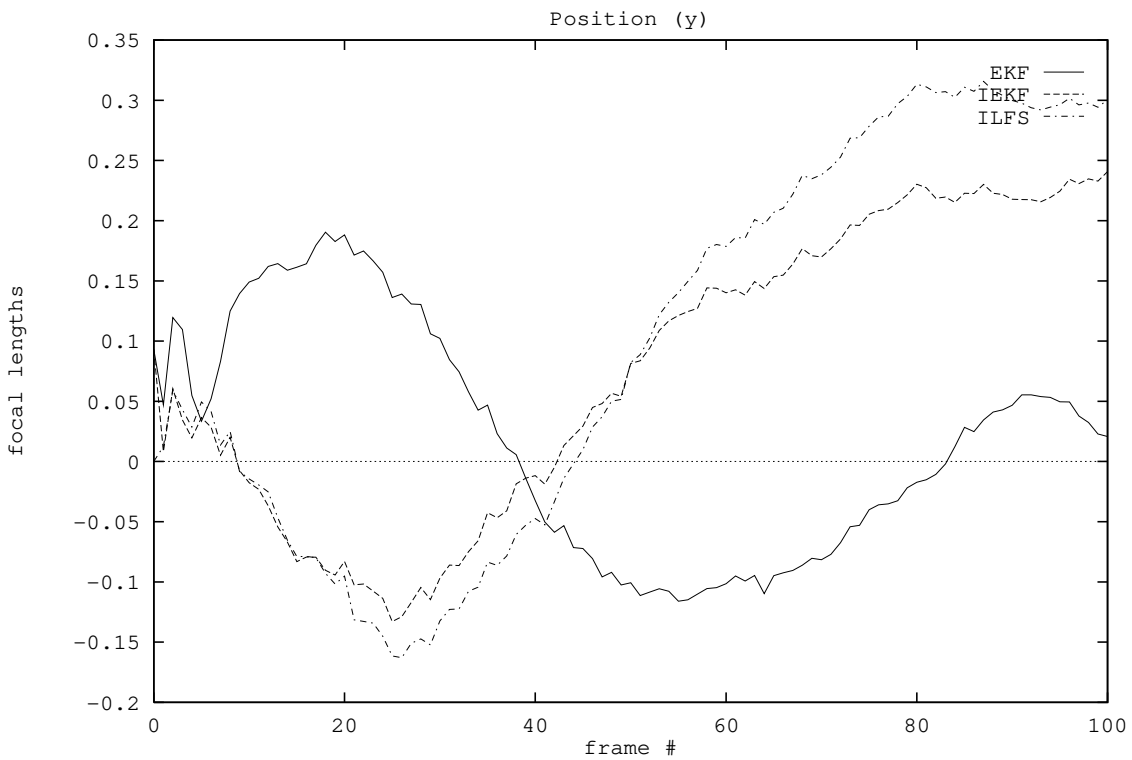


Figure 3.3: Experiment 1: Object motion in image plane

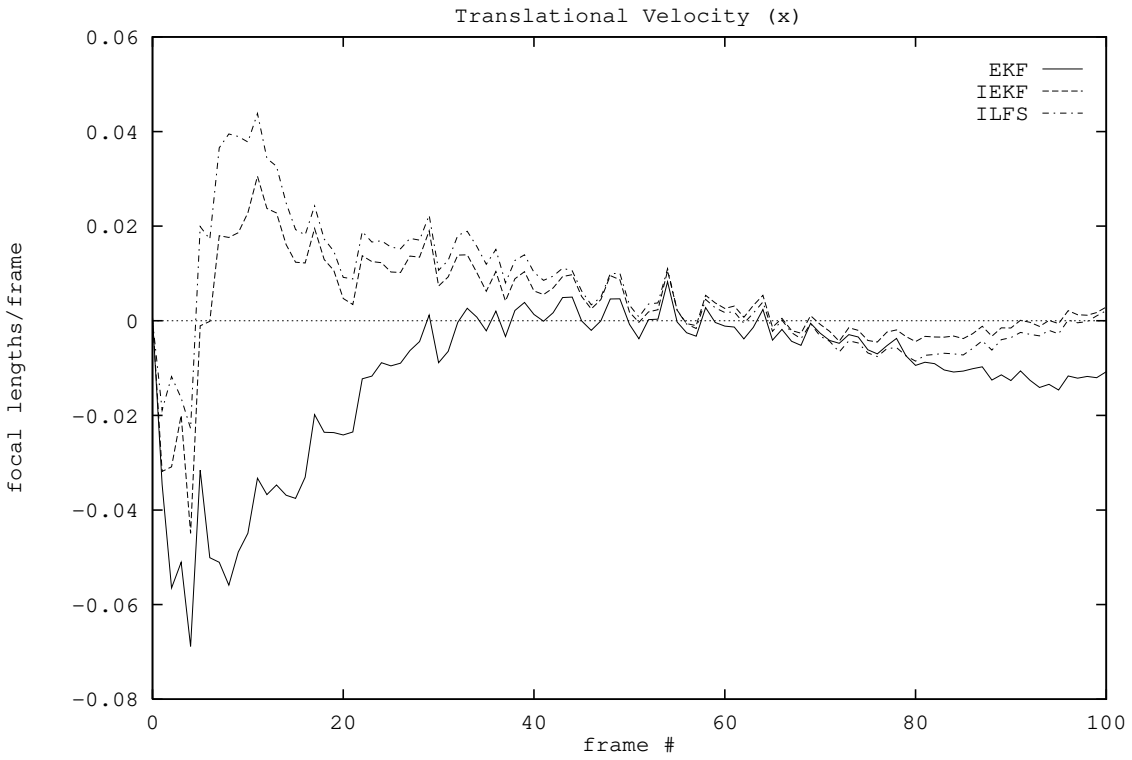


(a)

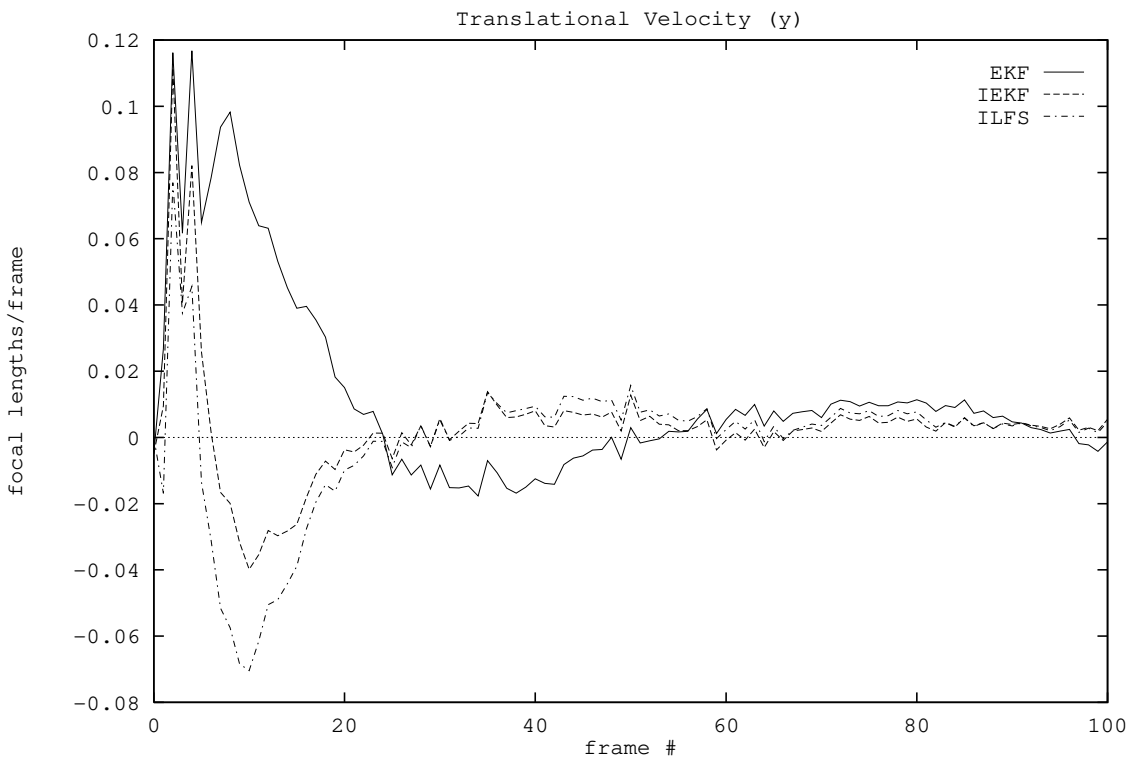


(b)

Figure 3.4: Experiment 1: Results of Monte Carlo simulations (bias)

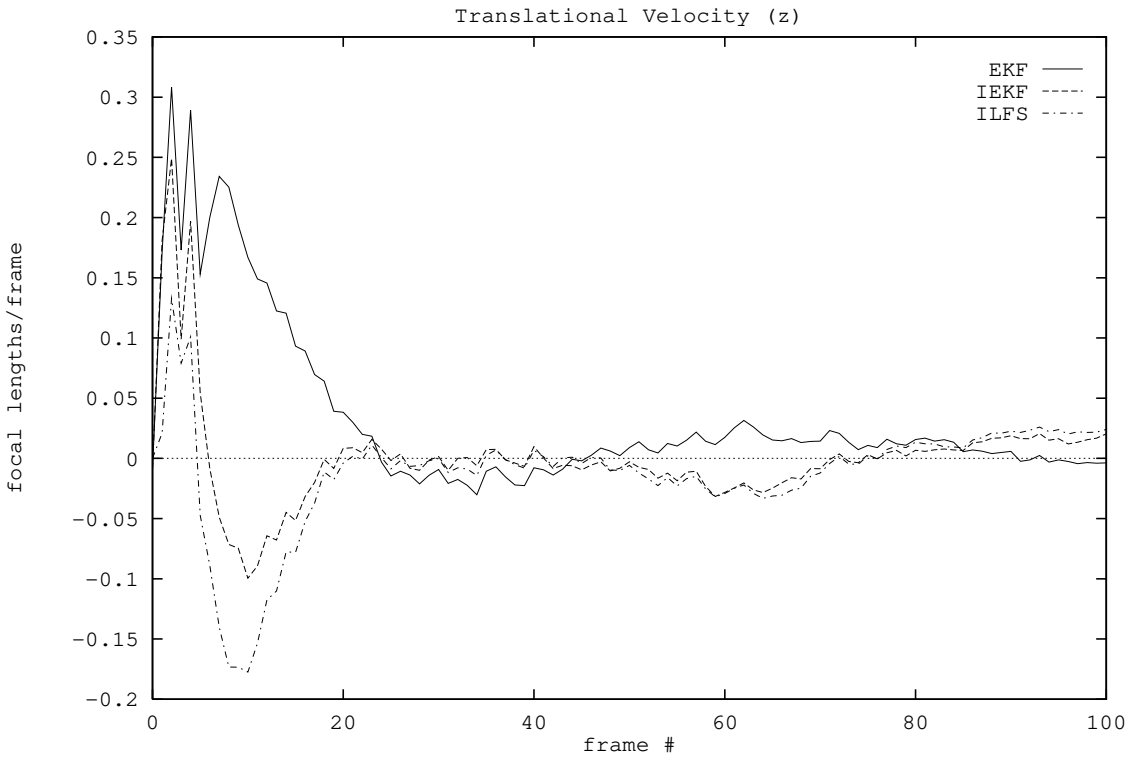


(c)

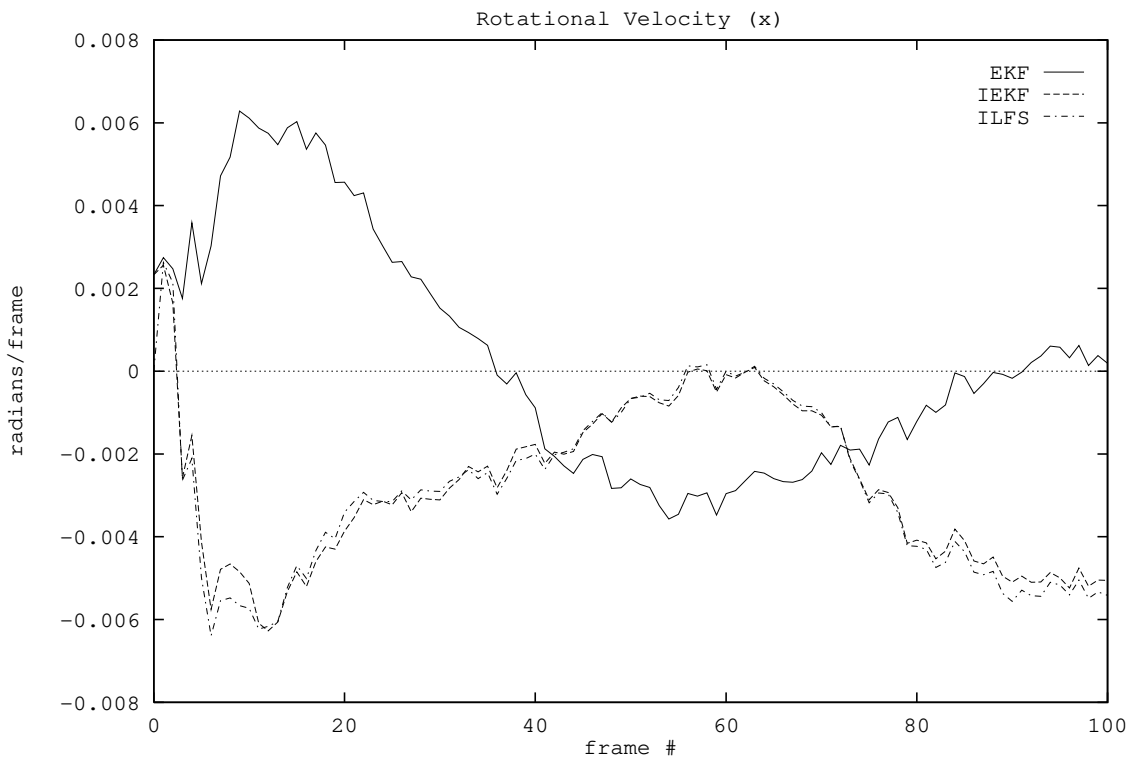


(d)

Figure 3.4: Experiment 1: Results of Monte Carlo simulations (bias)

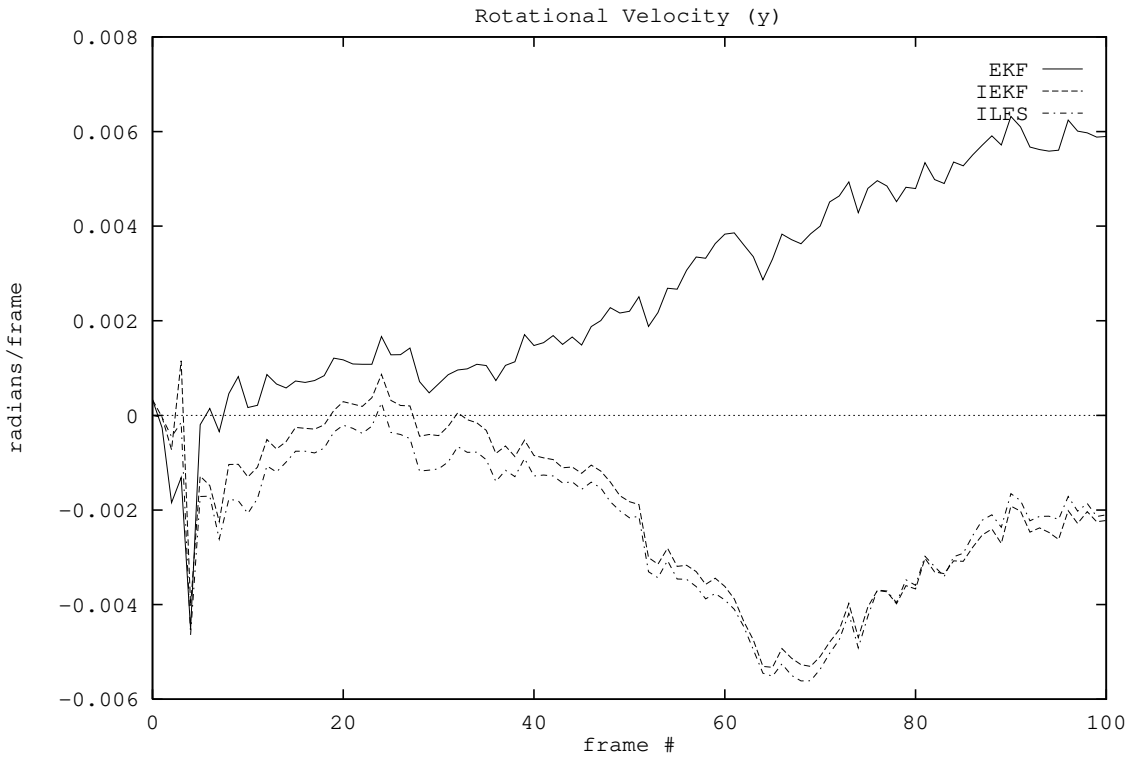


(e)

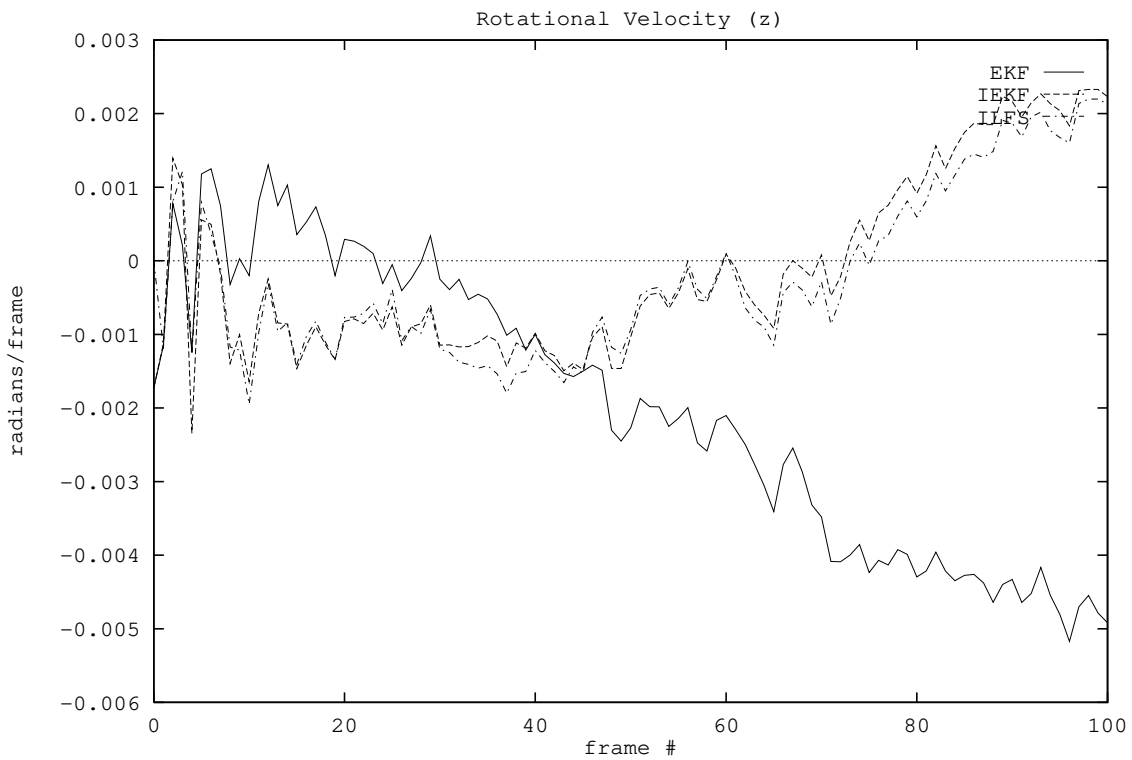


(f)

Figure 3.4: Experiment 1: Results of Monte Carlo simulations (bias)

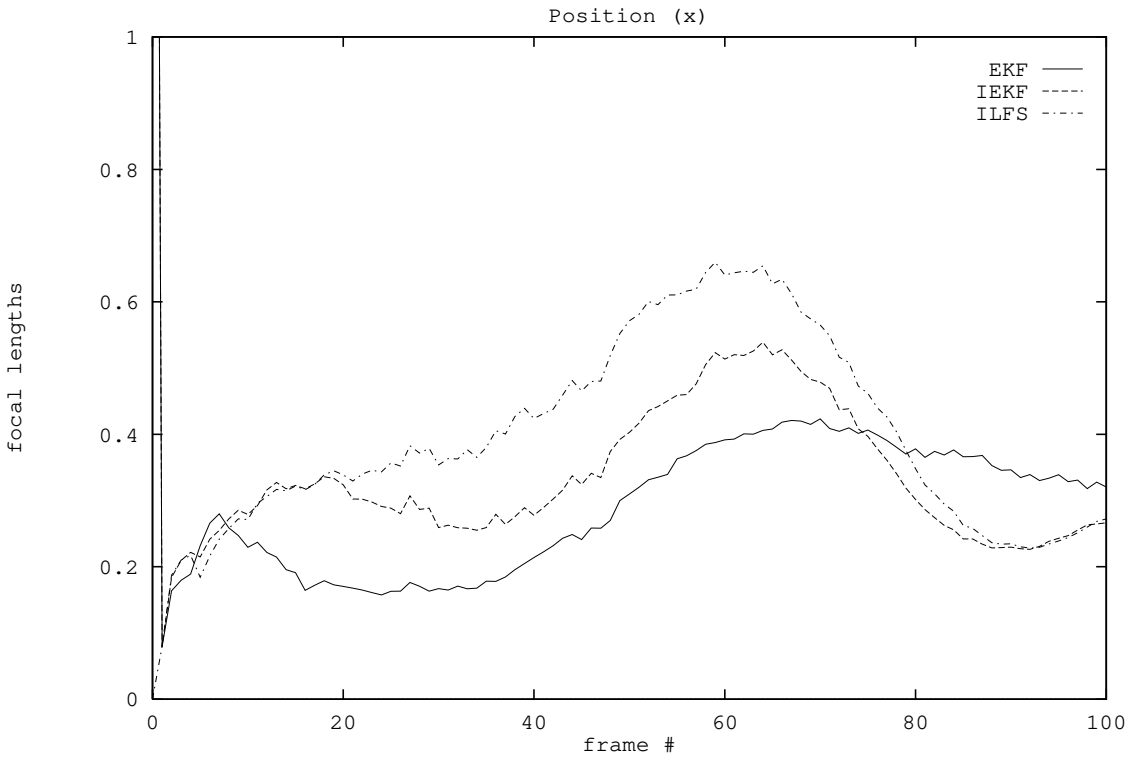


(g)

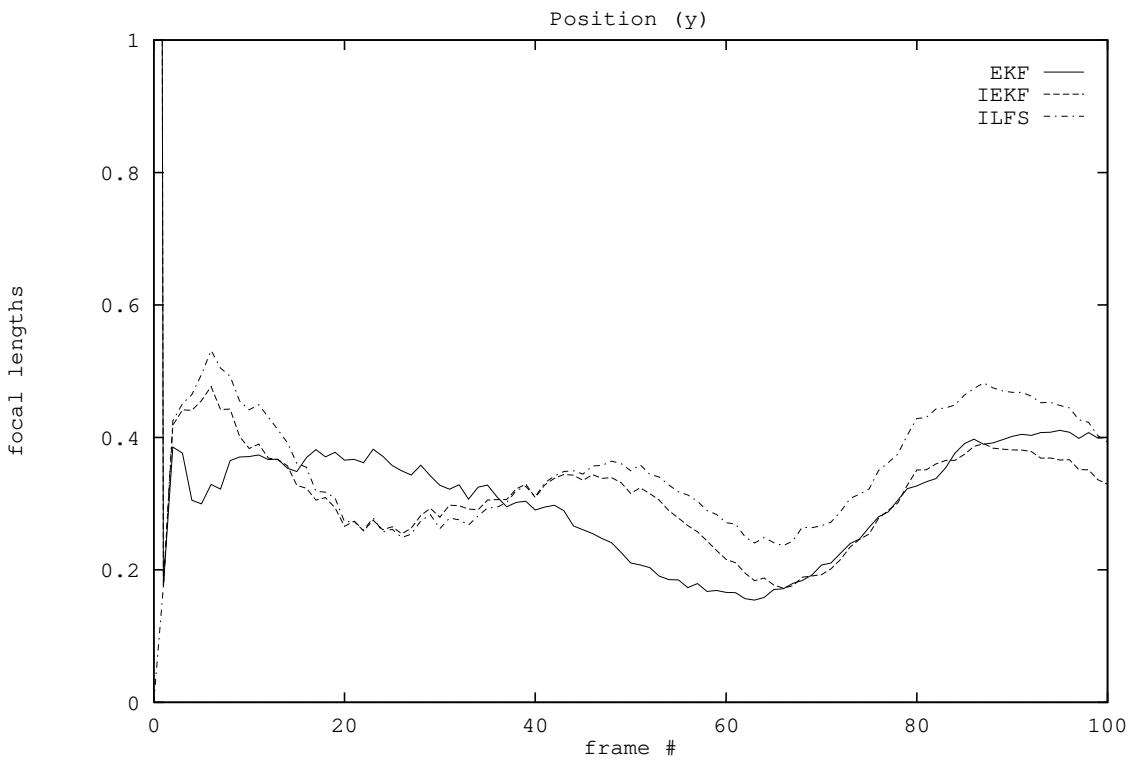


(h)

Figure 3.4: Experiment 1: Results of Monte Carlo simulations (bias)

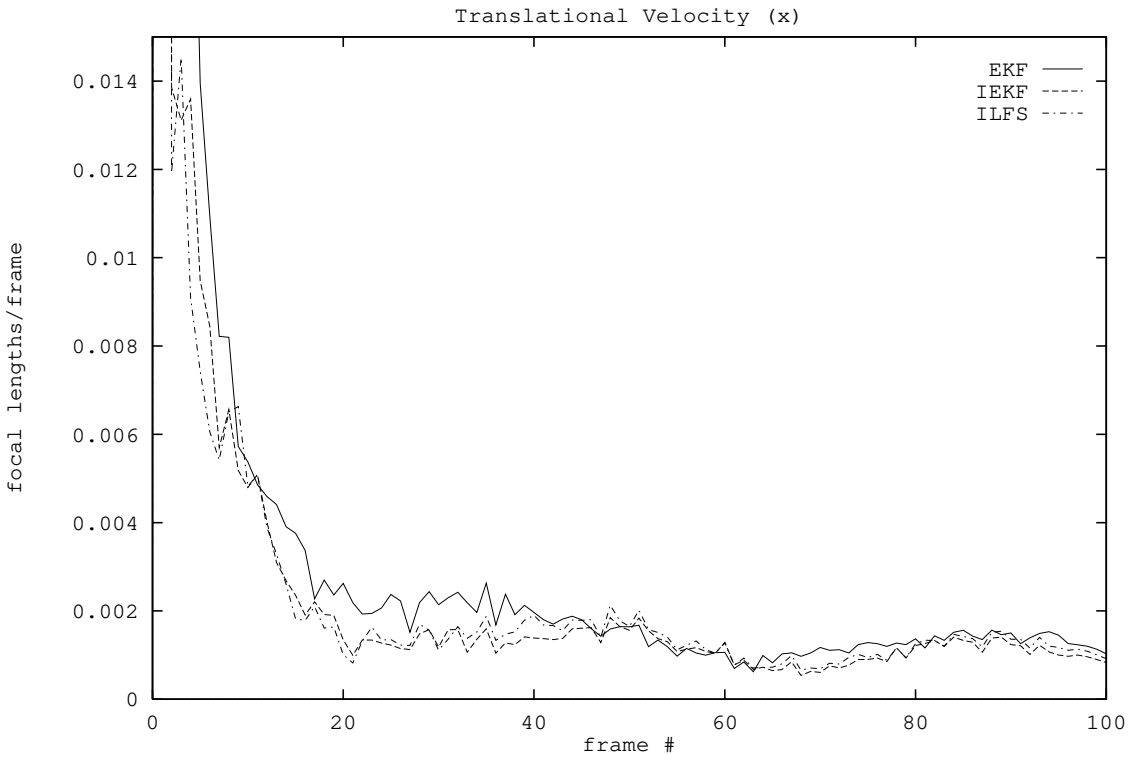


(a)

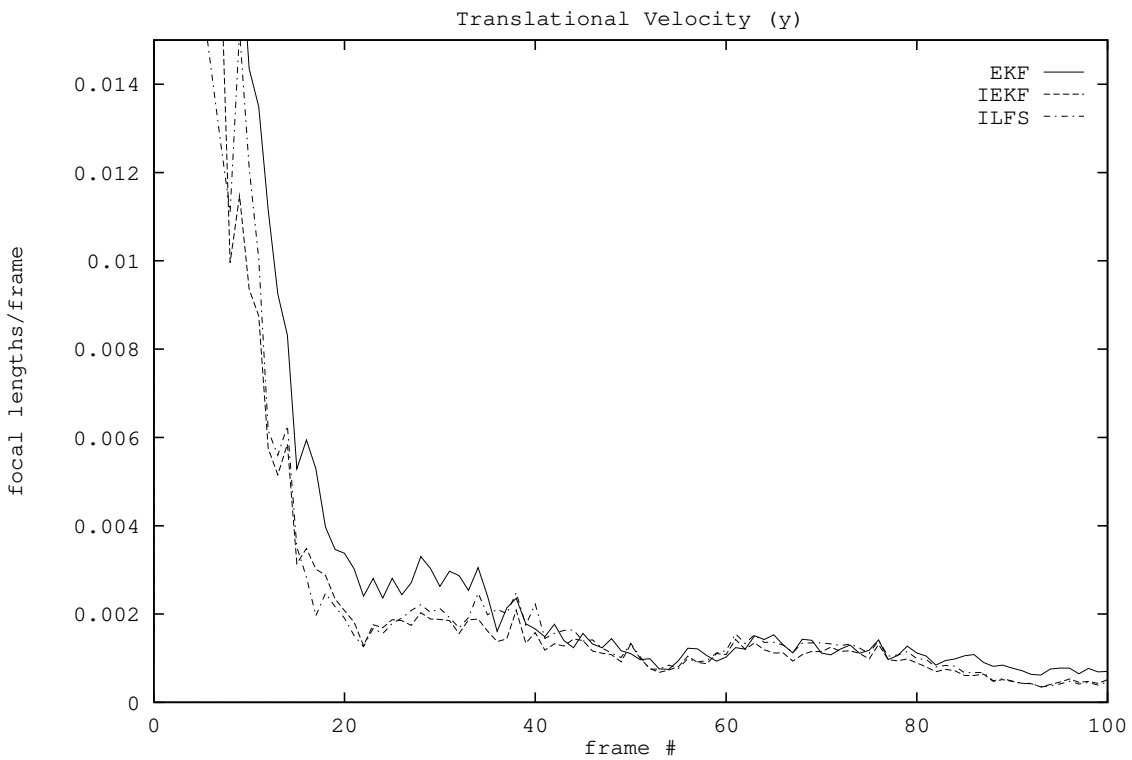


(b)

Figure 3.5: Experiment 1: Results of Monte Carlo simulations (variance)

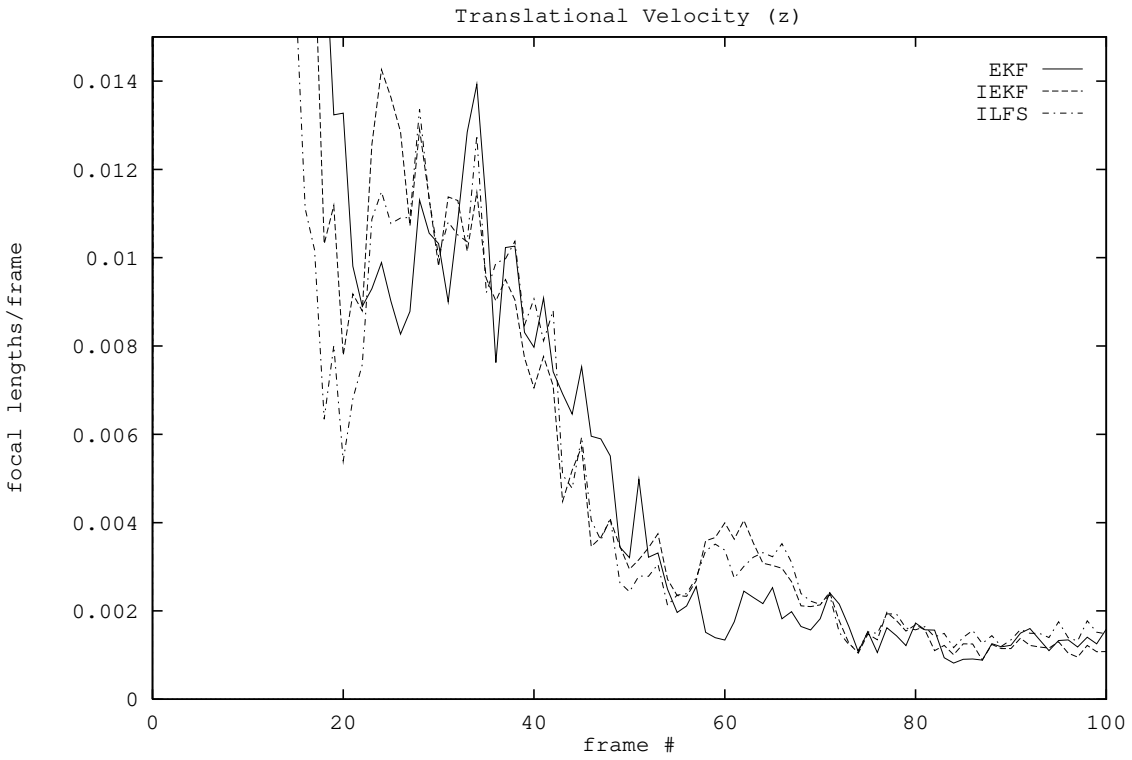


(c)

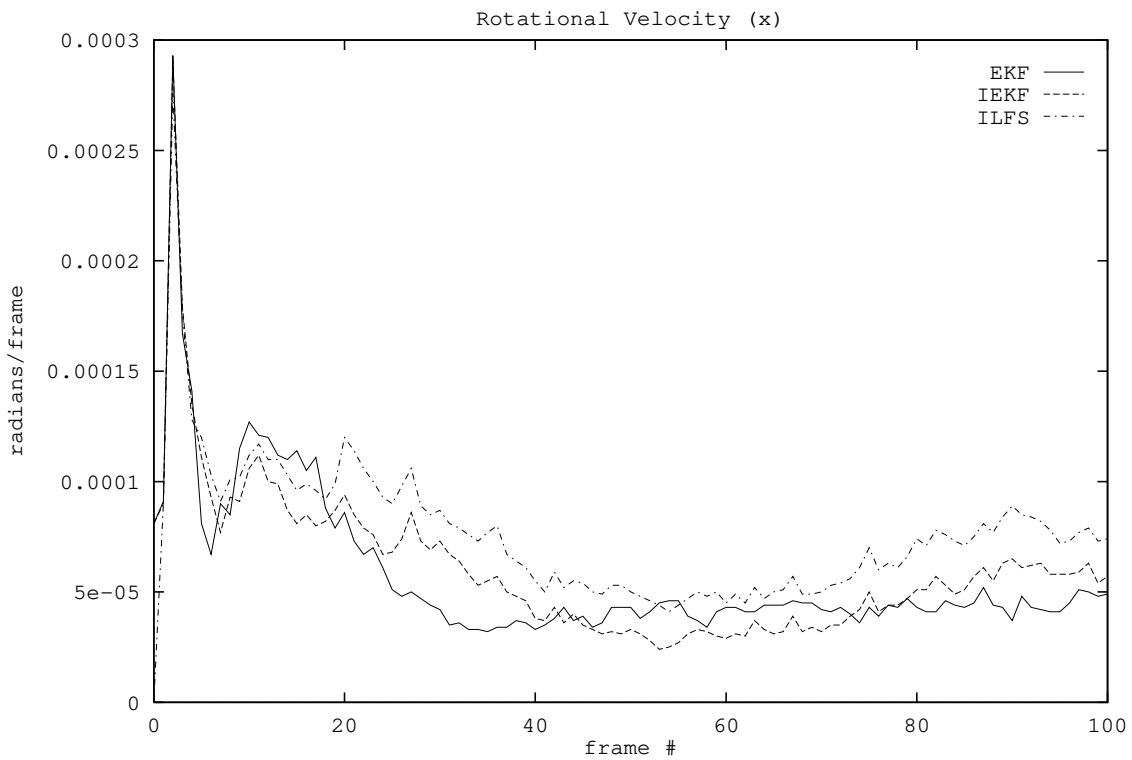


(d)

Figure 3.5: Experiment 1: Results of Monte Carlo simulations (variance)

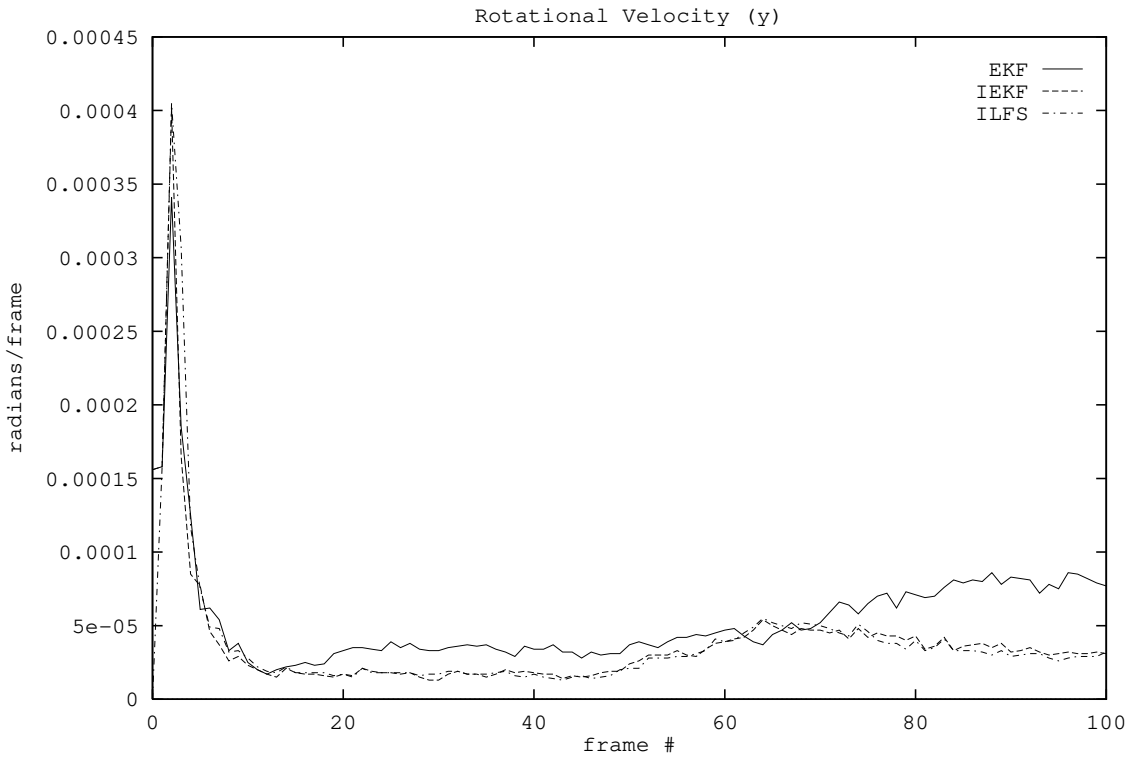


(e)

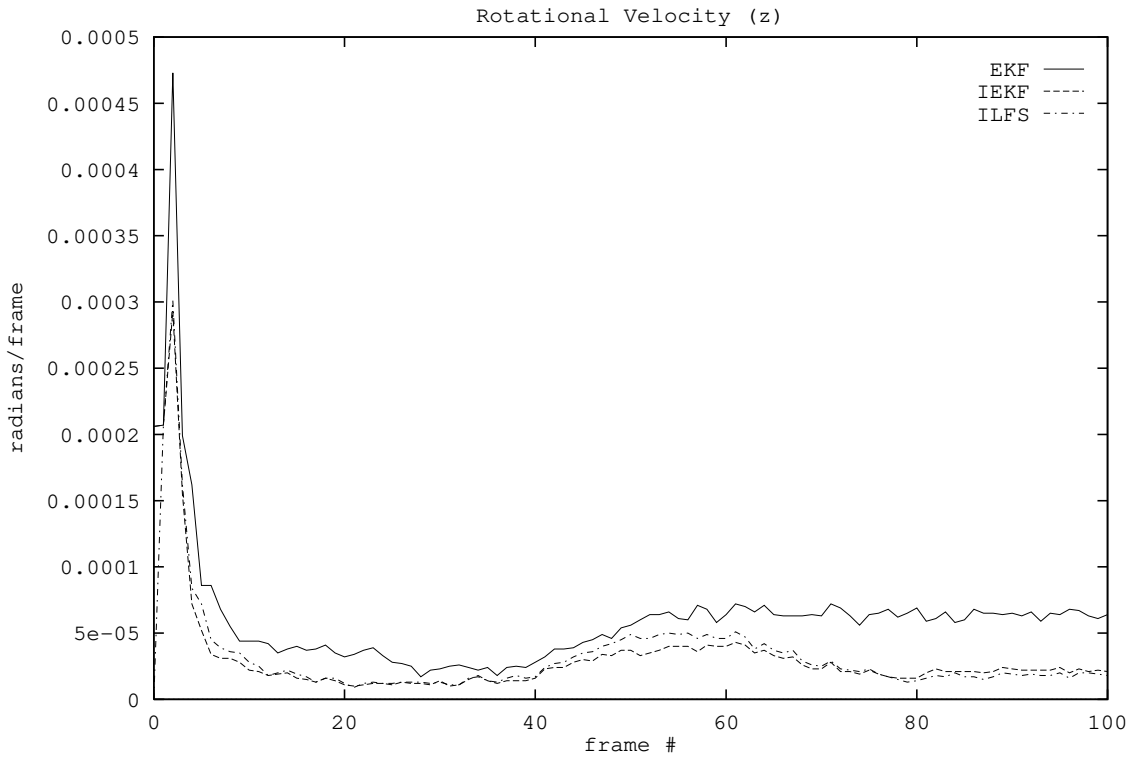


(f)

Figure 3.5: Experiment 1: Results of Monte Carlo simulations (variance)



(g)



(h)

Figure 3.5: Experiment 1: Results of Monte Carlo simulations (variance)

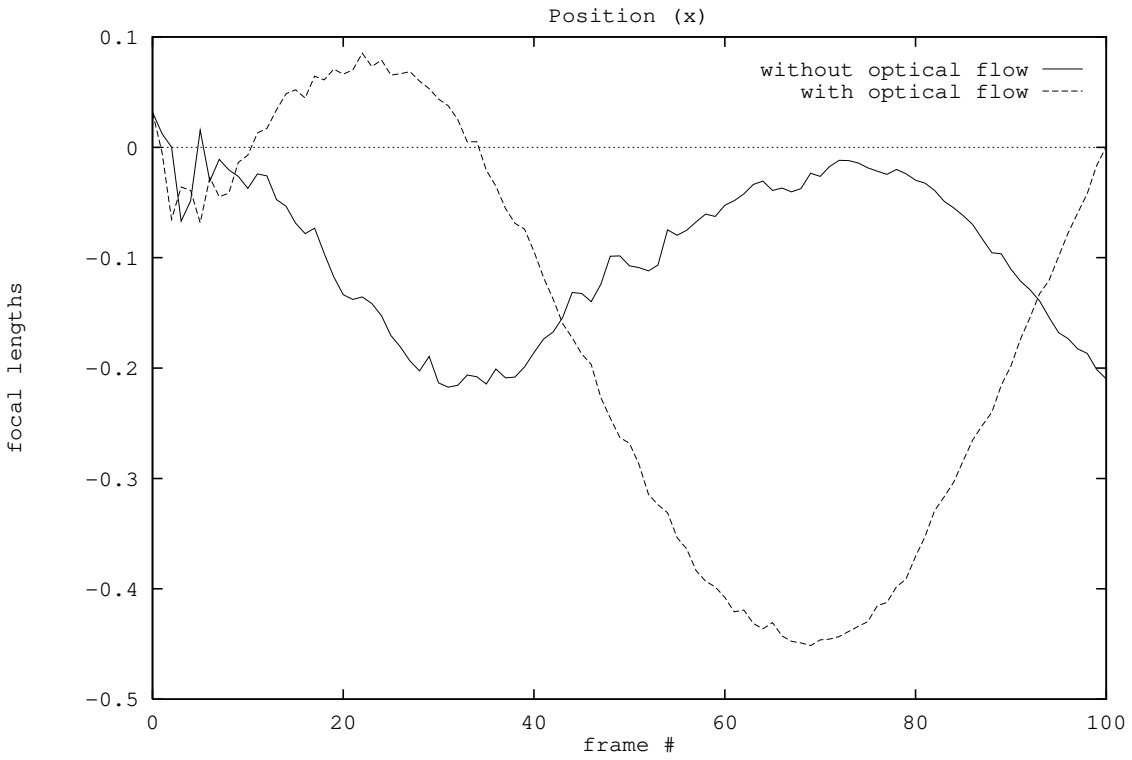
3.4.3 Experiment 2: Adding Optical Flow

The purpose of this experiment is to investigate the effect of adding optical flow measurements to a purely feature-based algorithm. The main difference between the BCC algorithm and the new hybrid feature/flow-based algorithm developed in Section 3.2 is that the former uses only image plane position measurements while the latter makes use of both image plane position measurements and optical flow measurements. The motion parameters used in this experiment are the same as those used in Experiment 1. The average absolute error in position measurements is 1.6 pixels and the average relative error in optical flow measurements is 15.95%.

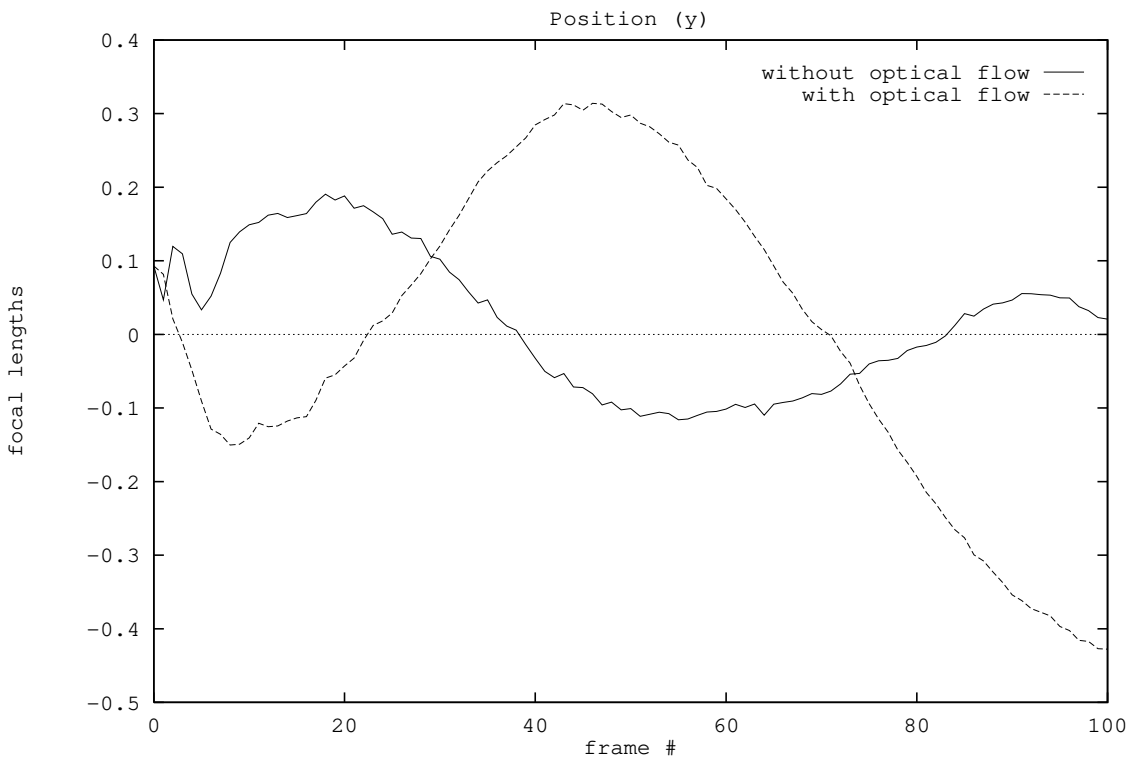
The simulation results are shown in Figures 3.6 (bias) and 3.7 (variance), with the solid line representing the estimates obtained by the BCC algorithm and the dashed line representing the estimates obtained using the new hybrid feature/flow-based algorithm. All three filters were used but only results of the EKF are shown because it performed better than the IEKF and ILFS. Not surprisingly, the improvements achieved by the hybrid algorithm can be seen predominantly in the velocity states. In the translational velocity states, improvements can be observed mainly during the filter initialization stage (the first 20 frames) where the bias error is significantly reduced. Improvements are even more obvious in the rotational velocity states: while the BCC algorithm diverges, the hybrid algorithm converges quickly and stays "locked on" to the true values.

One might argue that the large initial estimation errors produced by the BCC algorithm in the translational velocity states can be reduced or removed by simply increasing the number of Monte Carlo trials. However, this is *not* the case, as one can see in Figure 3.8 (where the results of averaging 30, 100, and 1000 trials are shown).

We have also experimented with different amount of noise added to the optical flow measurements. In particular, when the average relative error is about 100%, the two algorithms yield almost identical results in the translational velocity states, although the hybrid algorithm still offers better performance in the rotational velocity states.

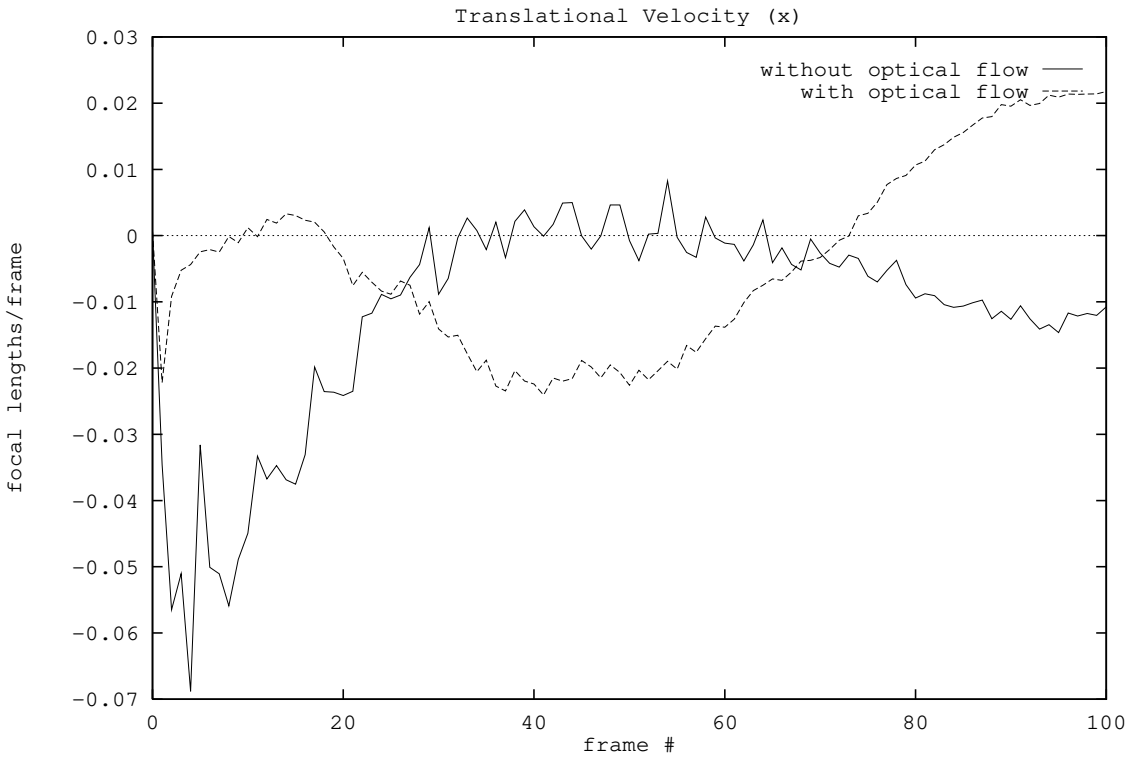


(a)

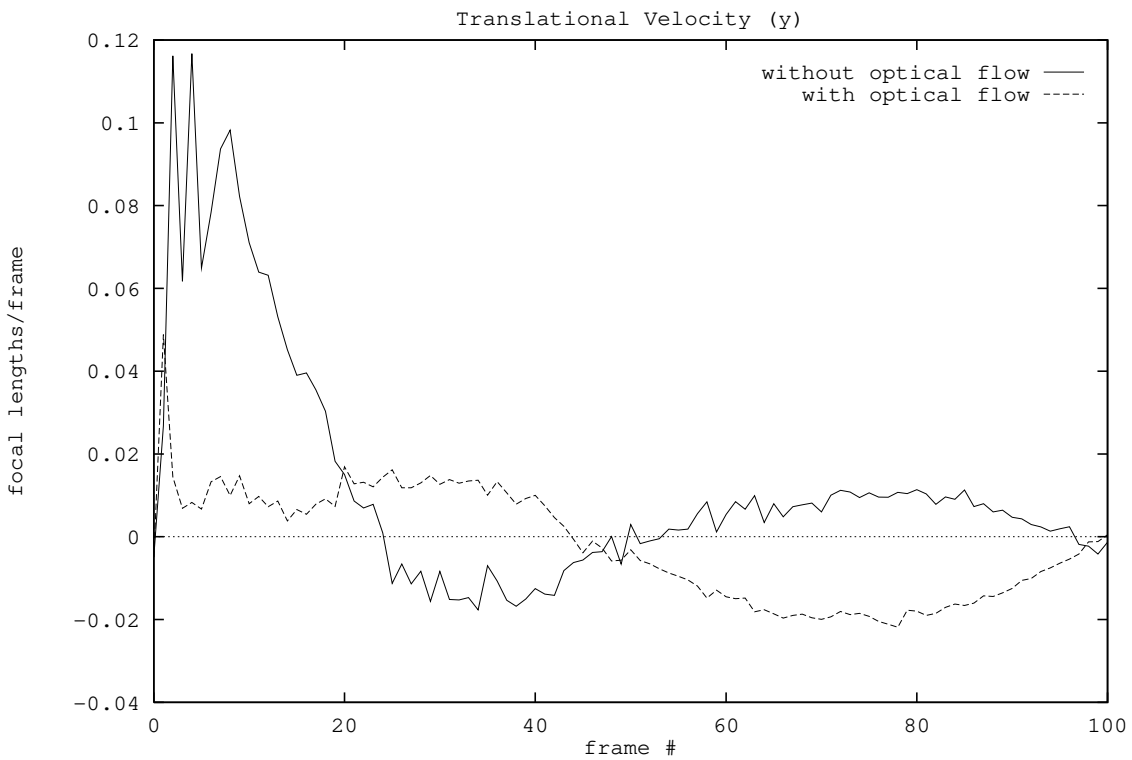


(b)

Figure 3.6: Experiment 2: Results of Monte Carlo simulations (bias)

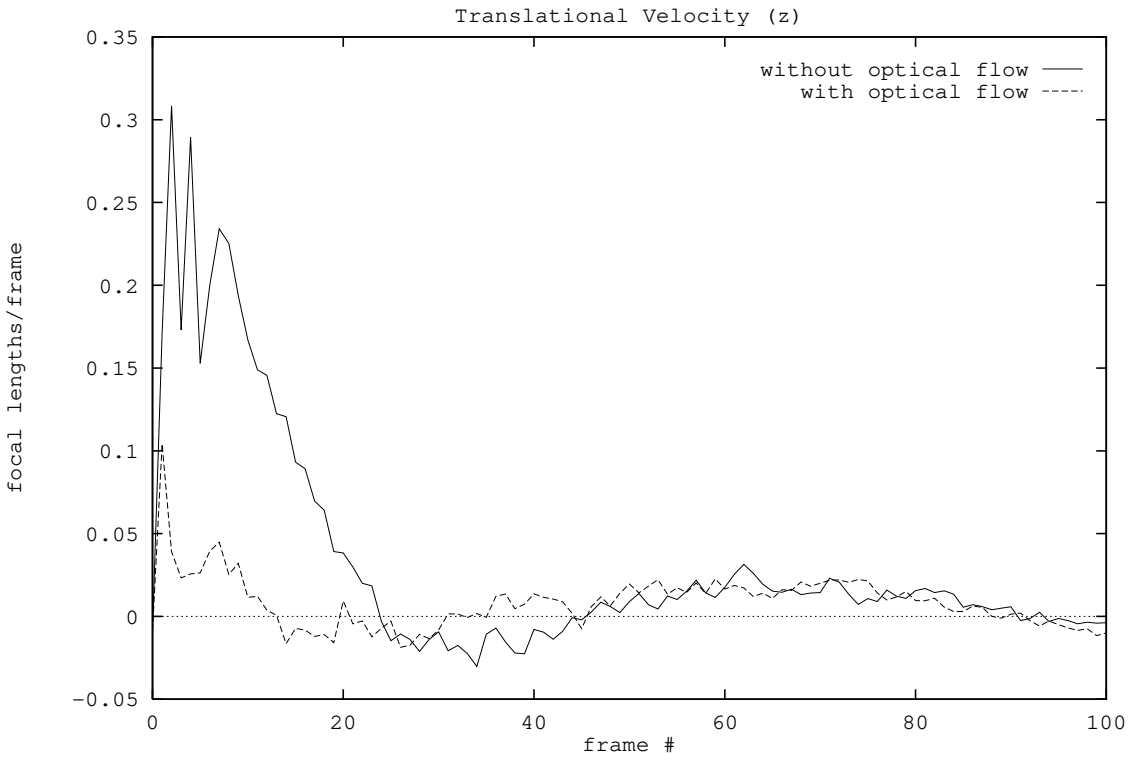


(c)

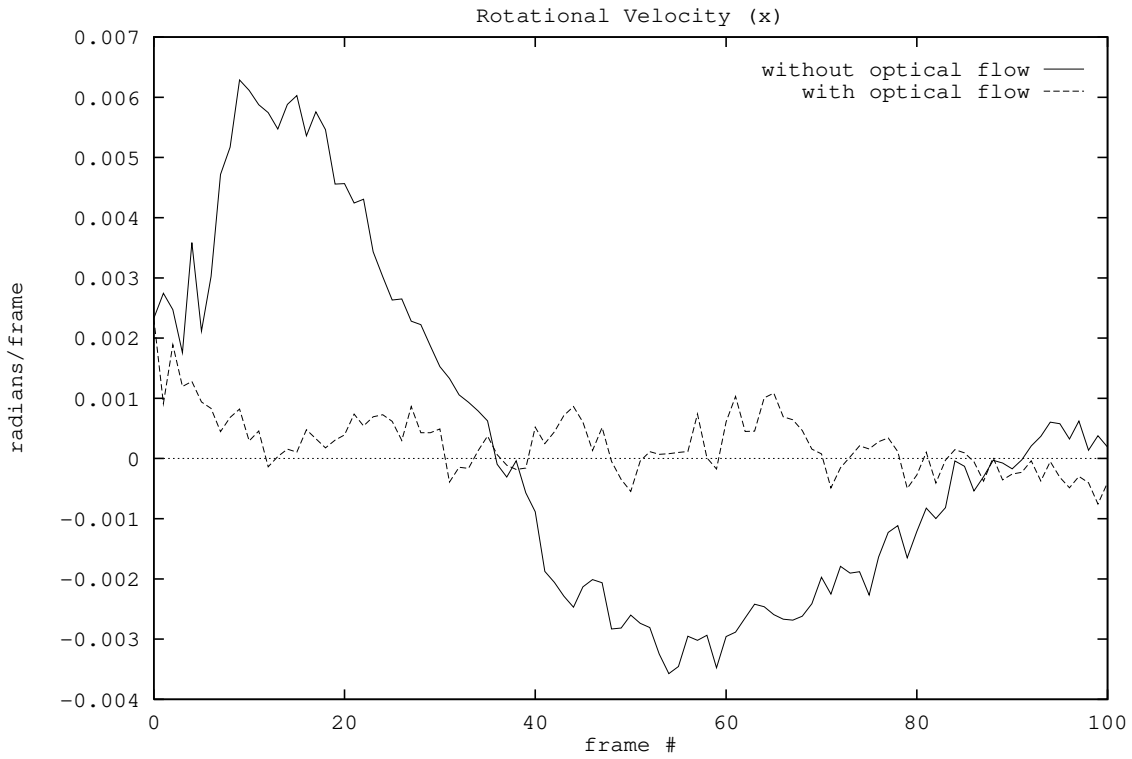


(d)

Figure 3.6: Experiment 2: Results of Monte Carlo simulations (bias)

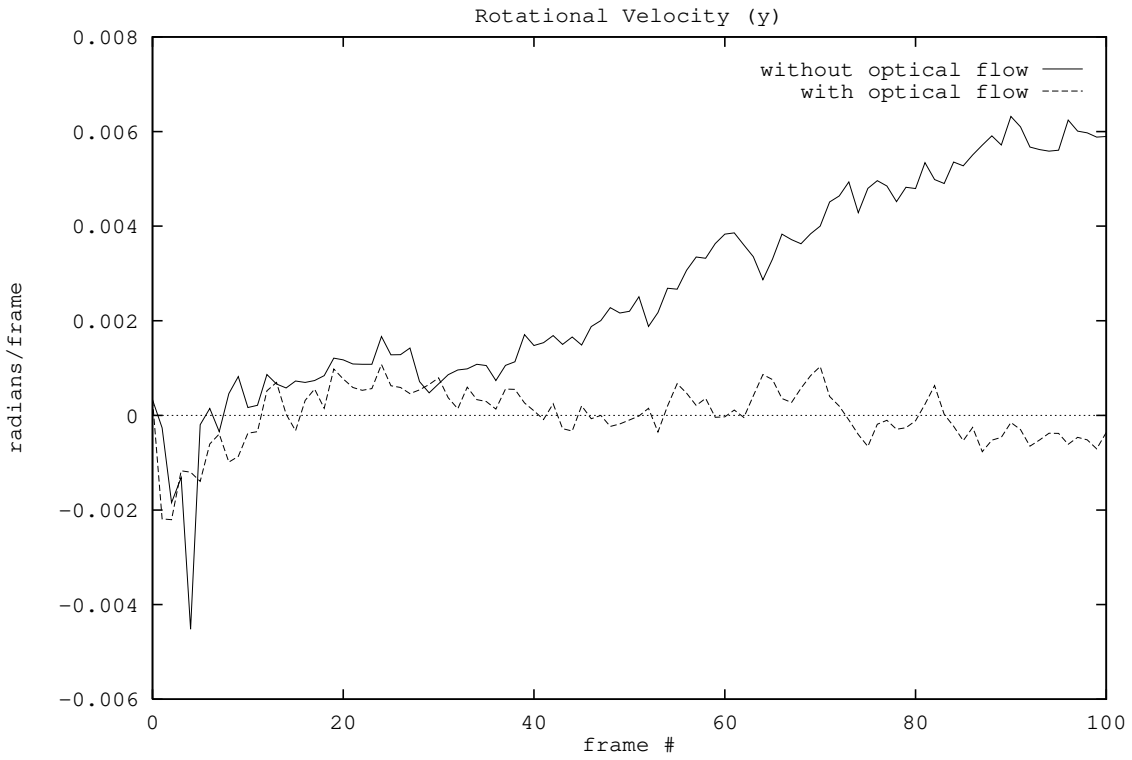


(e)

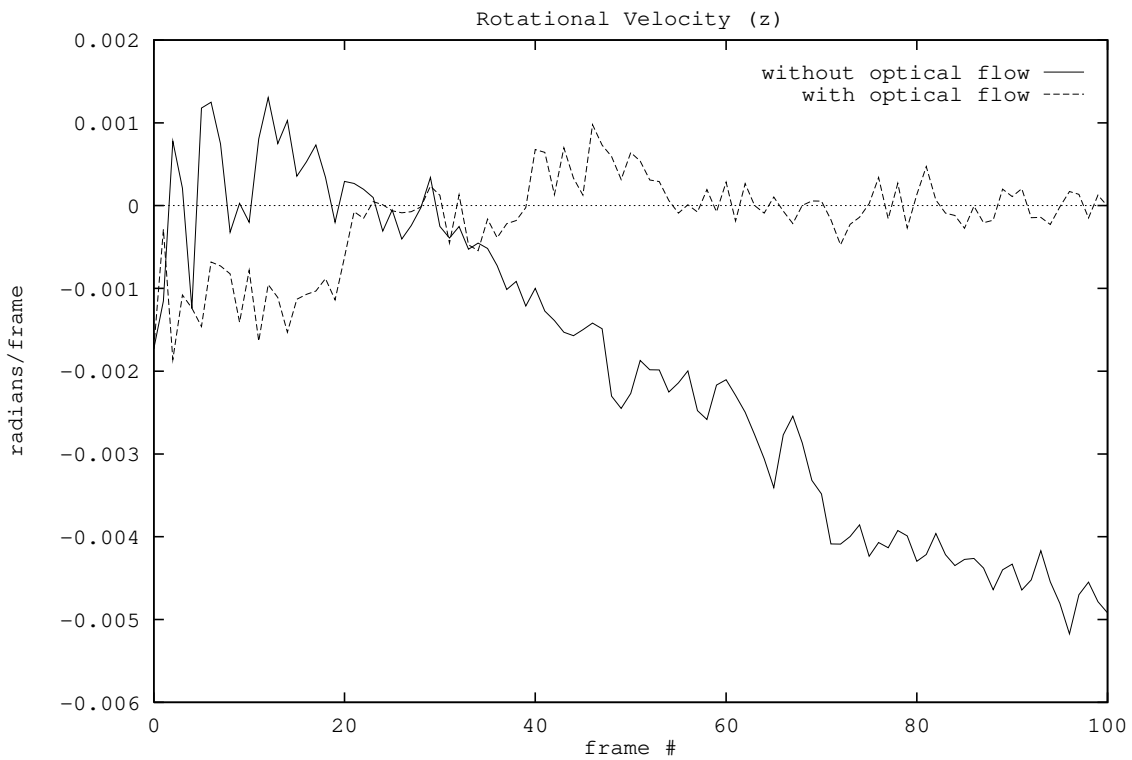


(f)

Figure 3.6: Experiment 2: Results of Monte Carlo simulations (bias)

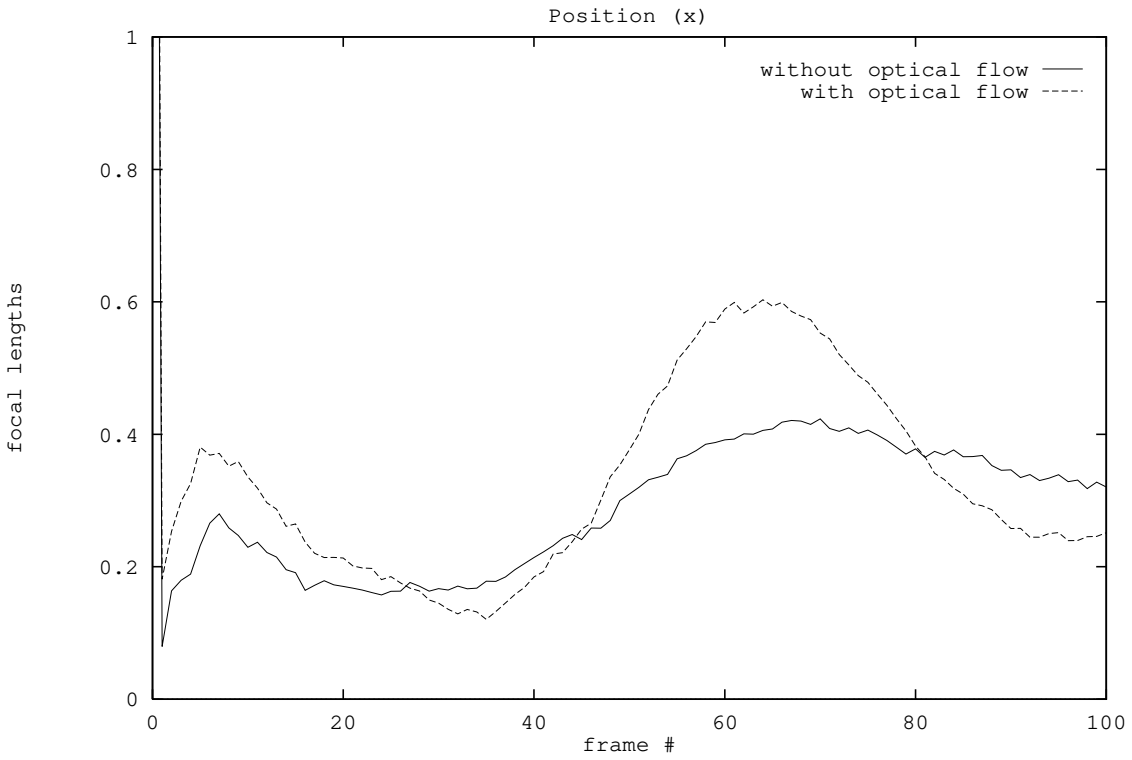


(g)

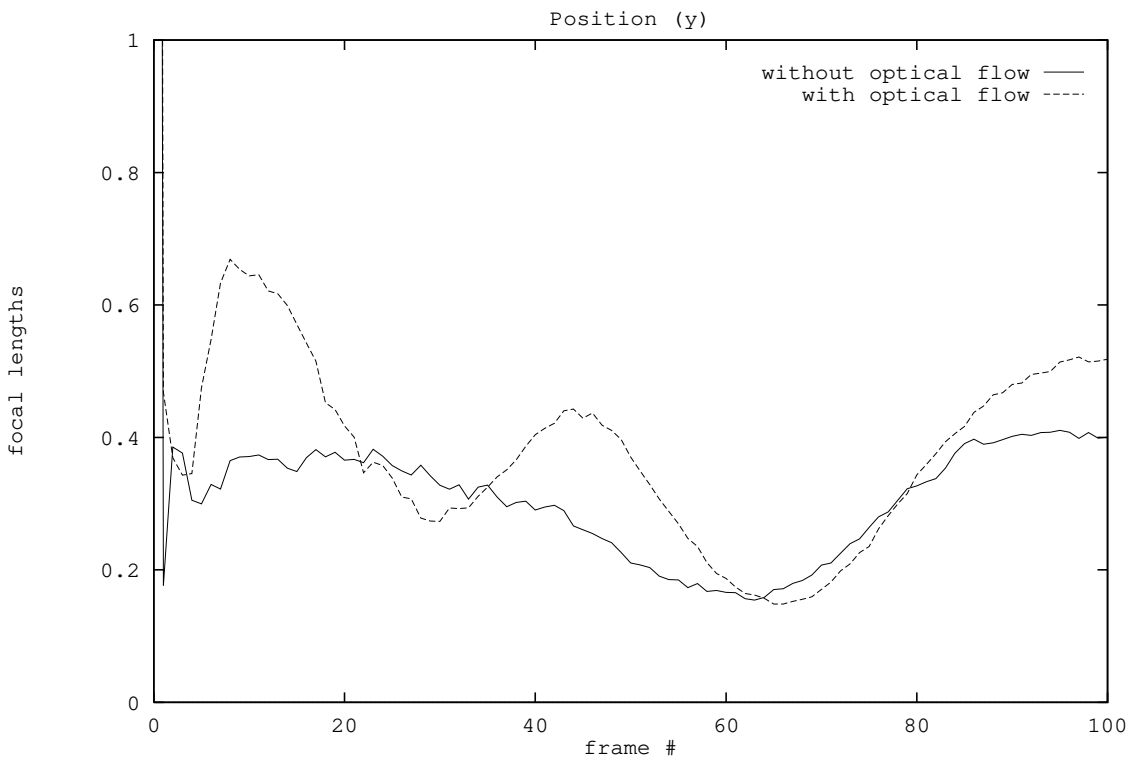


(h)

Figure 3.6: Experiment 2: Results of Monte Carlo simulations (bias)

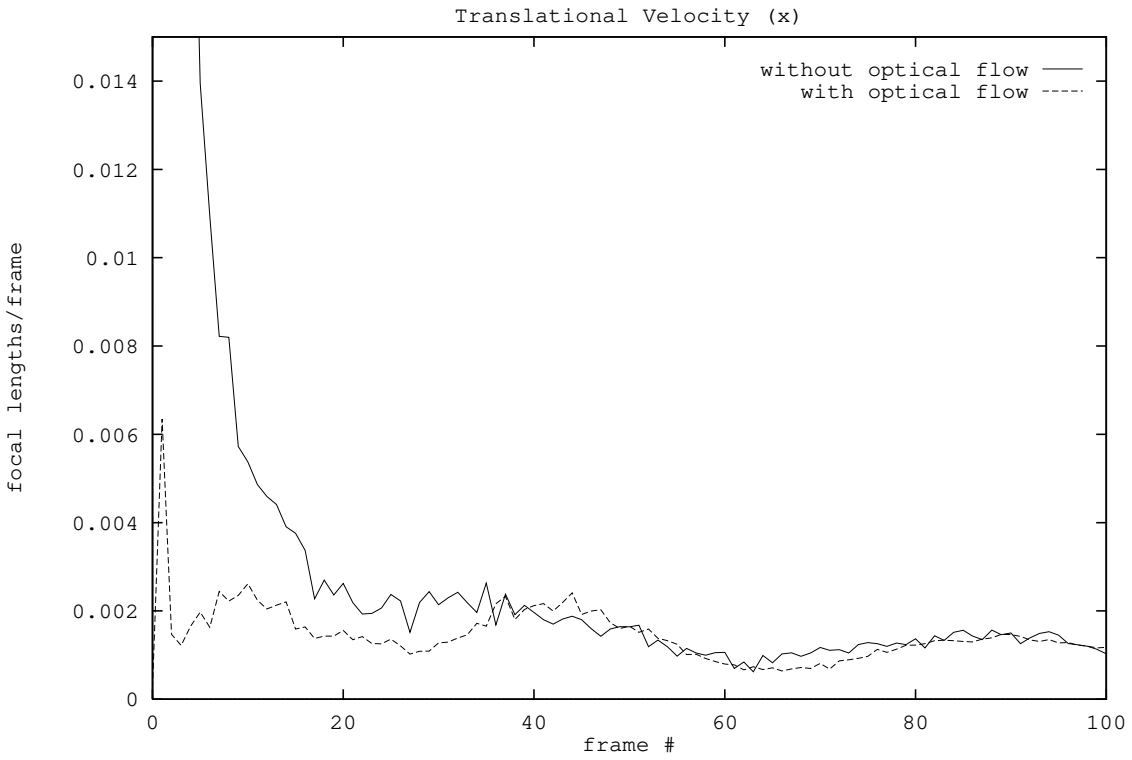


(a)

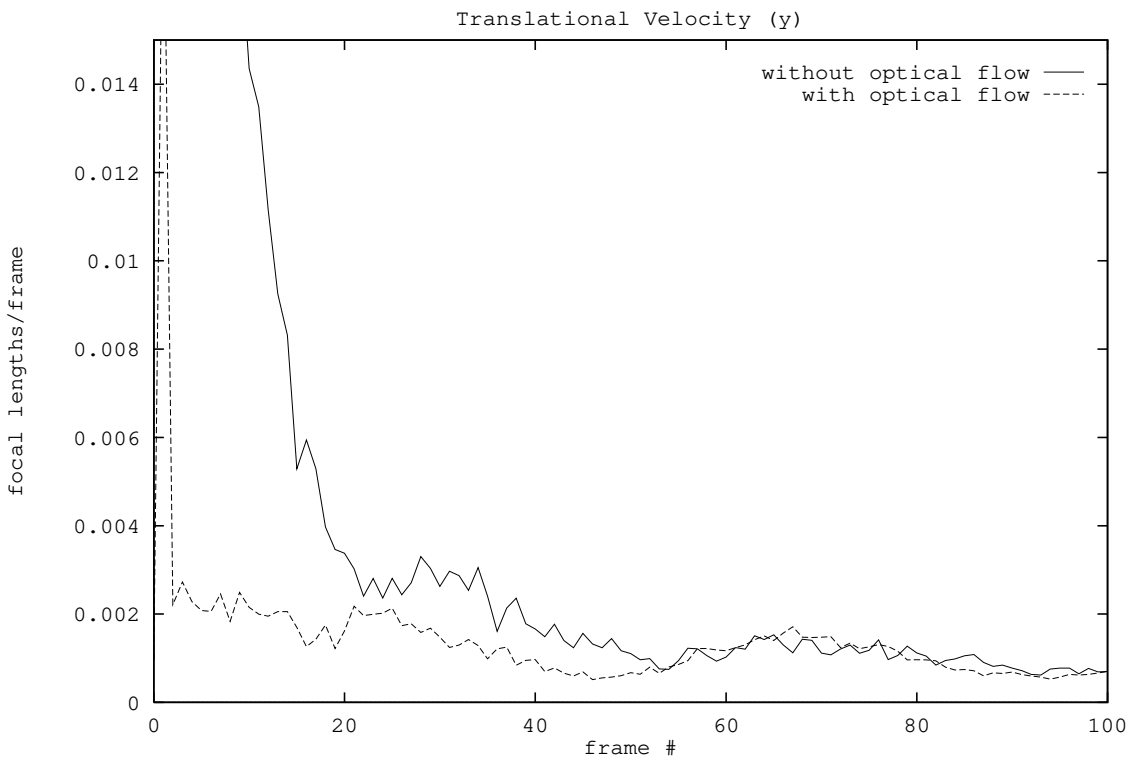


(b)

Figure 3.7: Experiment 2: Results of Monte Carlo simulations (variance)

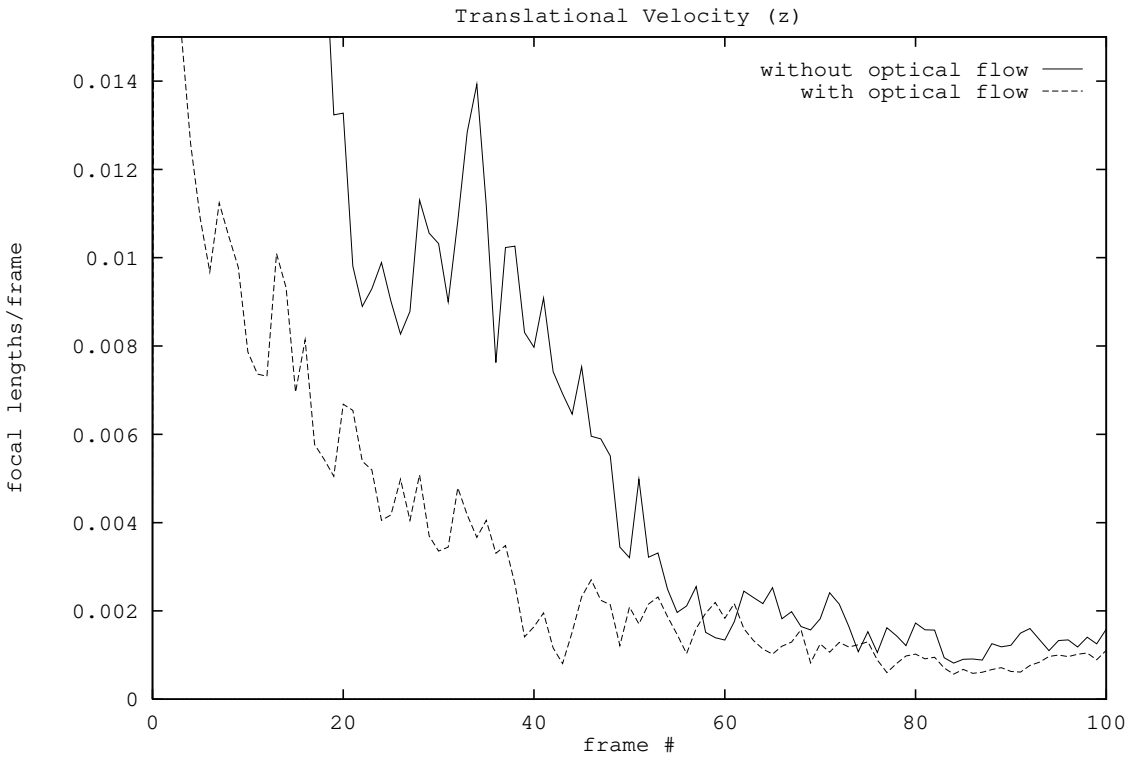


(c)

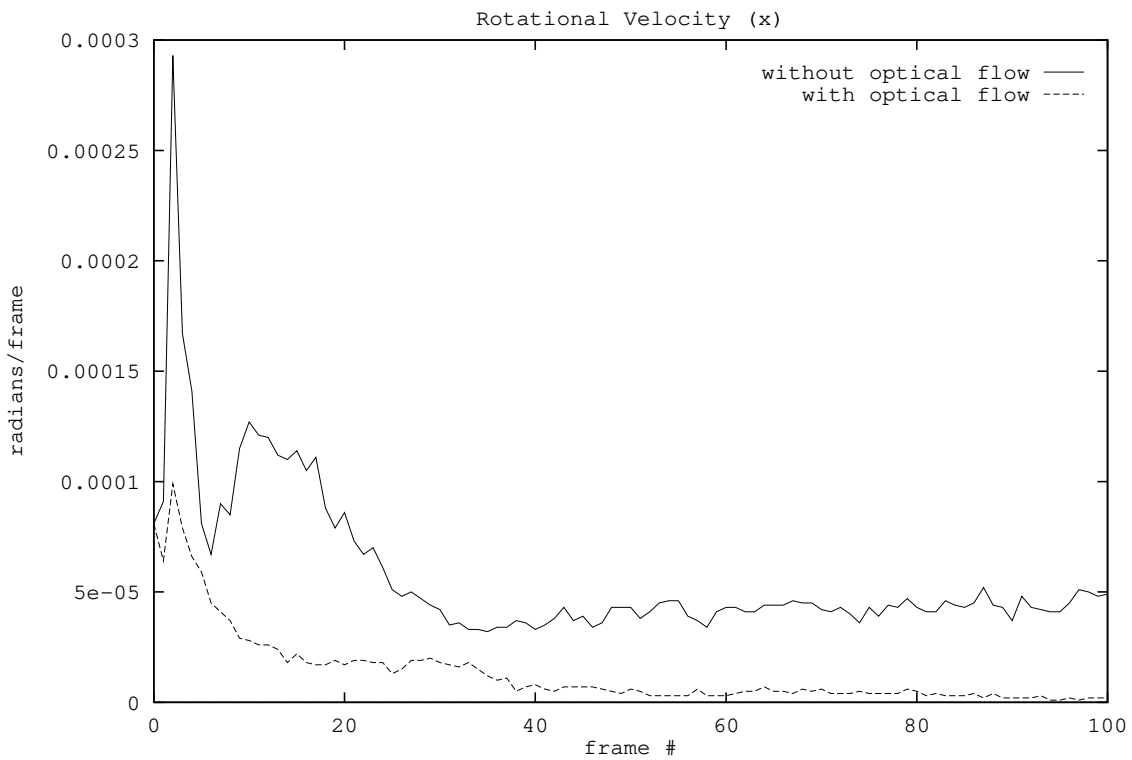


(d)

Figure 3.7: Experiment 2: Results of Monte Carlo simulations (variance)

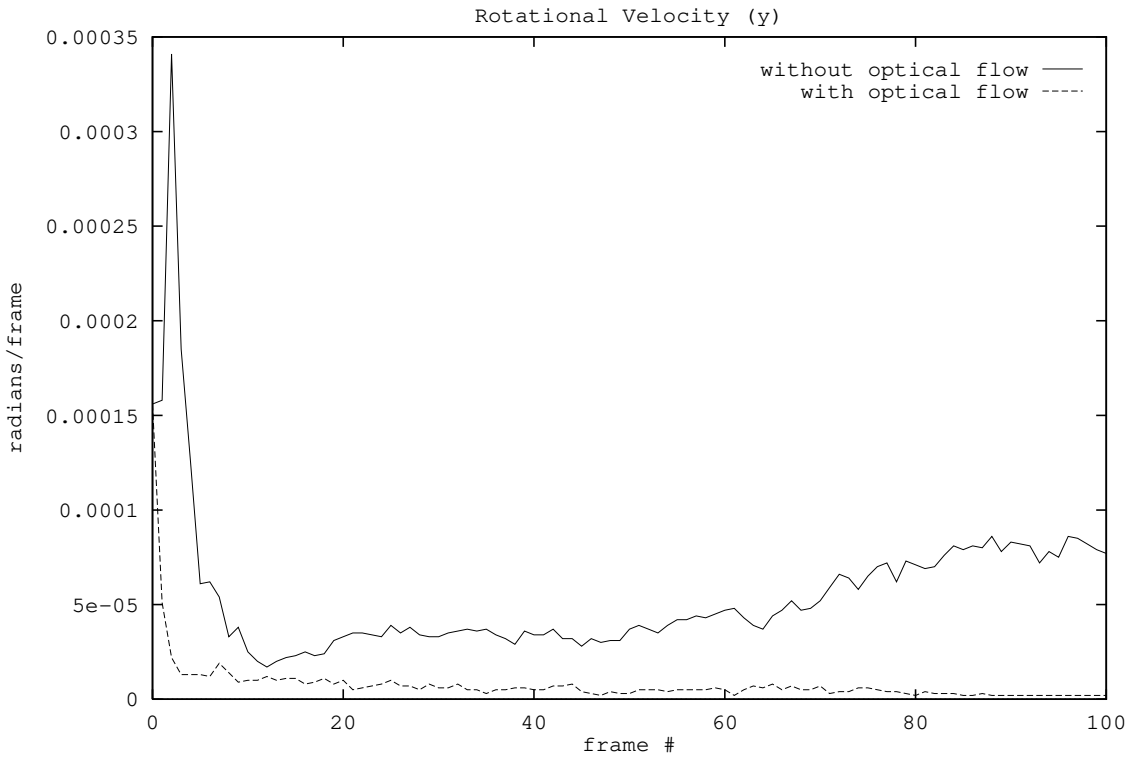


(e)

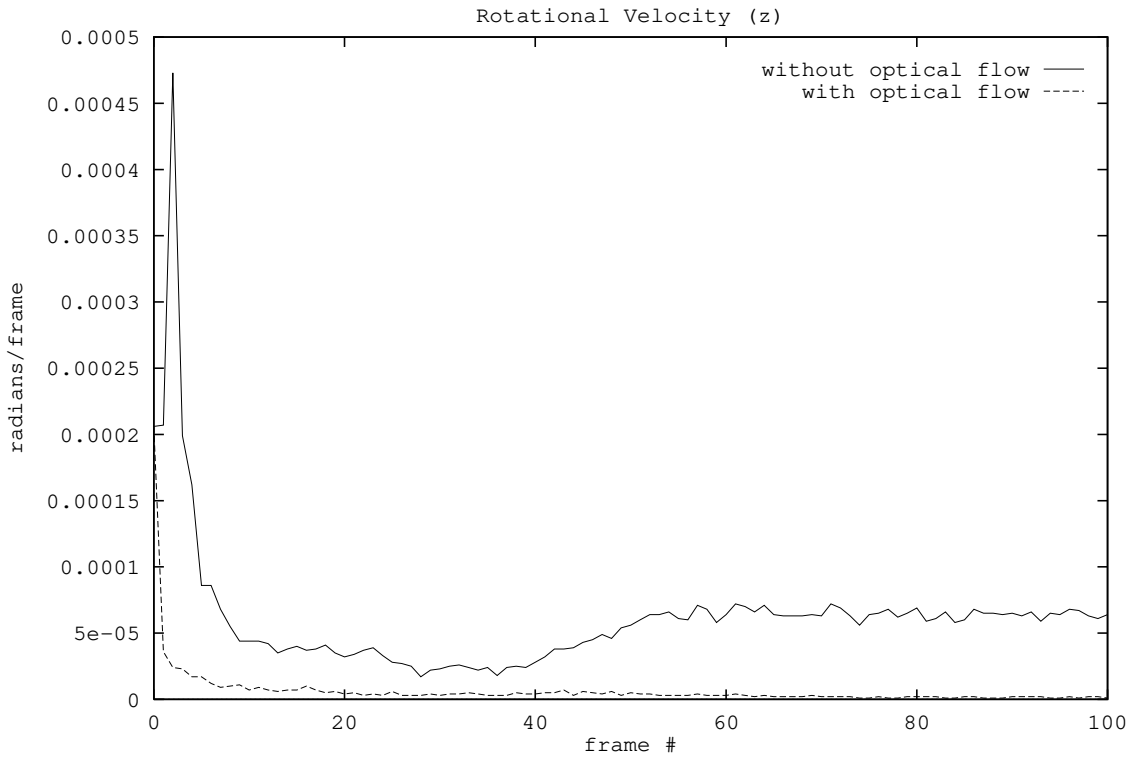


(f)

Figure 3.7: Experiment 2: Results of Monte Carlo simulations (variance)

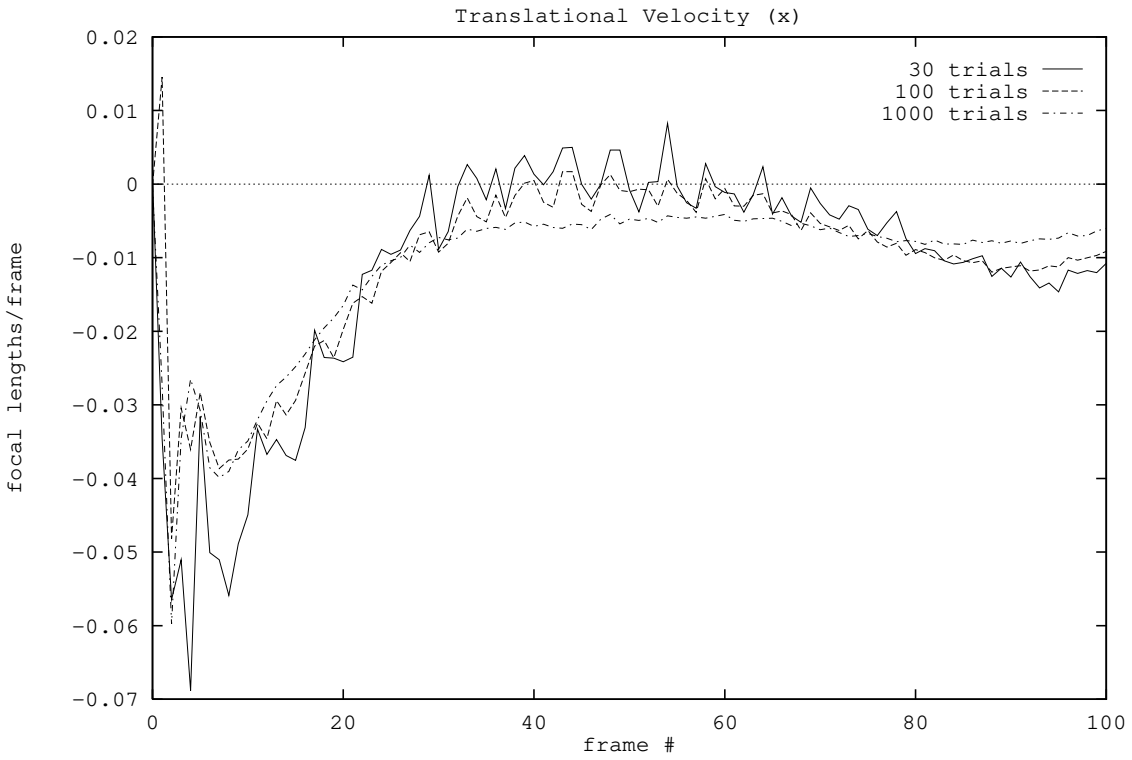


(g)

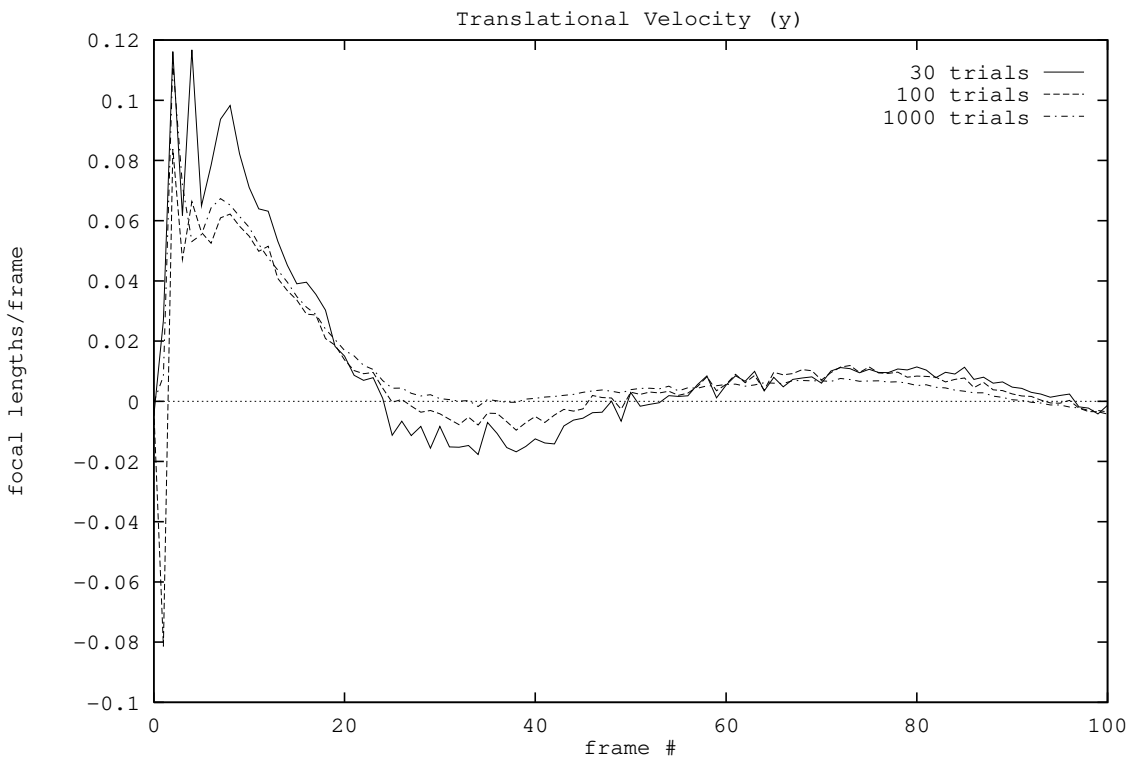


(h)

Figure 3.7: Experiment 2: Results of Monte Carlo simulations (variance)

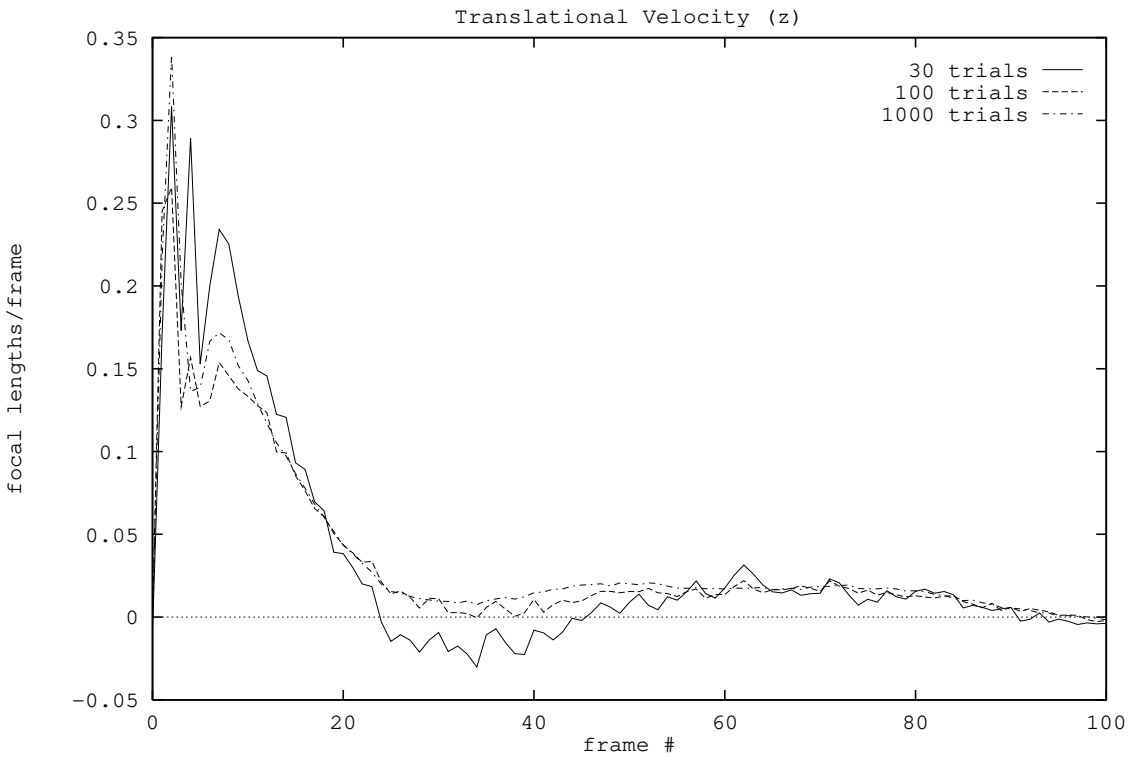


(a)



(b)

Figure 3.8: Experiment 2: Increasing the number of Monte Carlo trials (bias)



(c)

Figure 3.8: Experiment 2: Increasing the number of Monte Carlo trials (bias)

3.4.4 Experiment 3: Translational Acceleration

Although our model explicitly states that the object motion consists of constant translational and rotational velocities only, it is likely that we will encounter less-than-ideal situations in practice. This experiment examines the performance of the hybrid feature/flow-based algorithm in a situation where the model does not accurately describe the object motion. Specifically, we introduce translational acceleration to the object's motion. The position of the rotation centre, at any time t, is described by

$$\mathbf{s}_R(t) = \mathbf{s}_R(t_0) + (t - t_0)\mathbf{T} + \frac{1}{2}\mathbf{a}(t - t_0)^2. \tag{3.45}$$

The average absolute error in position measurements is 1.6 pixels and the average relative error in optical flow measurements is 11.93%. Table 3.2 summarizes the motion parameters used. The motion of the object, as seen in the image plane, is shown in Figure 3.9.

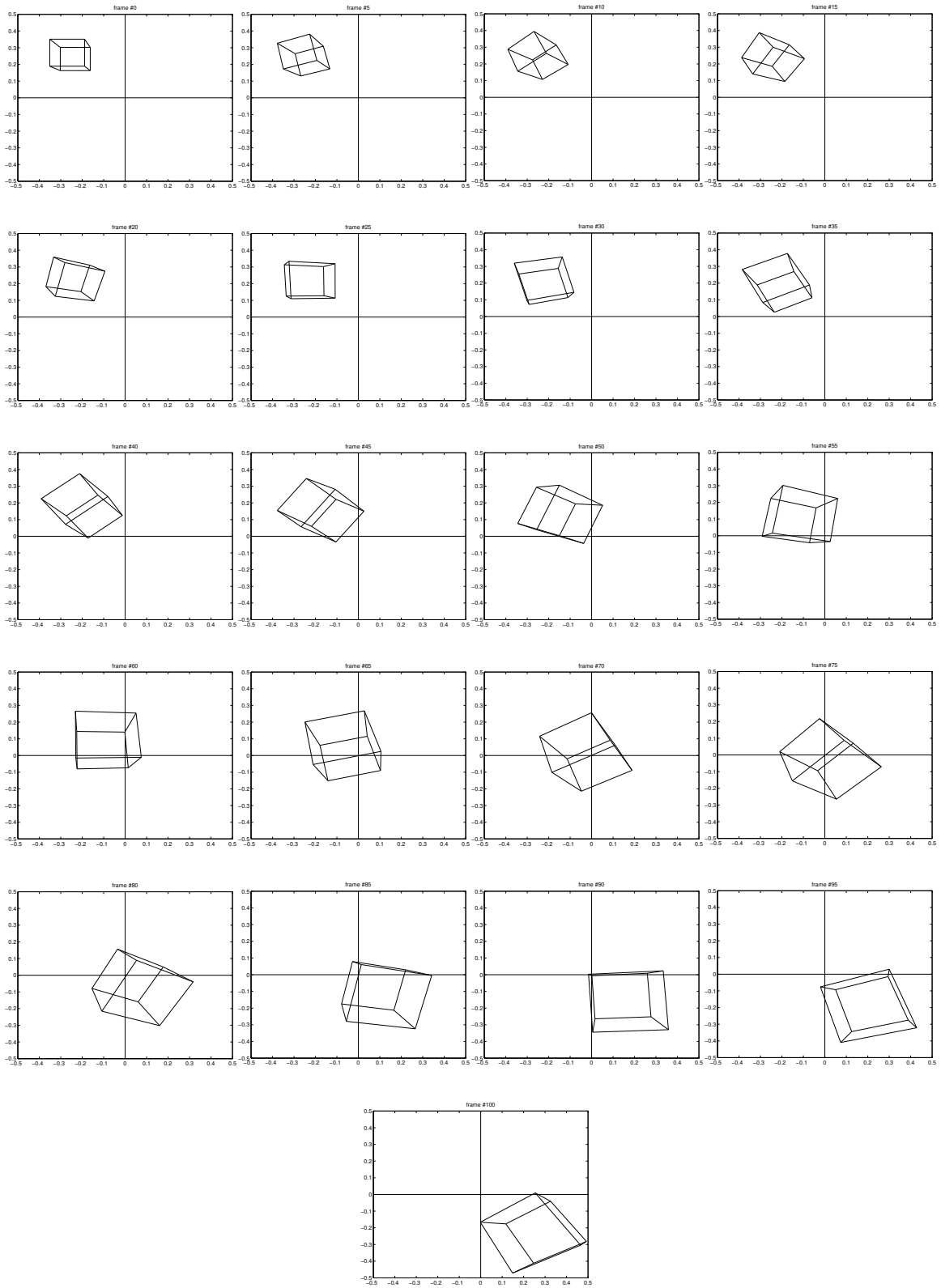


Figure 3.9: Experiment 3: Object motion in image plane

initial position $\mathbf{s}_{\mathrm{R}}(t_0)$ (FL)	$[-5 \quad 5 \quad 20]^T$
translational velocity $\mathbf{T}$ (FL/frame)	$[0.05 \quad -0.0625 \quad -0.2]^T$
rotational velocity $\boldsymbol{\omega}$ (radians/frame)	$[0.03 \quad 0.04 \quad 0.05]^T$
translational acceleration $\mathbf{a}$ (FL/frame^2)	$[0.0005 \quad -0.0002 \quad 0.002]^T$

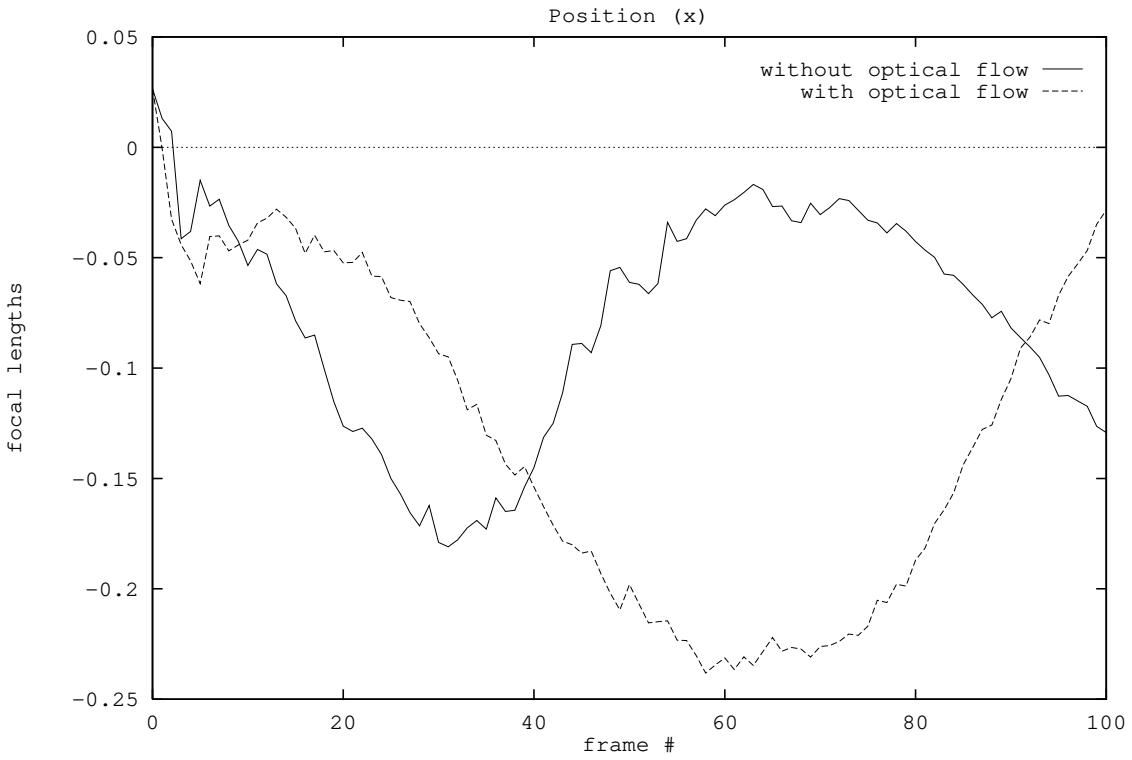
Table 3.2: Motion parameters for Experiment 3

The simulation results are shown in Figures 3.10 (bias) and 3.11 (variance). In this experiment, the results presented are those obtained using the EKF, which performed better than the IEKF and ILFS. As in the previous experiment, adding optical flow measurements reduces the estimation error in the translational velocity states during the initial frames. In the rotational velocity states, the hybrid algorithm consistently yields better estimates than the BCC algorithm.

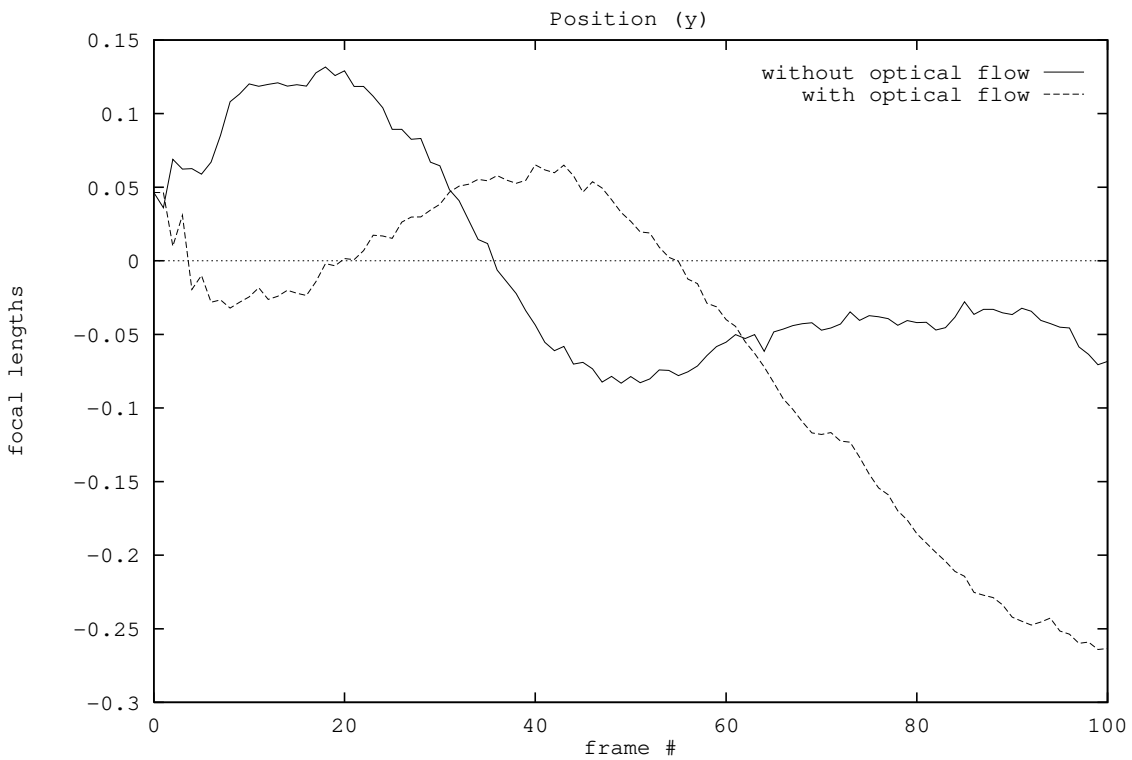
One could always account for the translational acceleration explicitly in the motion model (see [42, 97], for example). However, adding acceleration to the translational motion model will increase the size of the state vector as well as the state propagation matrix F. As a result, overall execution speed will, inevitably, be affected. Our simulation results indicate that proper use of process noise appears to be adequate in the case of translational acceleration. Therefore, depending on one's application, computational efficiency might be more important than the improved accuracy that can potentially be achieved by a more accurate motion model.

3.4.5 Experiment 4: Abrupt Change

In this experiment, the object is allowed to deviate from the model more drastically than the previous experiment. It changes its trajectory *abruptly* after 50 frames, which clearly violates our plant model. Instead of monitoring the magnitude of the residual error or innovation sequence and adjusting the process noise accordingly [5, 11], we simply use a fixed amount of process noise to handle the abrupt change. The average absolute error in position measurements is 1.6 pixels and the average relative error in optical flow measurements is 10.10%. Table 3.3 summarizes the motion parameters

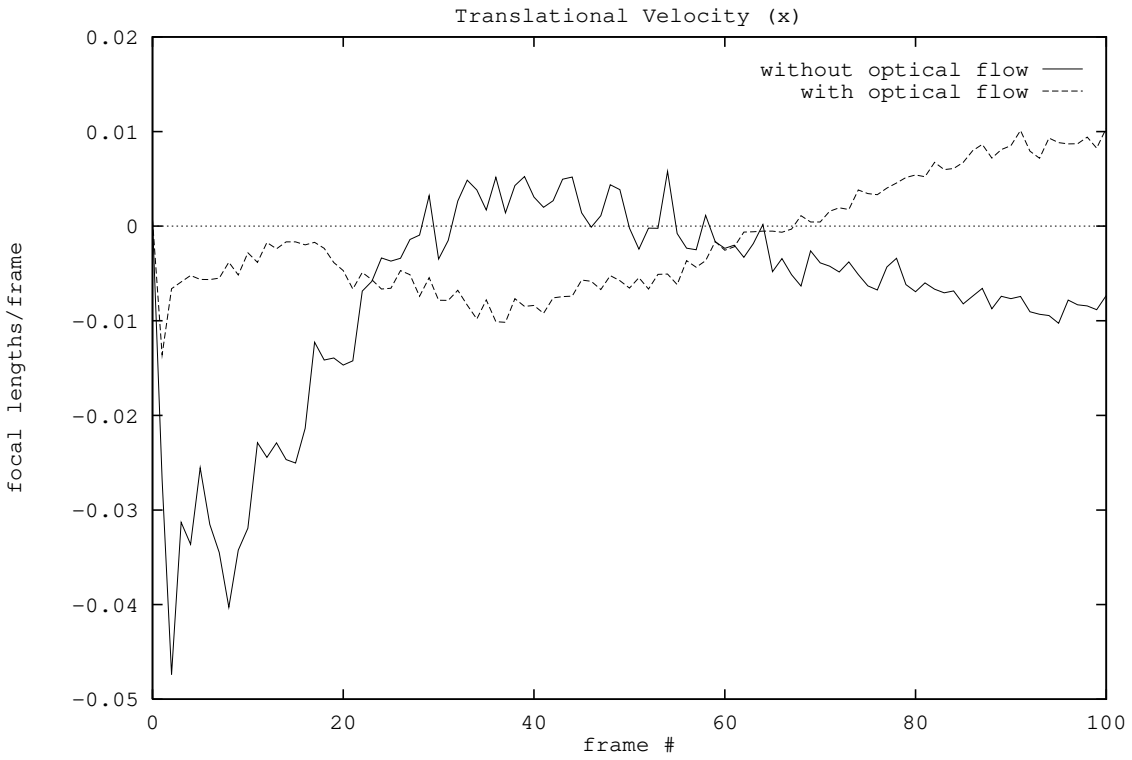


(a)

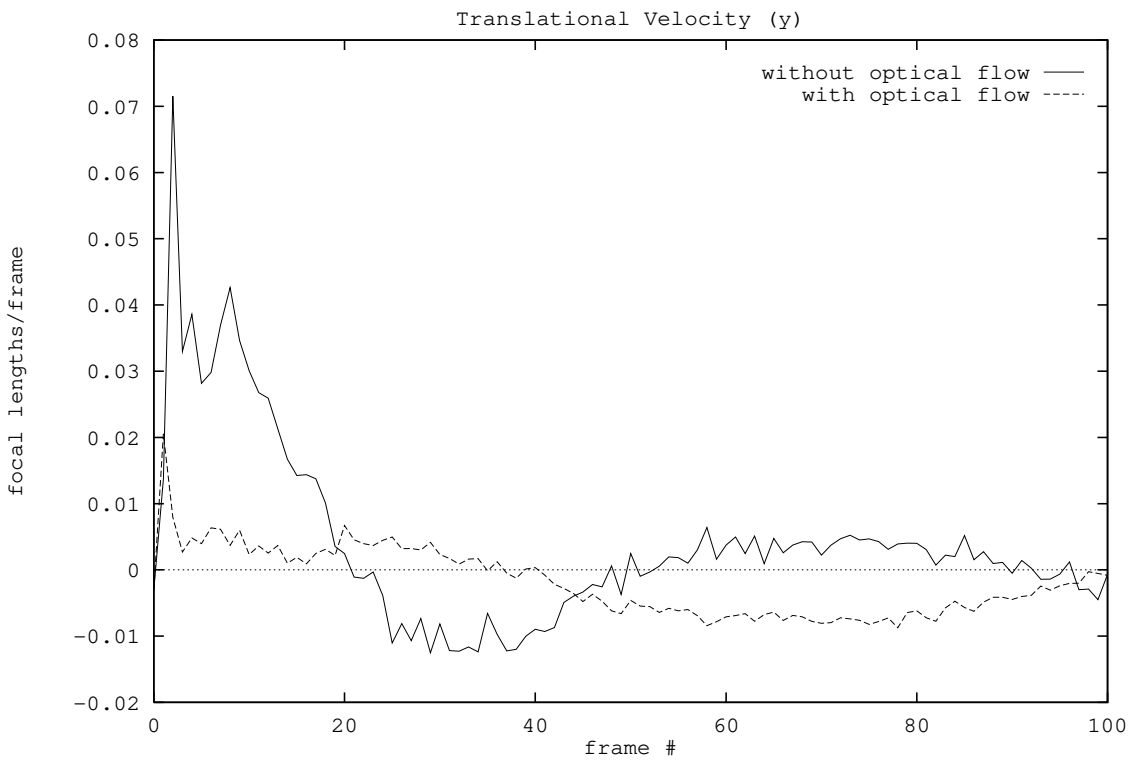


(b)

Figure 3.10: Experiment 3: Results of Monte Carlo simulations (bias)

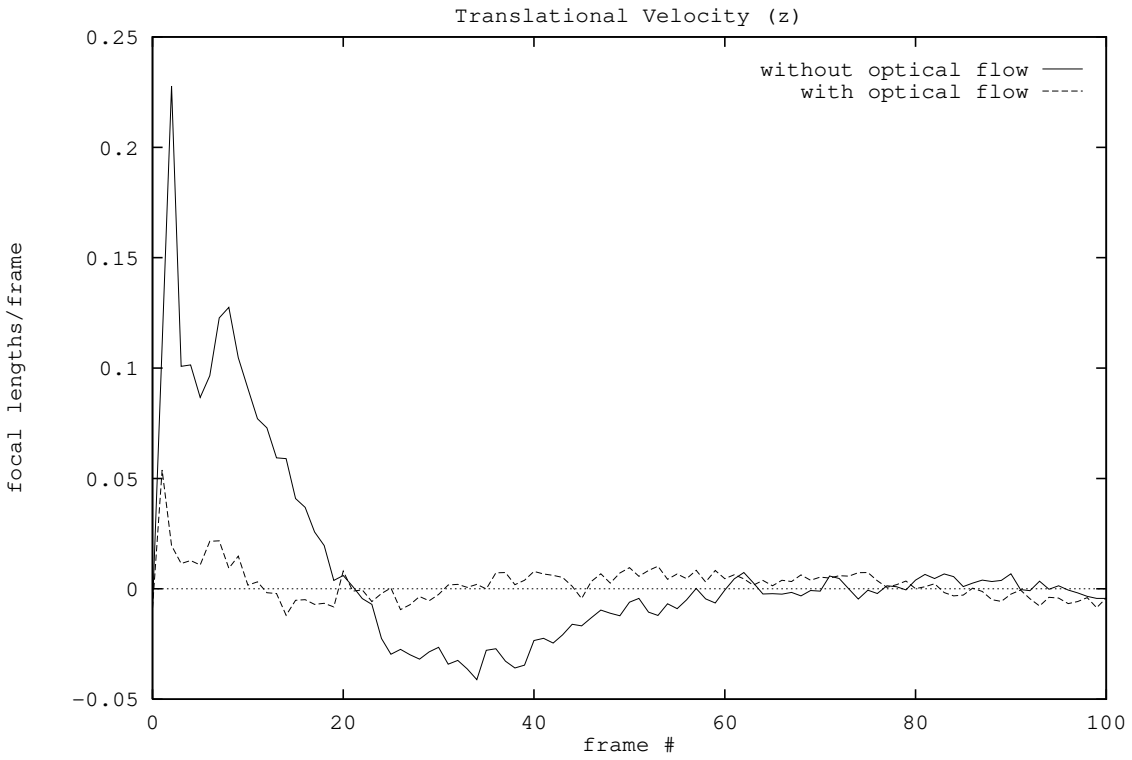


(c)

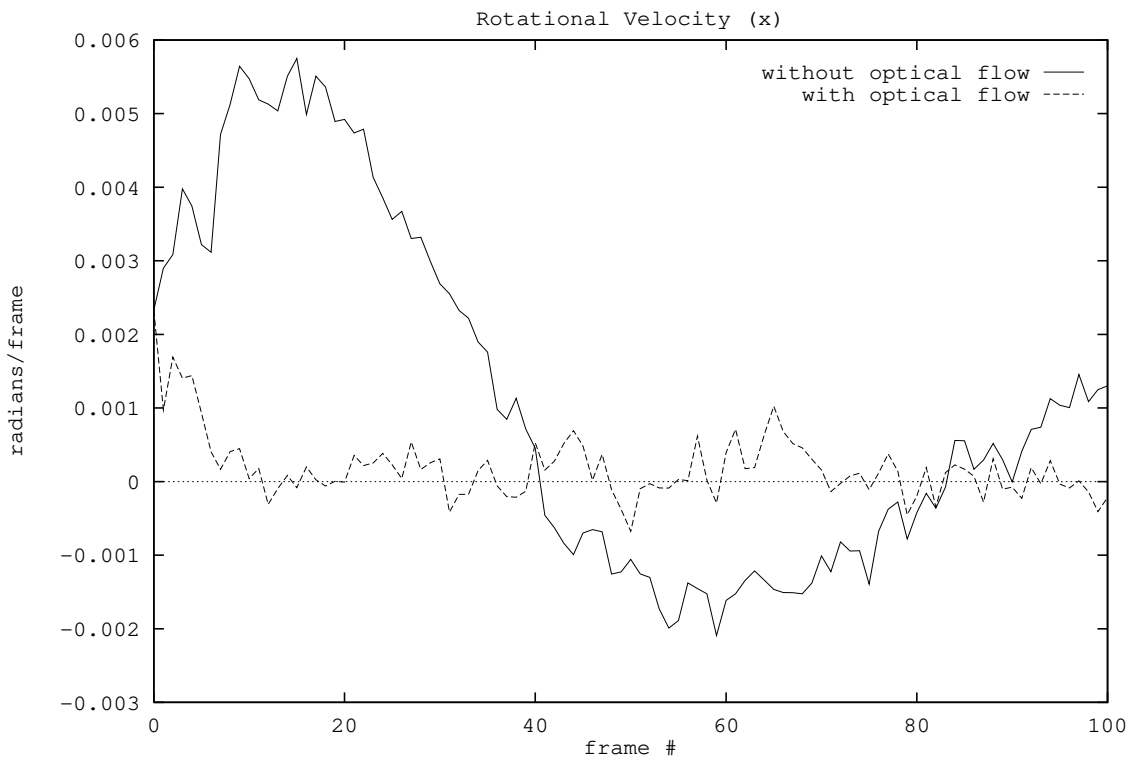


(d)

Figure 3.10: Experiment 3: Results of Monte Carlo simulations (bias)

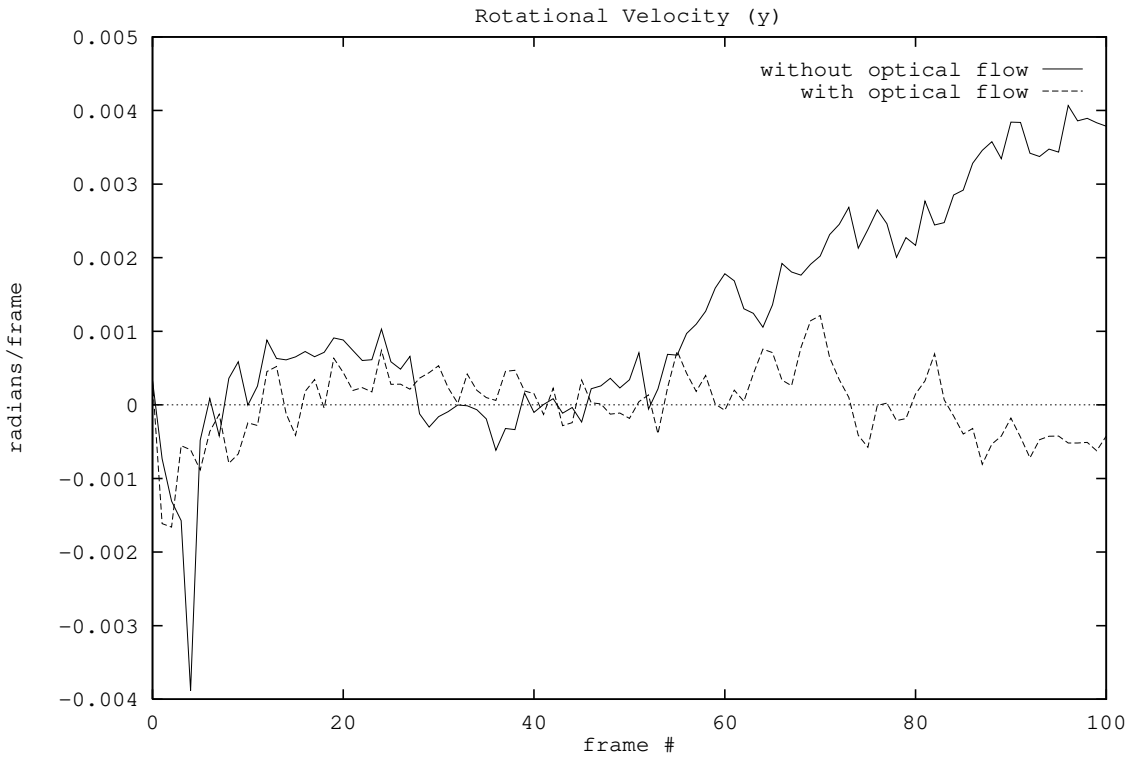


(e)

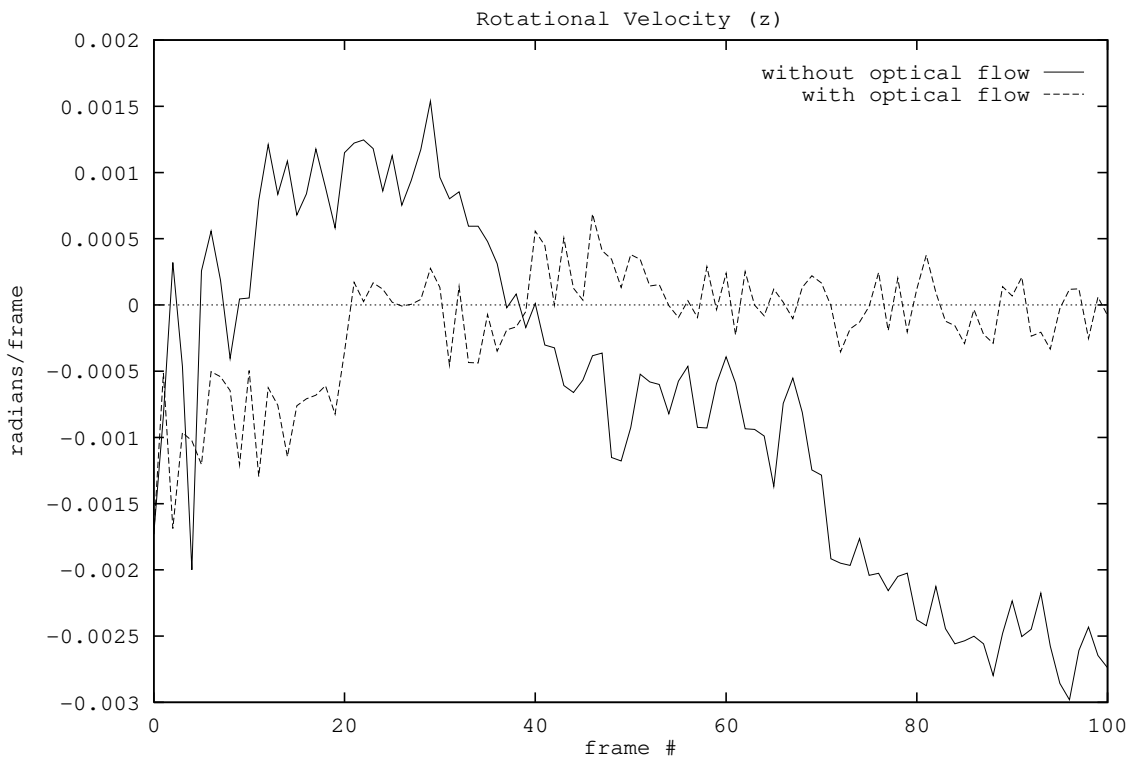


(f)

Figure 3.10: Experiment 3: Results of Monte Carlo simulations (bias)

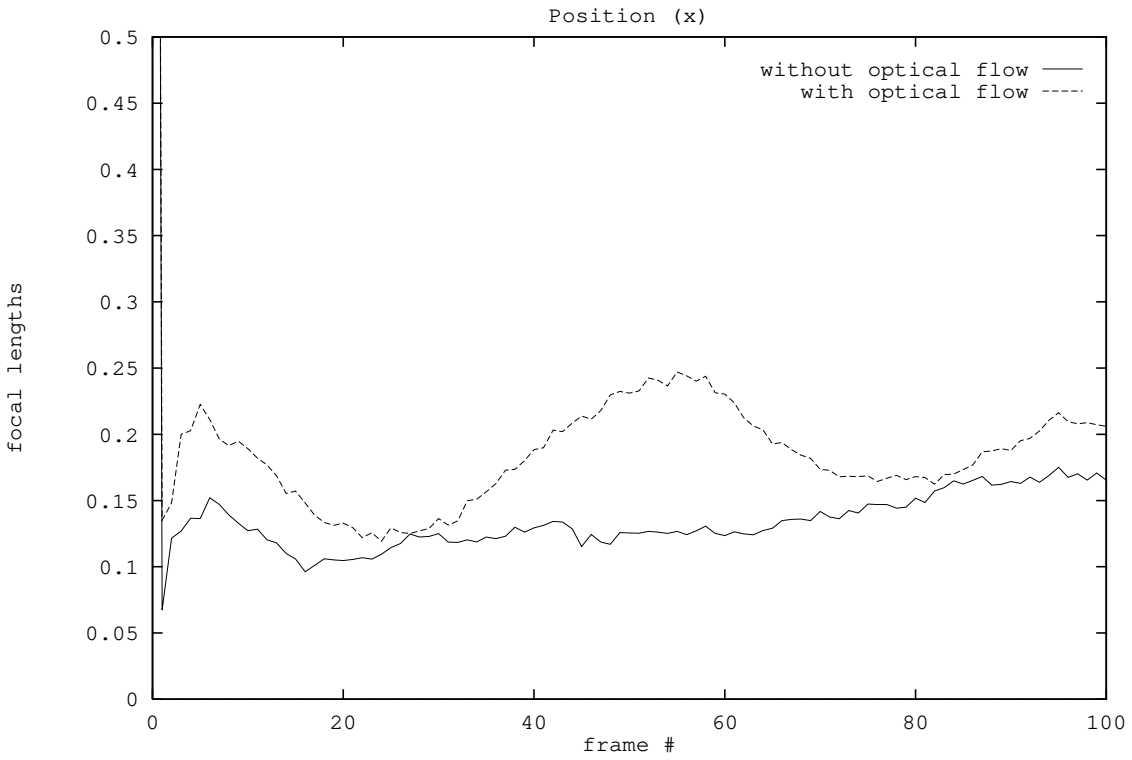


(g)

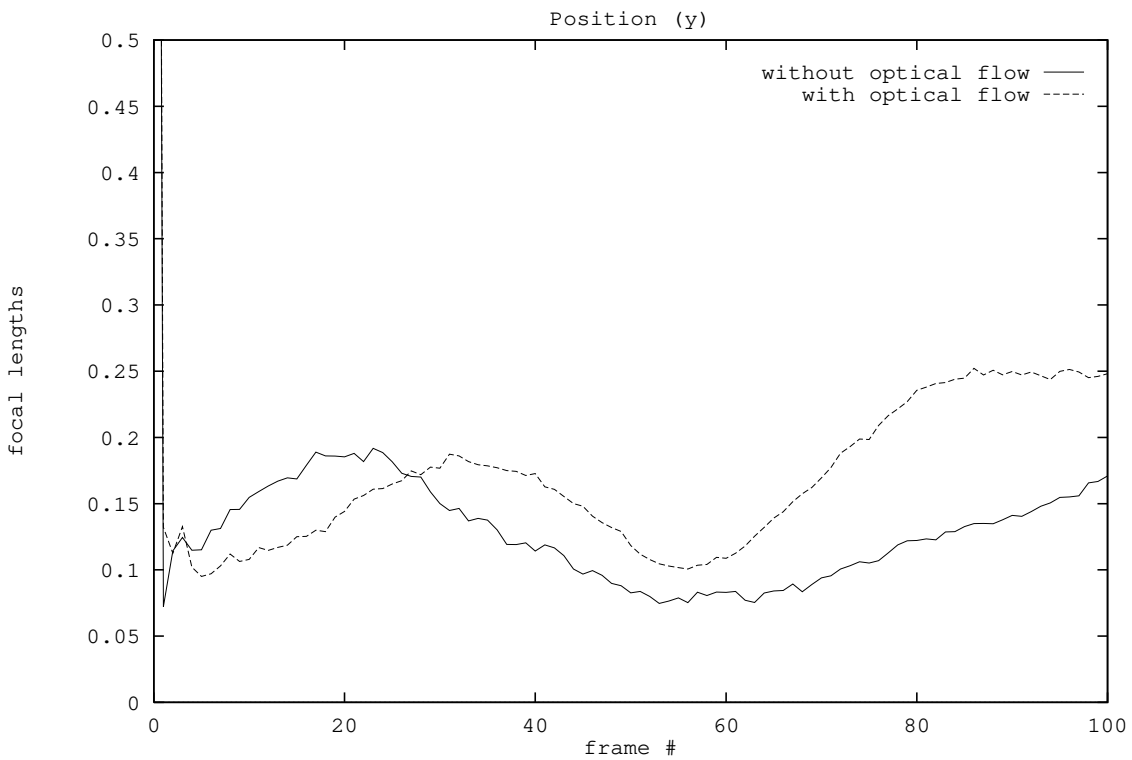


(h)

Figure 3.10: Experiment 3: Results of Monte Carlo simulations (bias)

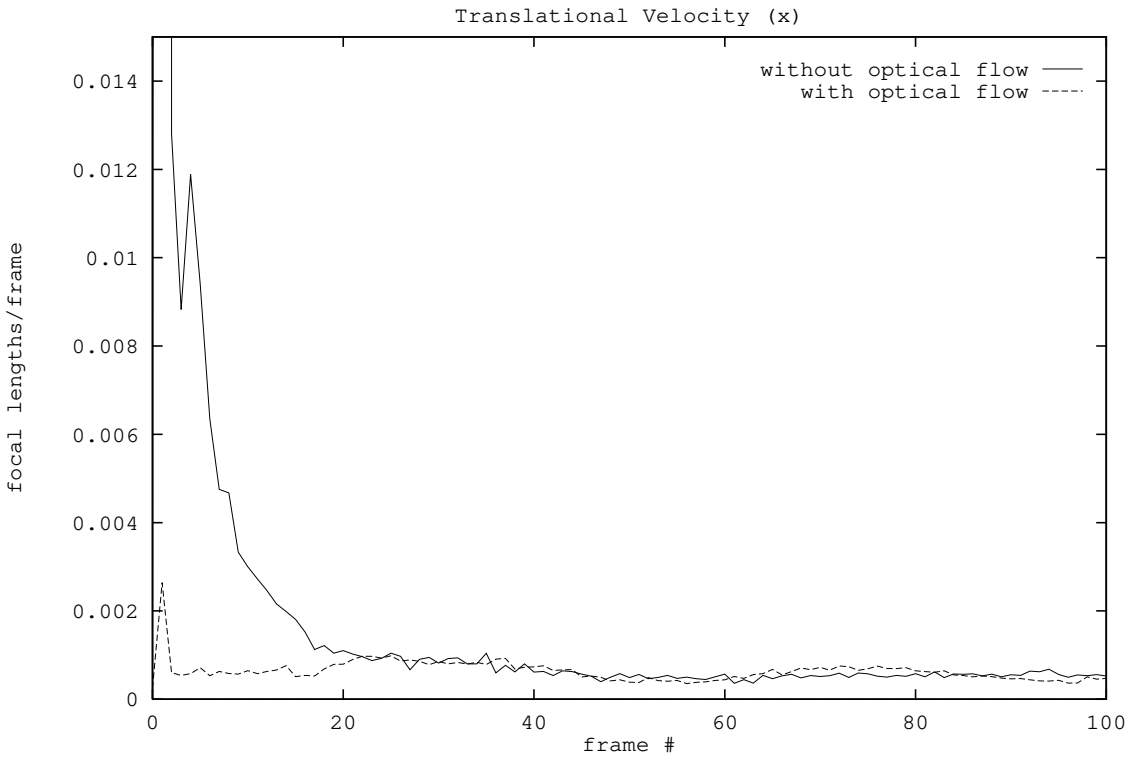


(a)

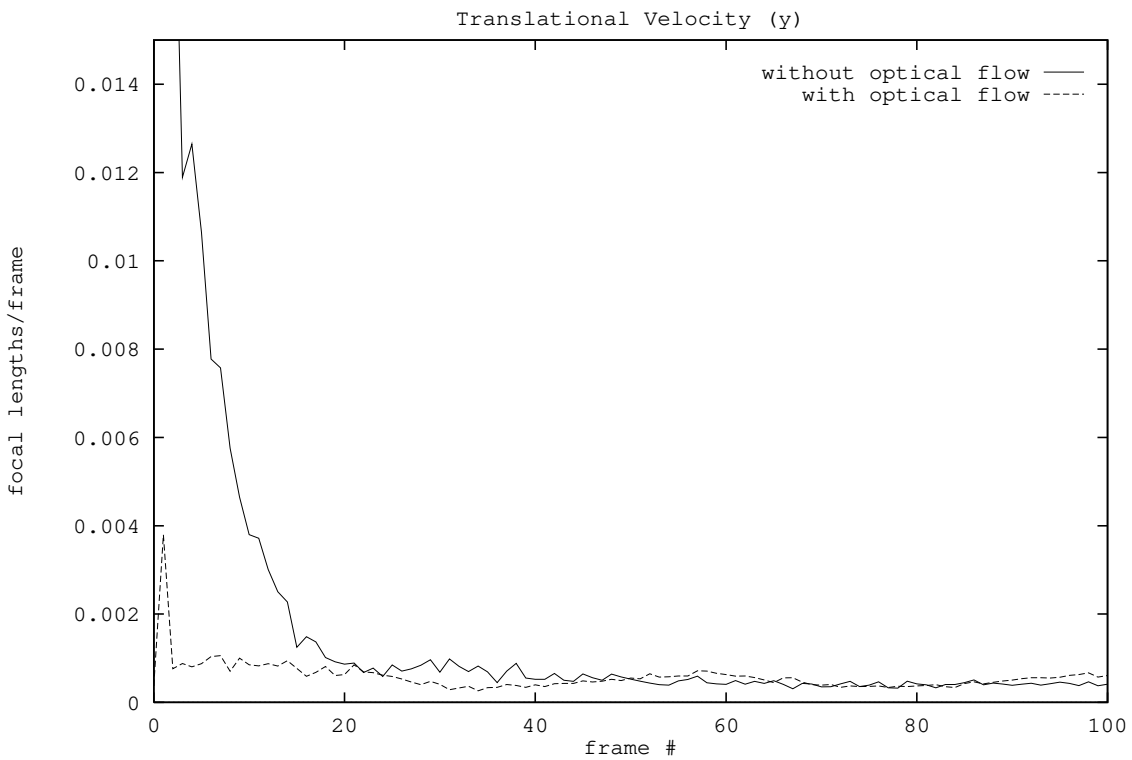


(b)

Figure 3.11: Experiment 3: Results of Monte Carlo simulations (variance)

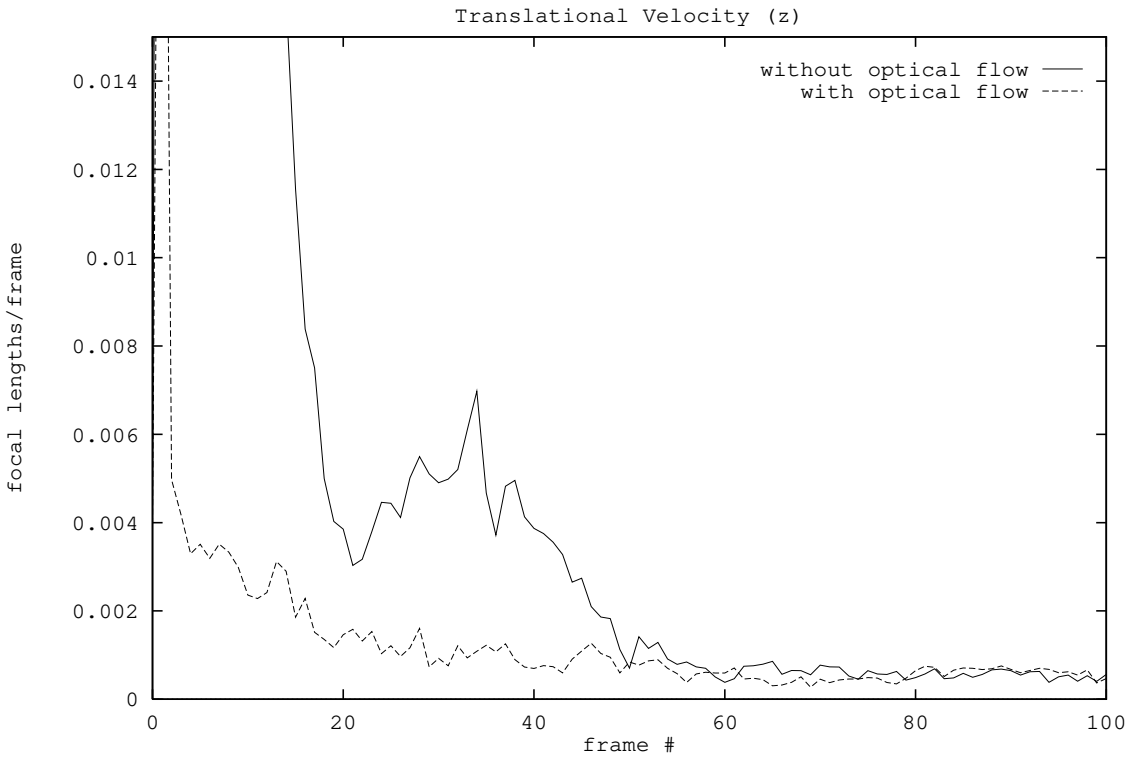


(c)

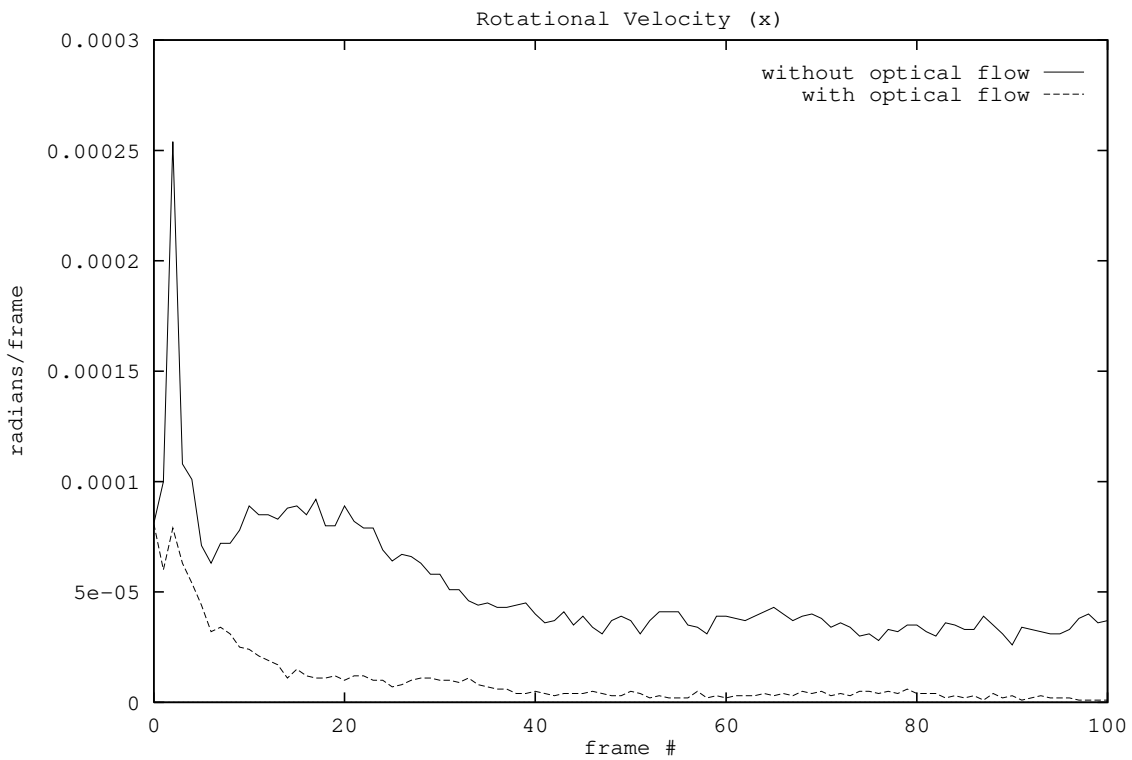


(d)

Figure 3.11: Experiment 3: Results of Monte Carlo simulations (variance)

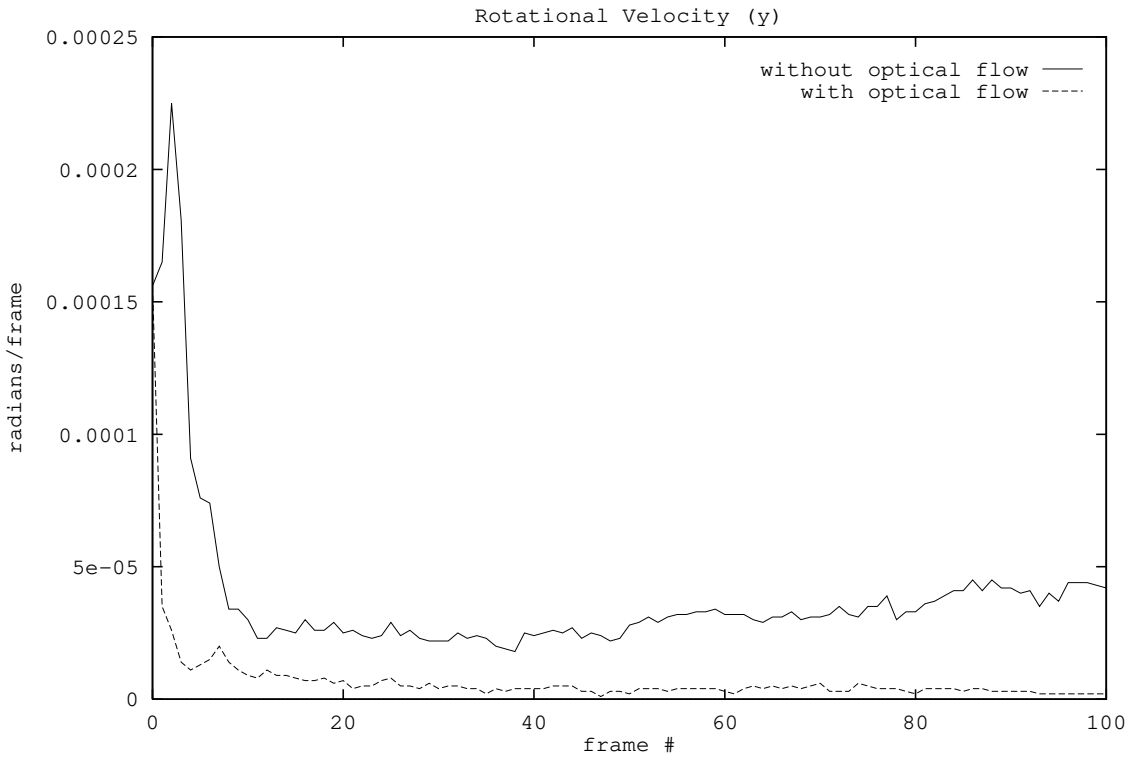


(e)

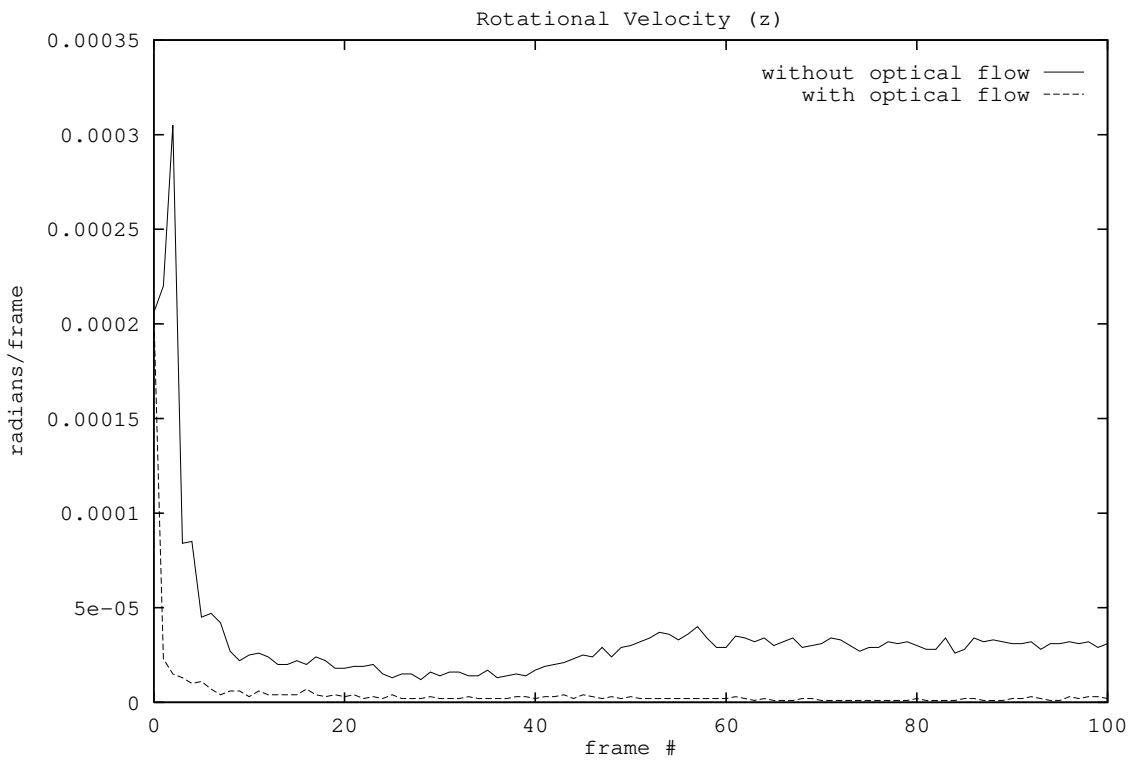


(f)

Figure 3.11: Experiment 3: Results of Monte Carlo simulations (variance)



(g)



(h)

Figure 3.11: Experiment 3: Results of Monte Carlo simulations (variance)

used before and after the abrupt change. The motion of the object, as seen in the image plane, is shown in Figure 3.12. This is not unlike the real-world situation in which a vehicle makes a sudden maneuver.

before

initial position $\mathbf{s}_R(t_0)$ (FL)	$[-5 \quad 5 \quad 20]^T$
translational velocity $\mathbf{T}$ (FL/frame)	$[0.05 \quad -0.1 \quad -0.2]^T$
rotational velocity ω (radians/frame)	$[0.03 \quad 0.04 \quad 0.05]^T$

after

translational velocity $\mathbf{T}$ (FL/frame)	$[0.11 \ -0.06 \ 0.1]^T$
rotational velocity ω (radians/frame)	$[-0.02 \ 0 \ 0.02]^T$

Table 3.3: Motion parameters for Experiment 4

The simulation results are shown in Figures 3.13 (bias) and 3.14 (variance). Again, in this experiment, the results presented are those obtained using the EKF. Because there is no iteration, the EKF tends to "react" more quickly to the abrupt change than the IEKF and ILFS. Here the emphasis is on the accuracy of the estimates made *after* the abrupt change. With the aid of optical flow measurements, the EKF reacted more quickly to the abrupt change (with the exception of state 4, the y-component of the translational velocity). The results from this experiment are consistent with those from Experiments 2 and 3.

3.4.6 Experiment 5: Feature Displacements

This final experiment investigates the feasibility of obtaining optical flow measurements using the feature displacement method described in Section 3.2.2. The motion parameters used in this experiment are the same as those used in Experiment 1. The number of image frames, not including the initial frame, is still 100 but only estimates for 99 frames are available due to the one-frame delay inherent in the forward difference approximation. The average absolute error in position measurements is 1.6

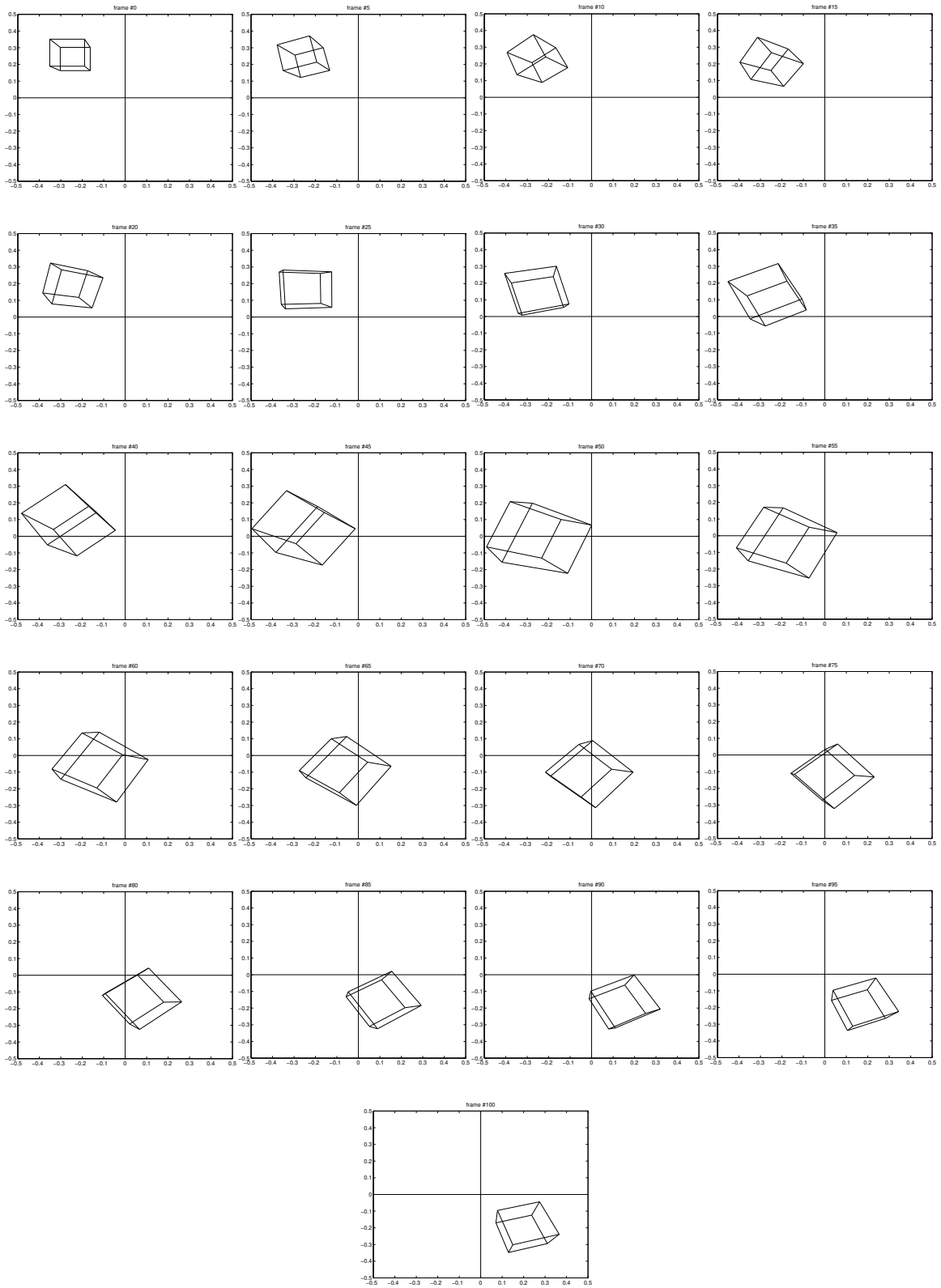
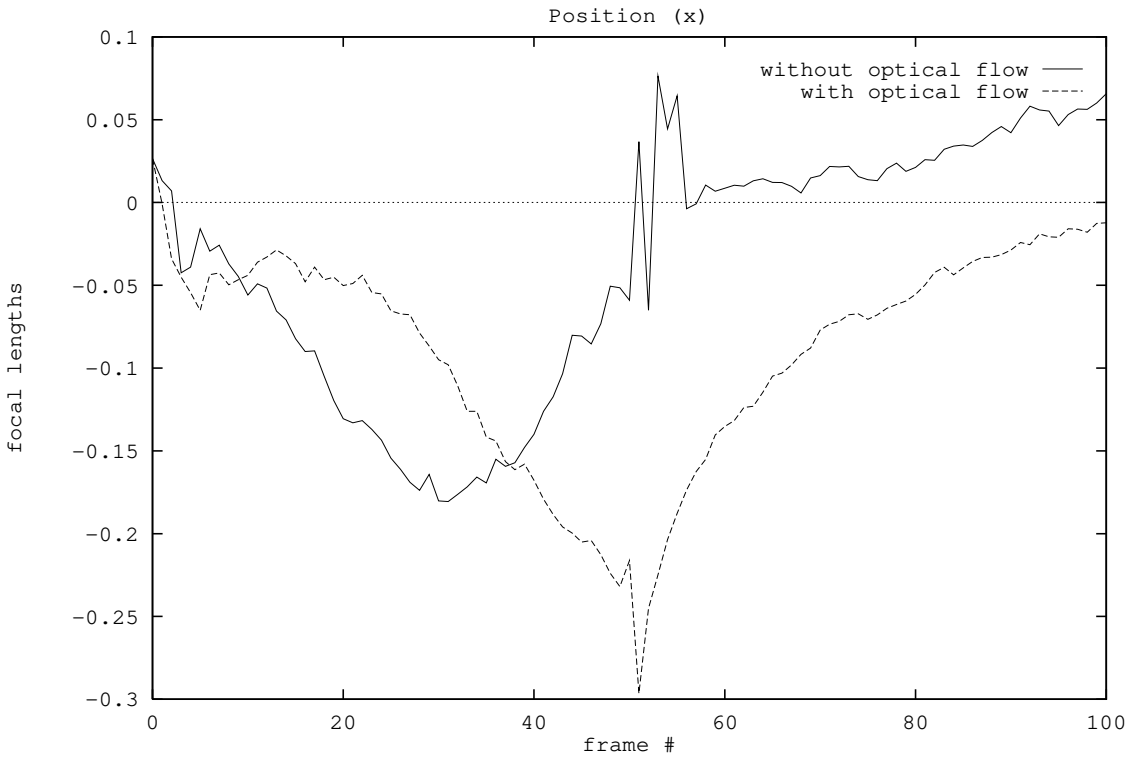
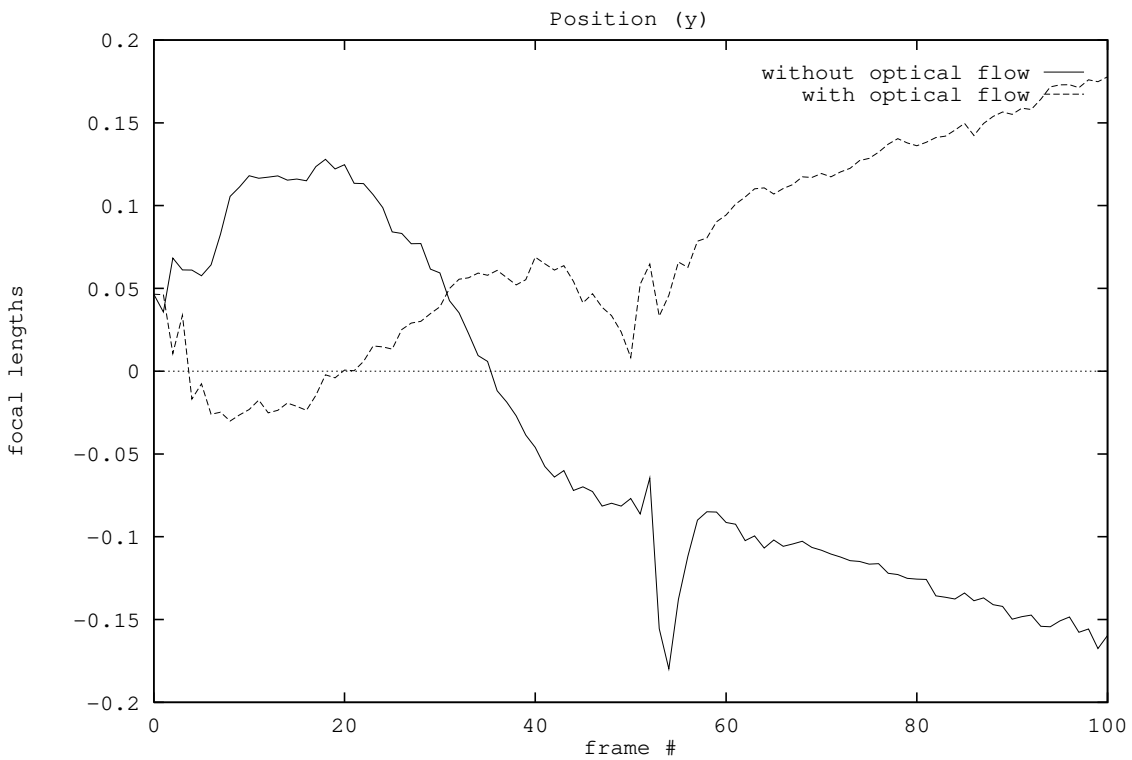


Figure 3.12: Experiment 4: Object motion in image plane

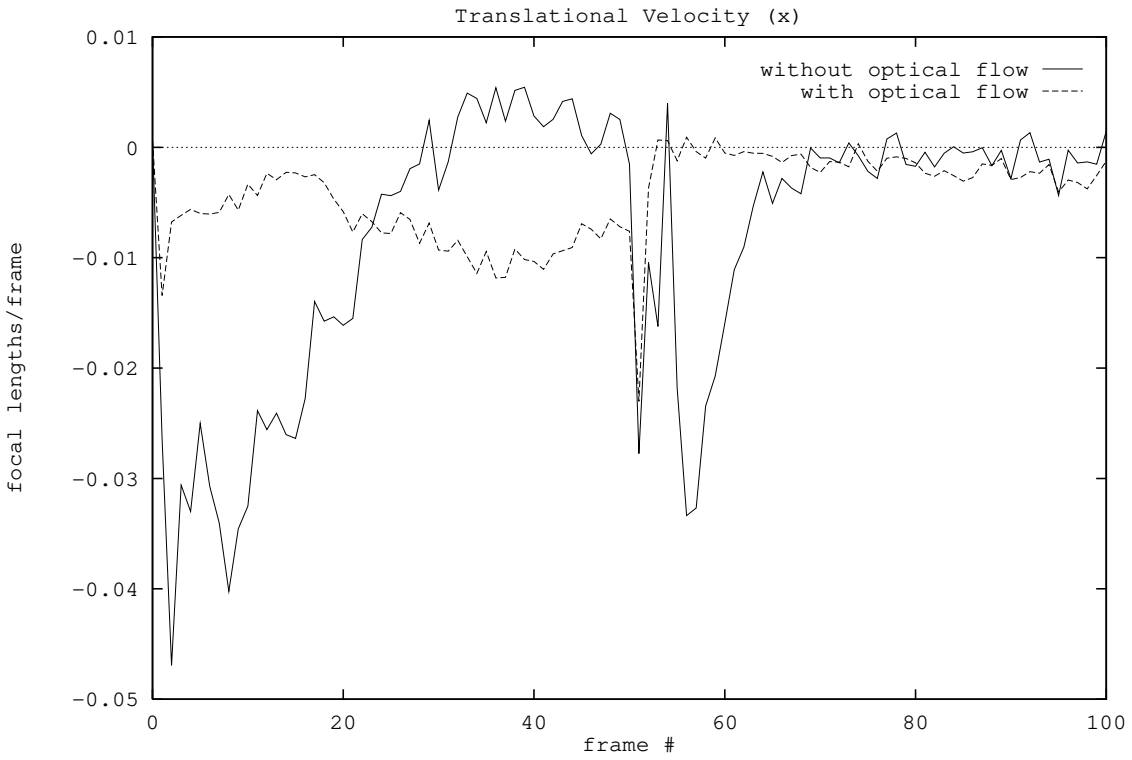


(a)

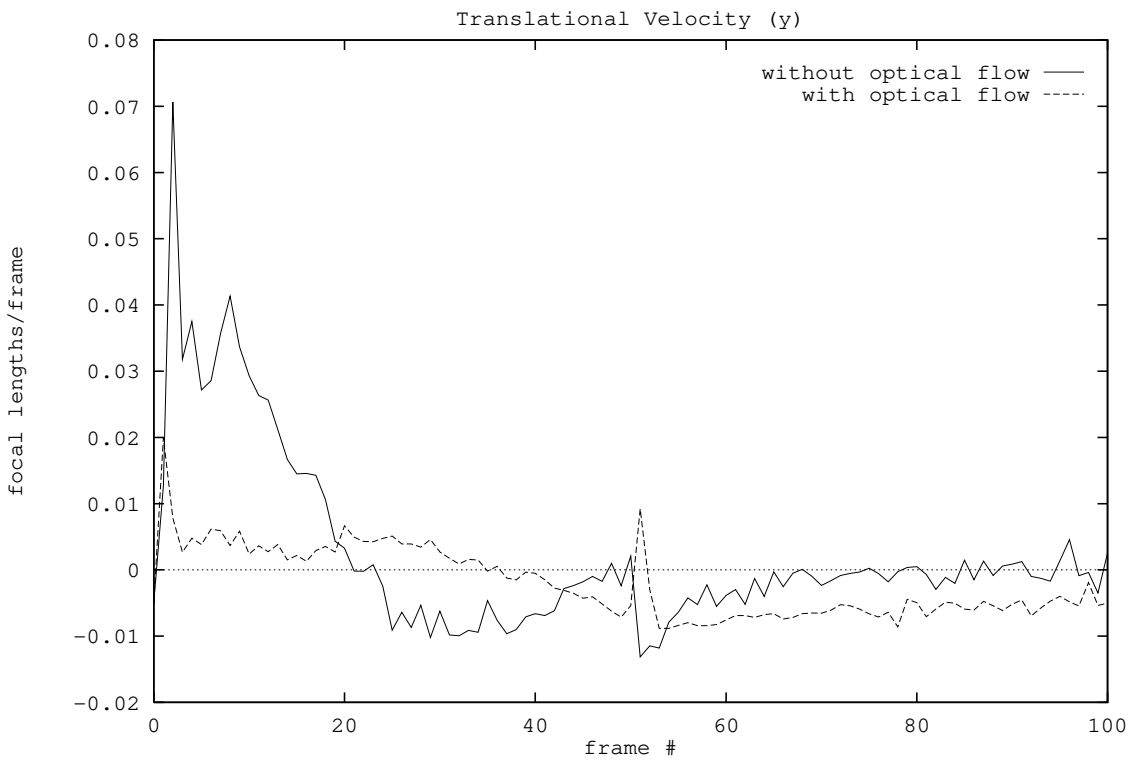


(b)

Figure 3.13: Experiment 4: Results of Monte Carlo simulations (bias)

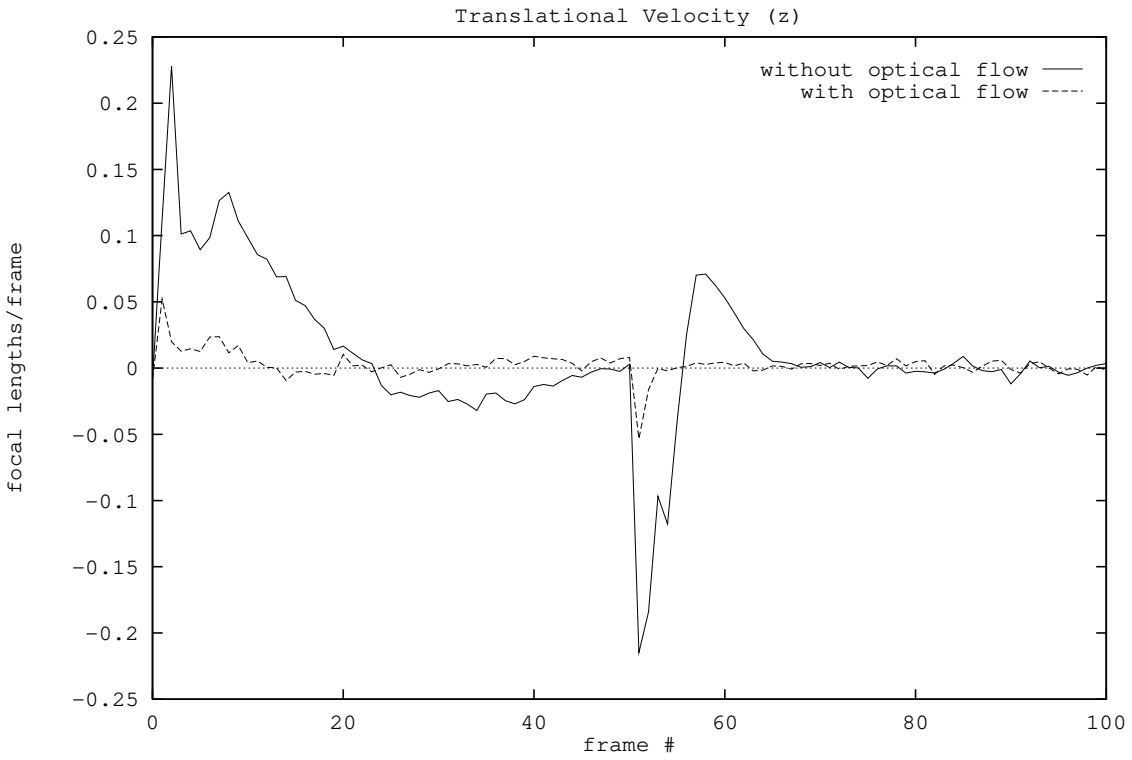


(c)

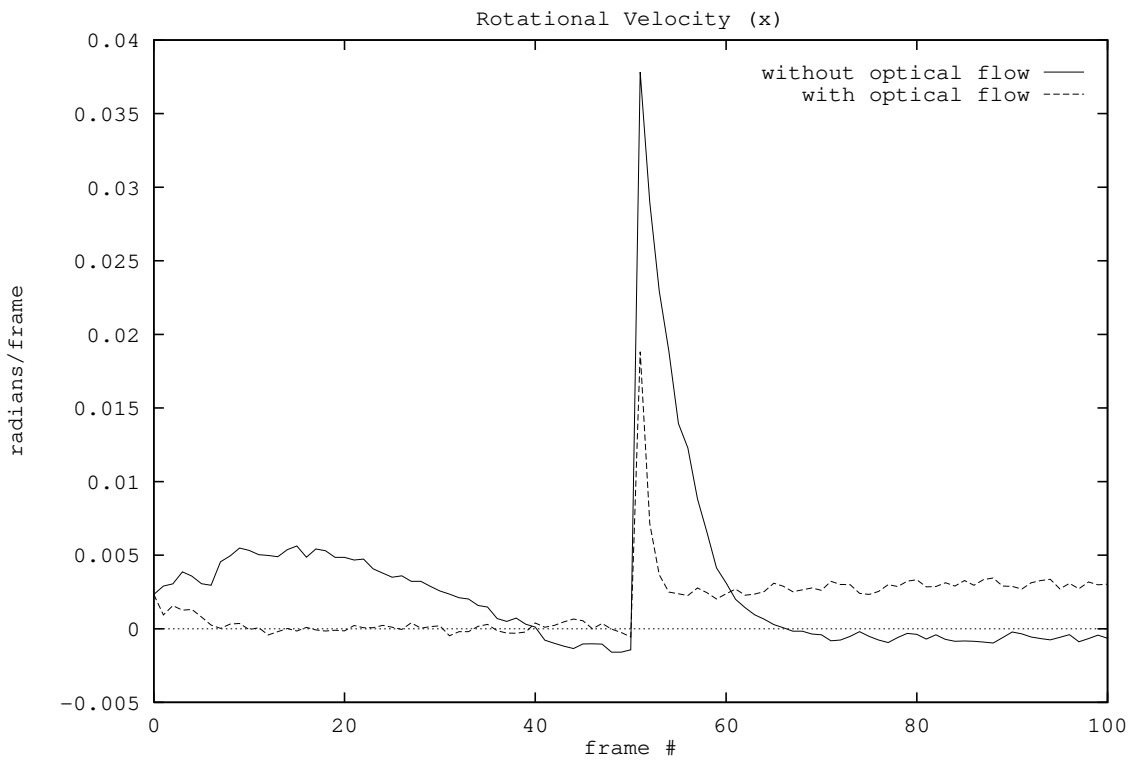


(d)

Figure 3.13: Experiment 4: Results of Monte Carlo simulations (bias)

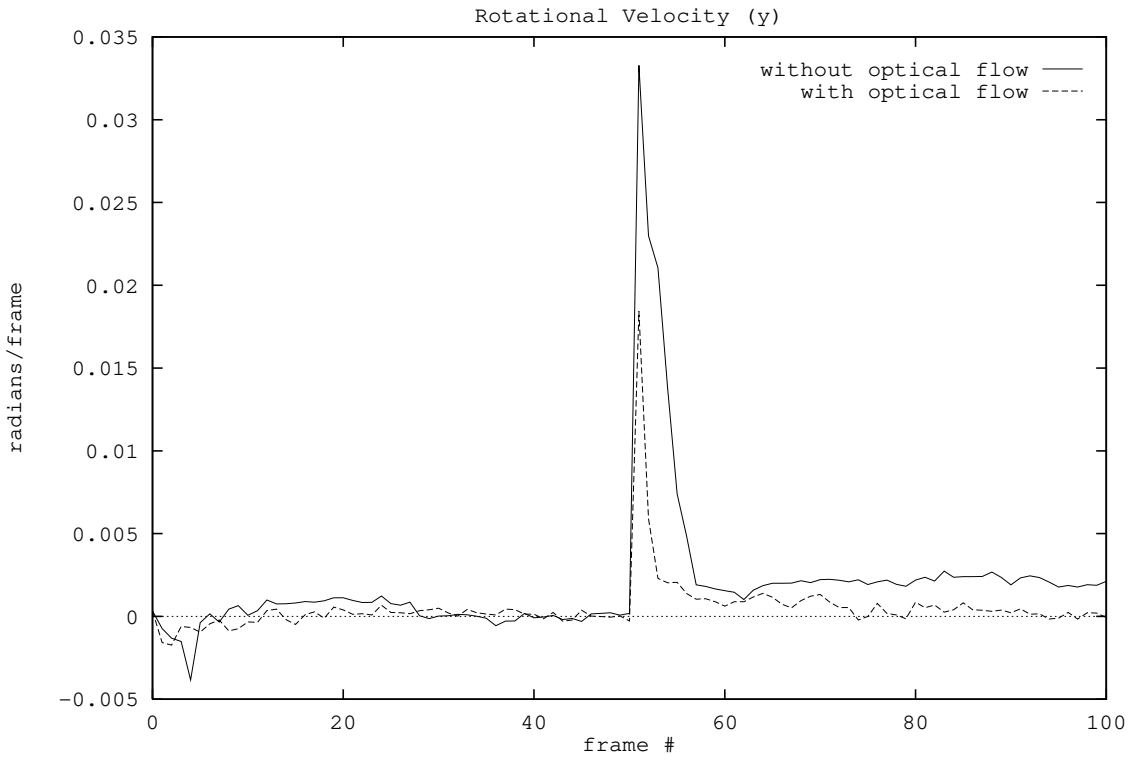


(e)

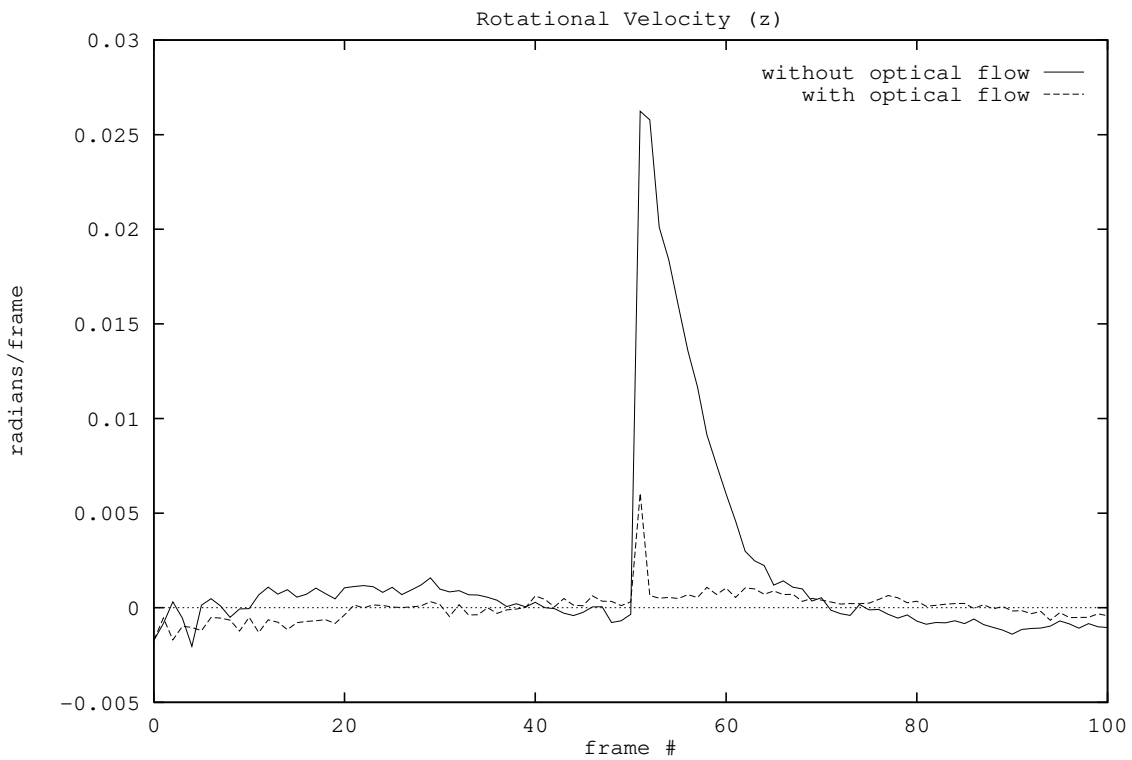


(f)

Figure 3.13: Experiment 4: Results of Monte Carlo simulations (bias)

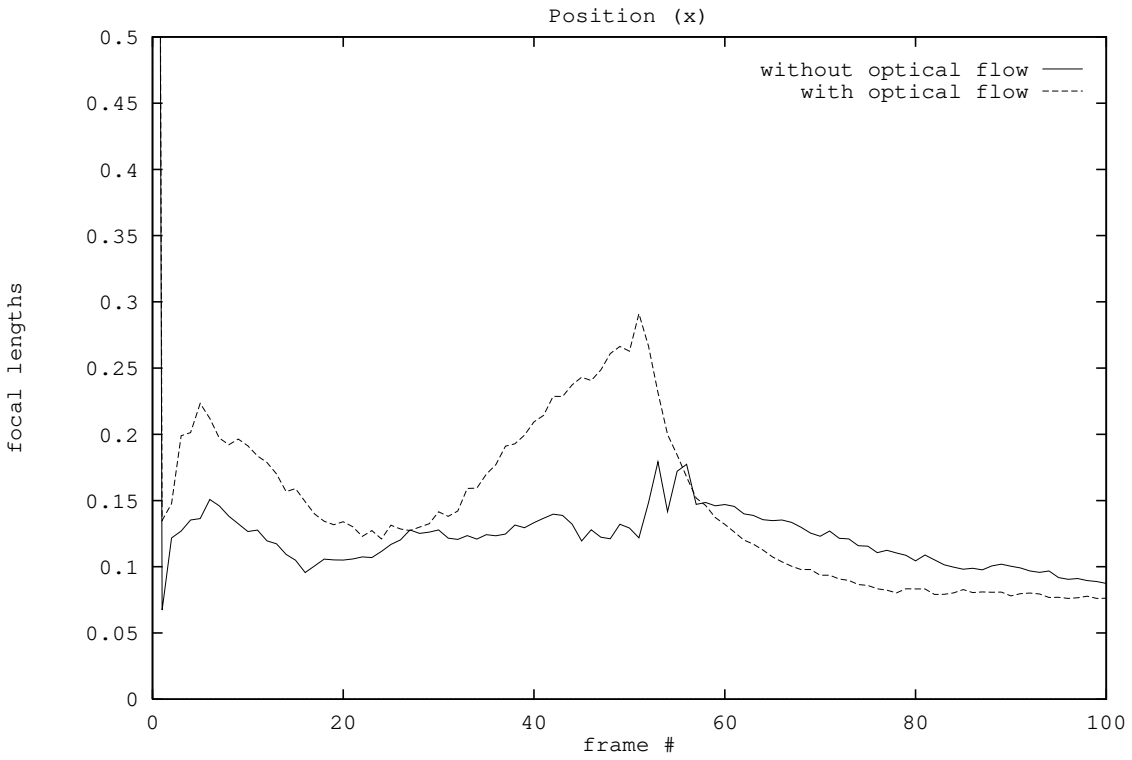


(g)

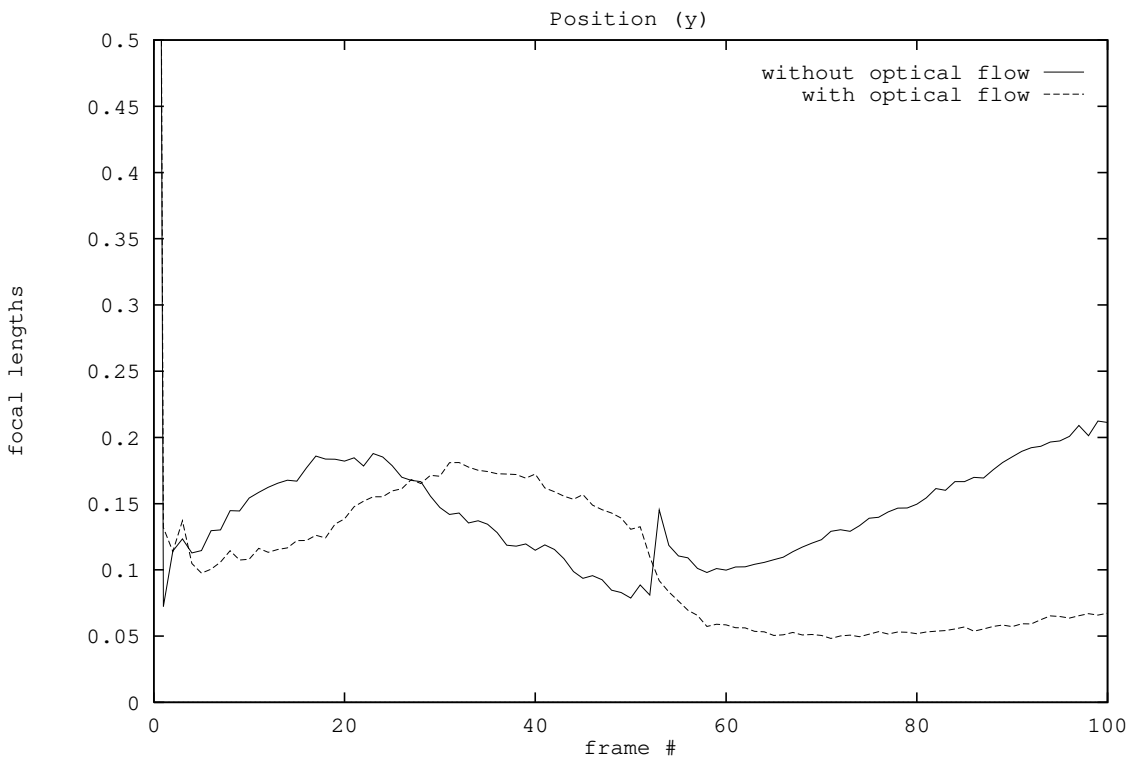


(h)

Figure 3.13: Experiment 4: Results of Monte Carlo simulations (bias)

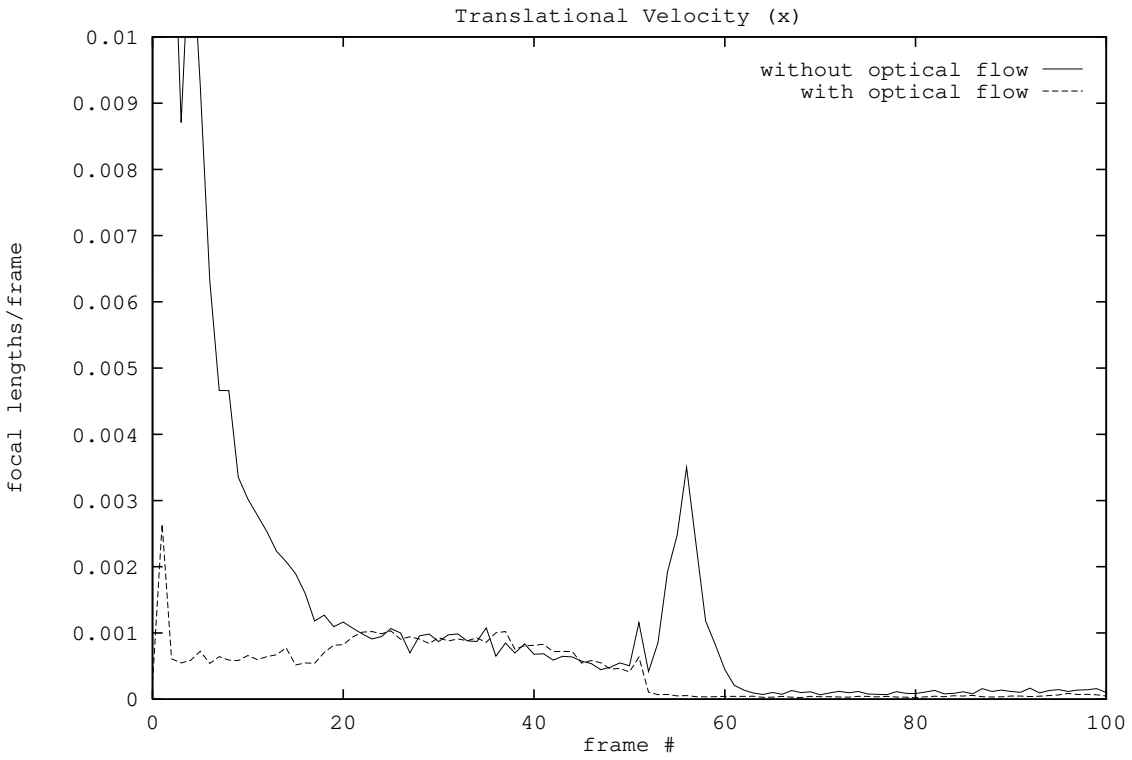


(a)

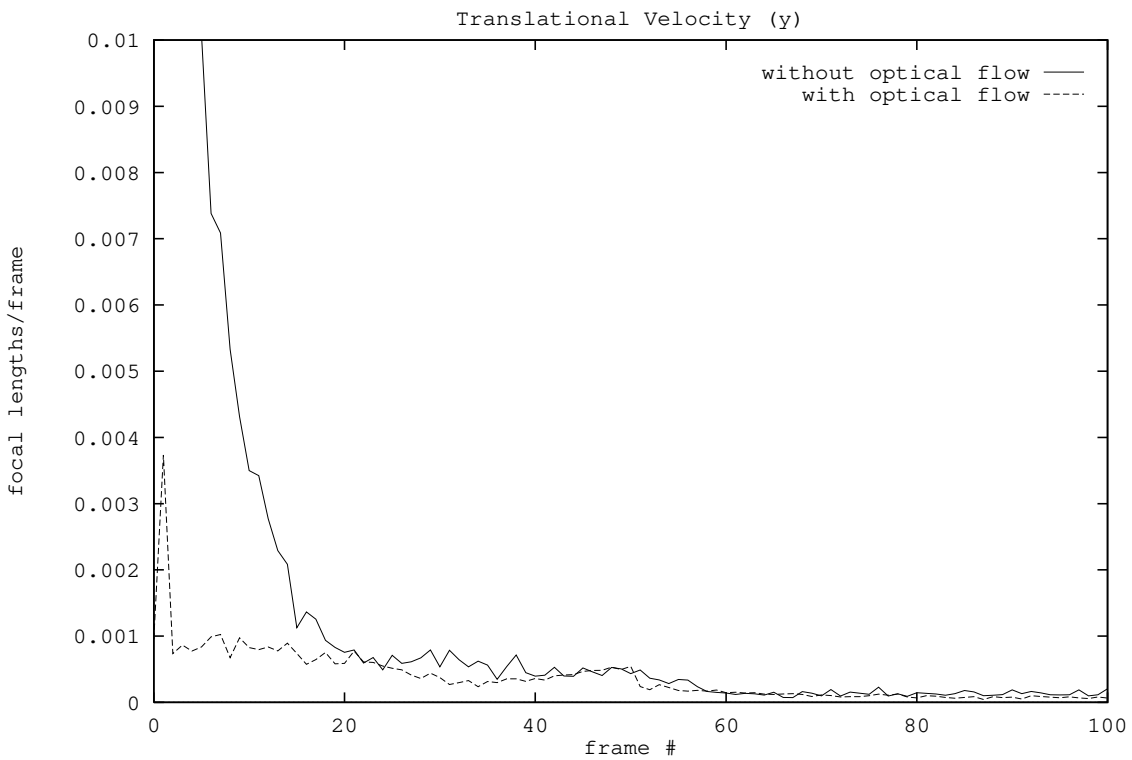


(b)

Figure 3.14: Experiment 4: Results of Monte Carlo simulations (variance)

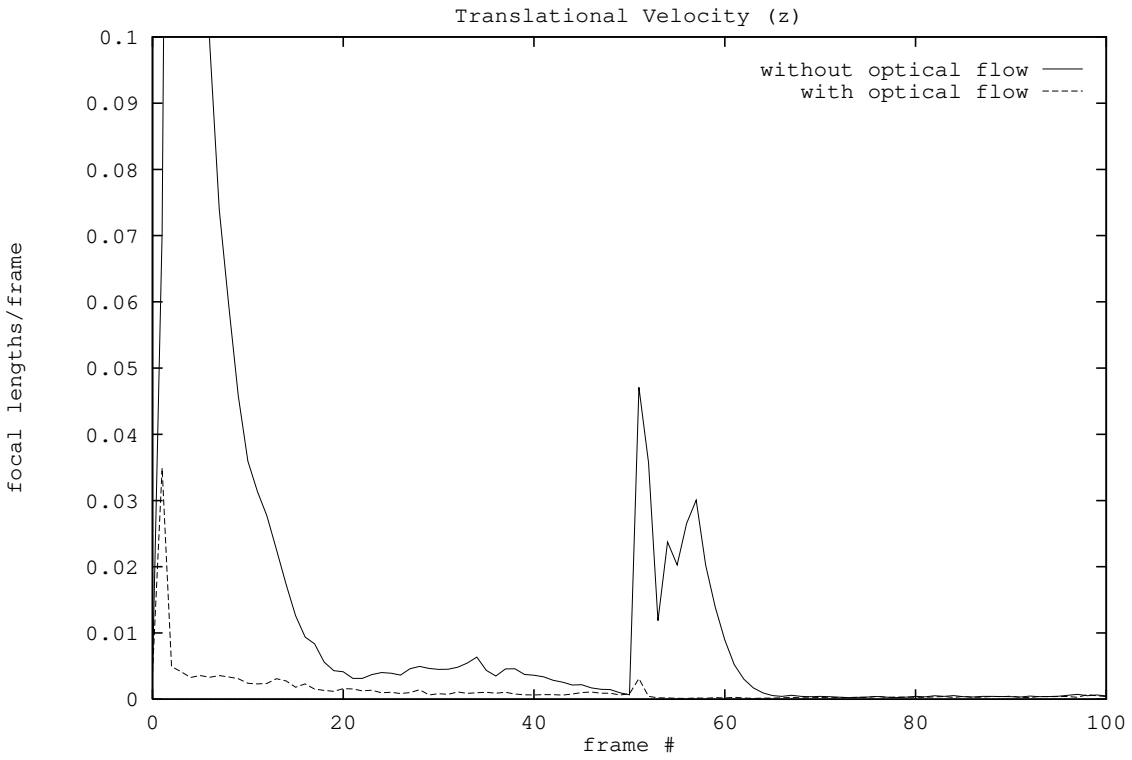


(c)

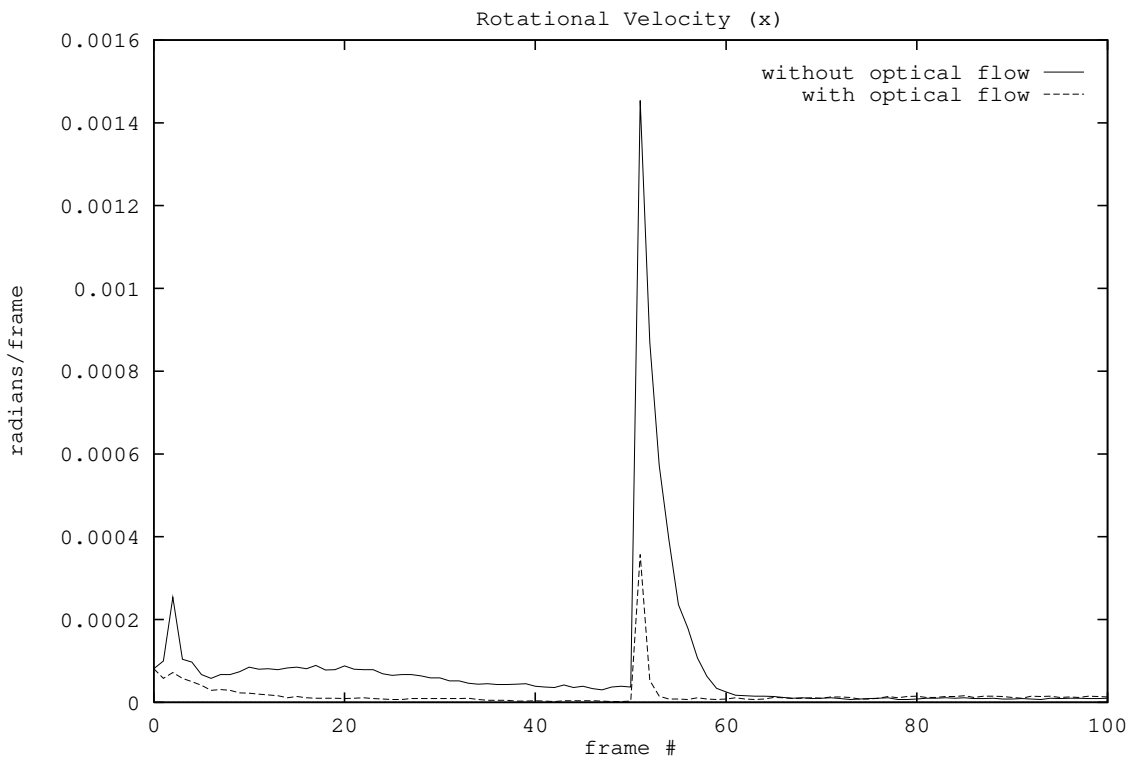


(d)

Figure 3.14: Experiment 4: Results of Monte Carlo simulations (variance)

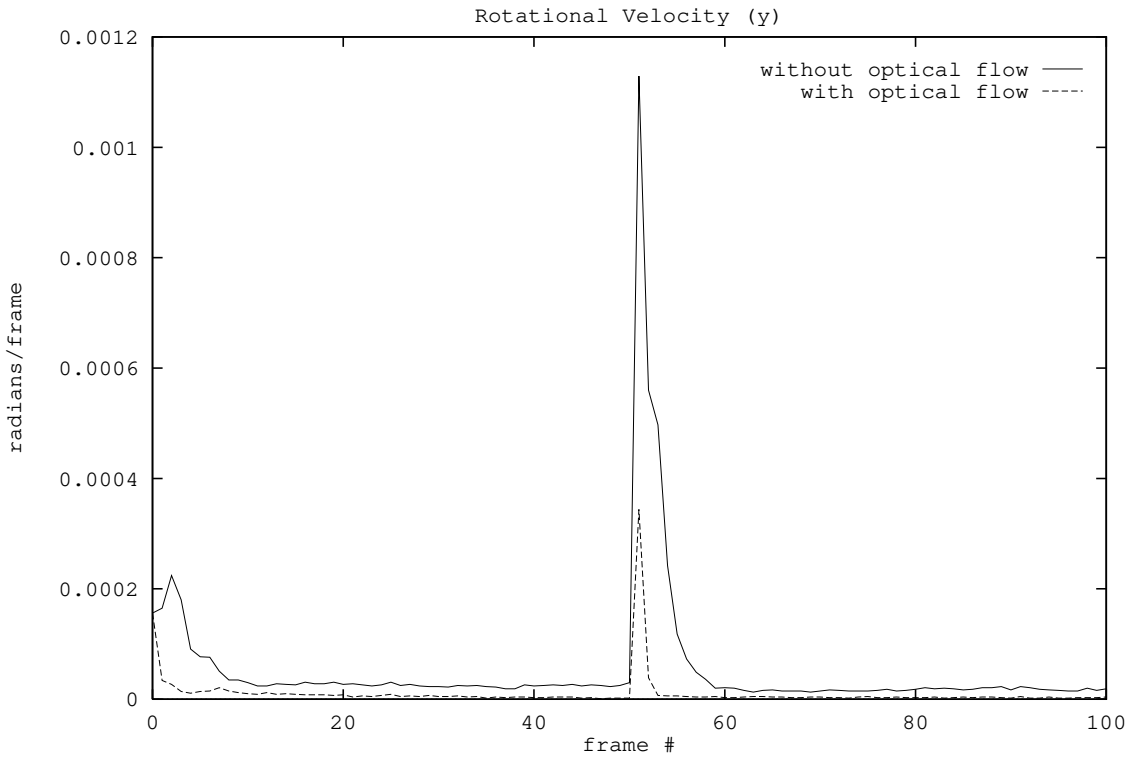


(e)

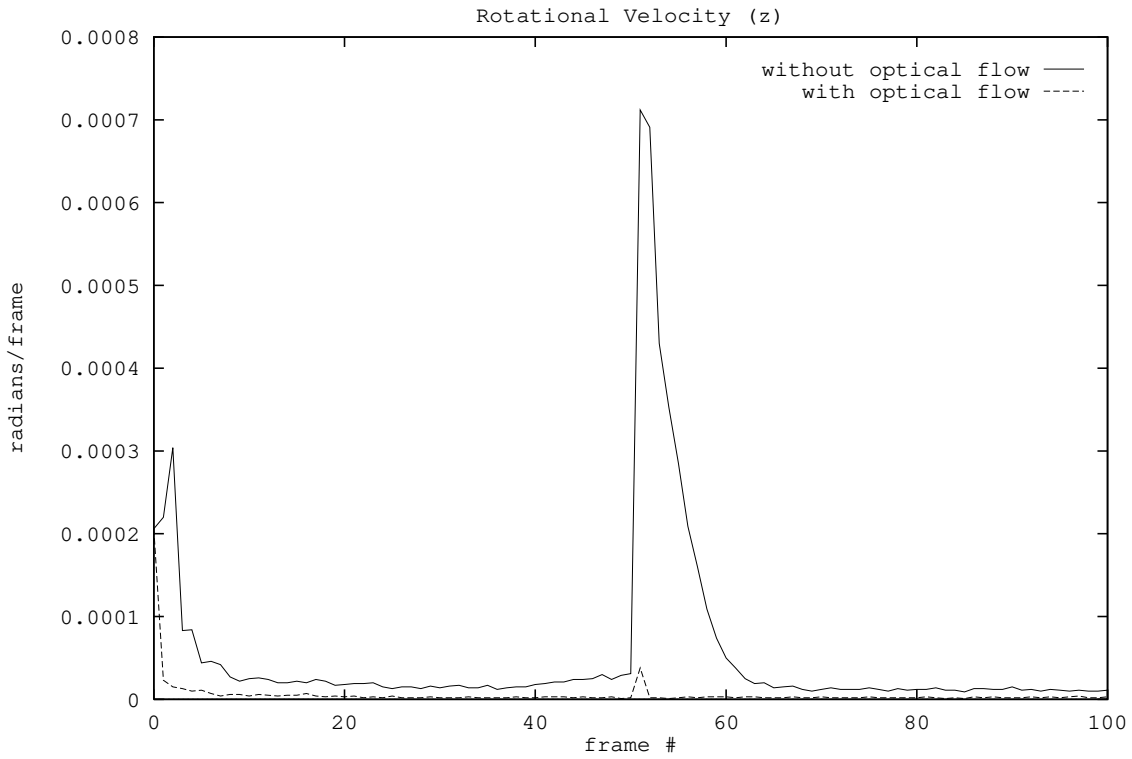


(f)

Figure 3.14: Experiment 4: Results of Monte Carlo simulations (variance)



(g)



(h)

Figure 3.14: Experiment 4: Results of Monte Carlo simulations (variance)

pixels and the average relative error in optical flow measurements is 90.74%!

The simulation results in Figures 3.15 and 3.16 indicate that the FDM version of the hybrid algorithm offers only marginal improvement over the BCC algorithm in the translational velocity states while still maintaining considerably better performance in the rotational velocity states. However, this should not be surprising. First of all, one must realize that the error in optical flow measurements is about 90%. Secondly, consider the noisy position measurements of a feature point during frames k and $k+1$:

$$\mathbf{p}_k(\epsilon) = (x_k(\epsilon), y_k(\epsilon)) = (x_k + \delta_{x_k}, y_k + \delta_{y_k}) \tag{3.46}$$

$$\mathbf{p}_{k+1}(\epsilon) = (x_{k+1}(\epsilon), y_{k+1}(\epsilon)) = (x_{k+1} + \delta_{x_{k+1}}, y_{k+1} + \delta_{y_{k+1}}) \tag{3.47}$$

where the errors $(x_k(\epsilon), y_k(\epsilon), x_{k+1}(\epsilon), y_{k+1}(\epsilon))$ are additive, independent, identically distributed, zero-mean Gaussian variables with variance σ_n^2. The errors in the components of the optical flow vector are:

$$u_k(\epsilon) = \frac{\delta_{x_{k+1}} - \delta_{x_k}}{\Delta t} \tag{3.48}$$

$$v_k(\epsilon) = \frac{\delta_{y_{k+1}} - \delta_{y_k}}{\Delta t}. \tag{3.49}$$

It can be shown that

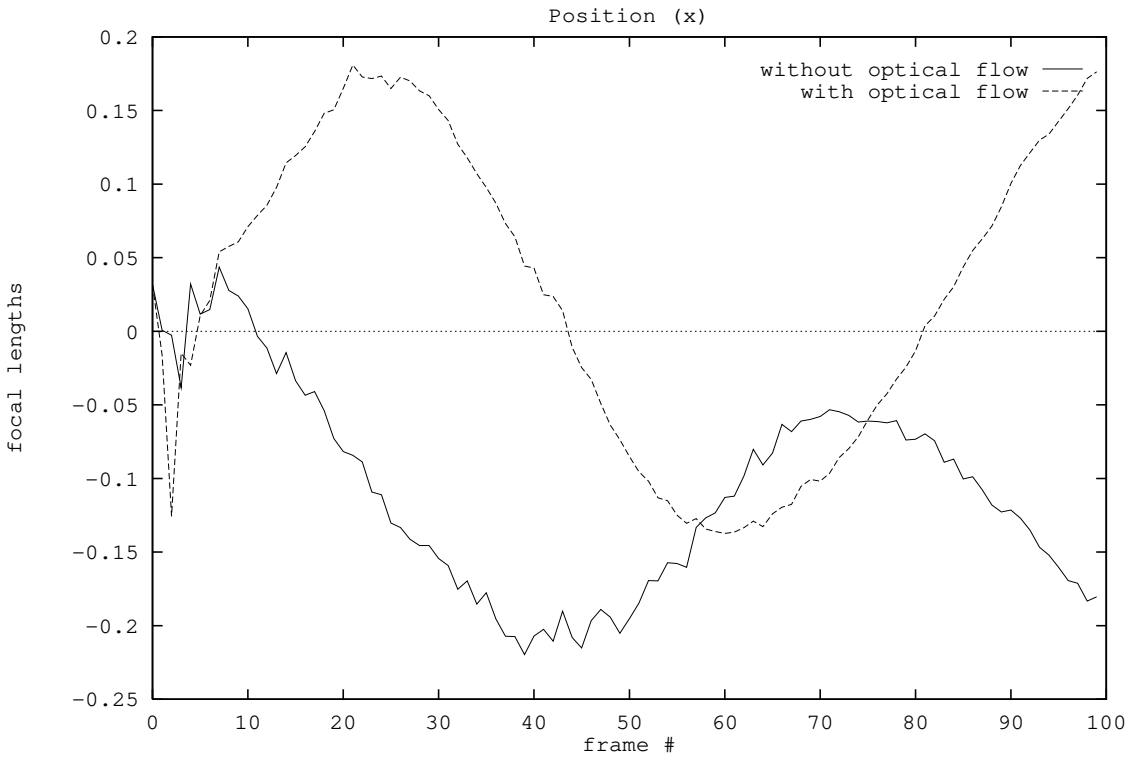
$$E\{\delta_{u_k}\} = E\{\delta_{v_k}\} = 0 \tag{3.50}$$

and

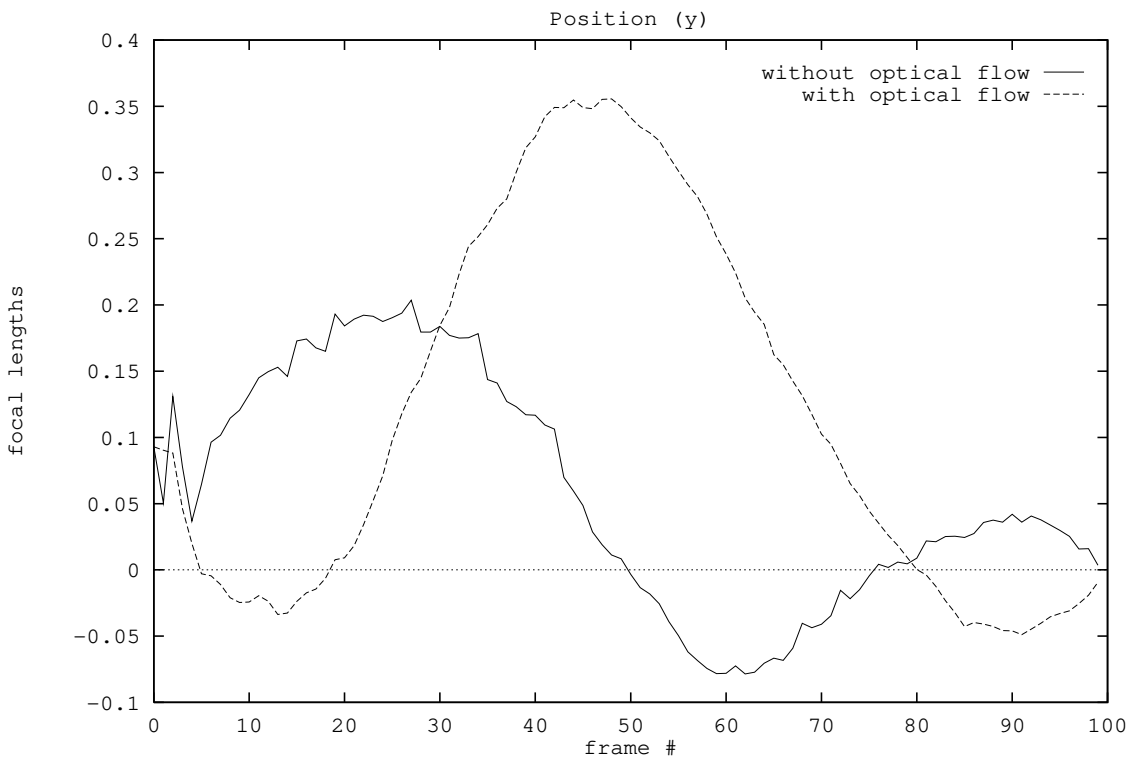
$$E\{\delta_{u_k}^2\} = E\{\delta_{v_k}^2\} = \frac{2\sigma_n^2}{\Delta t}. \tag{3.51}$$

One should be aware of the hidden assumption that (3.35) and (3.36) would yield the true optical flow measurements under noise-free conditions. This assumption, however, is only valid in certain situations. To see this, let us treat the Cartesian coordinates of a brightness pattern as a function of time and apply Taylor's theorem [96]:

$$x(t + \Delta t) = x(t) + \Delta t \dot{x}(t) + \int \frac{1}{2} \Delta t^2 \ddot{x}(t) dt. \tag{3.52}$$

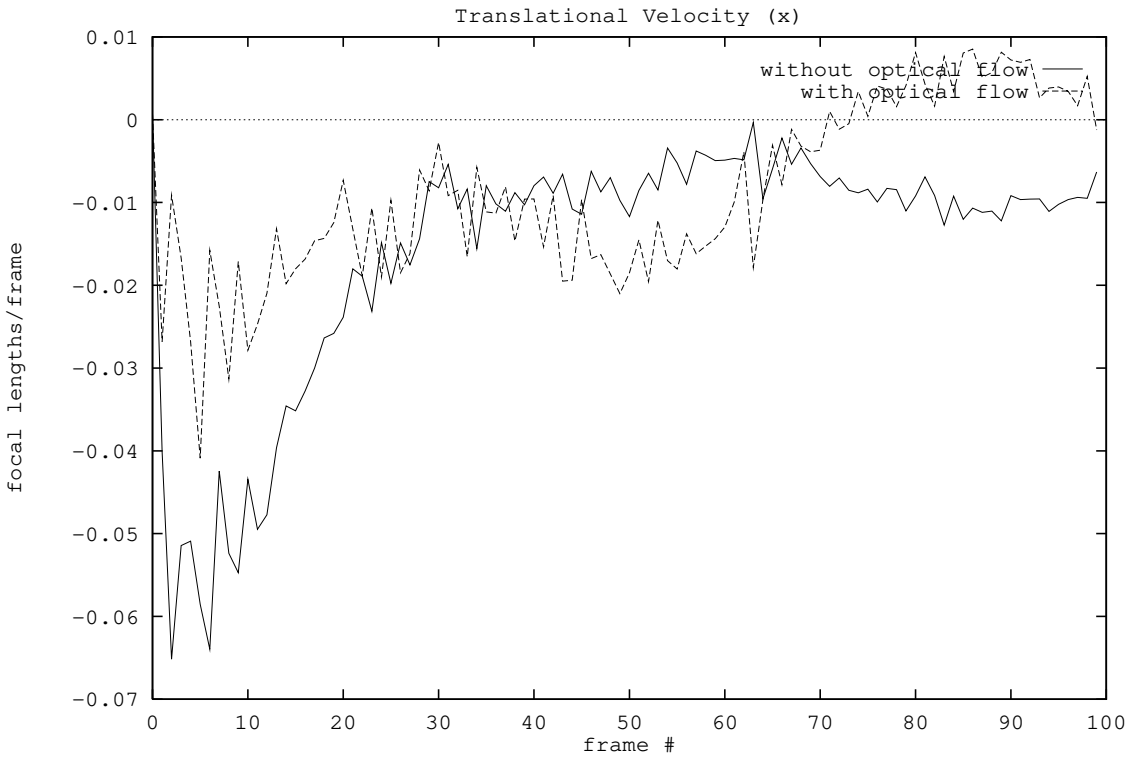


(a)

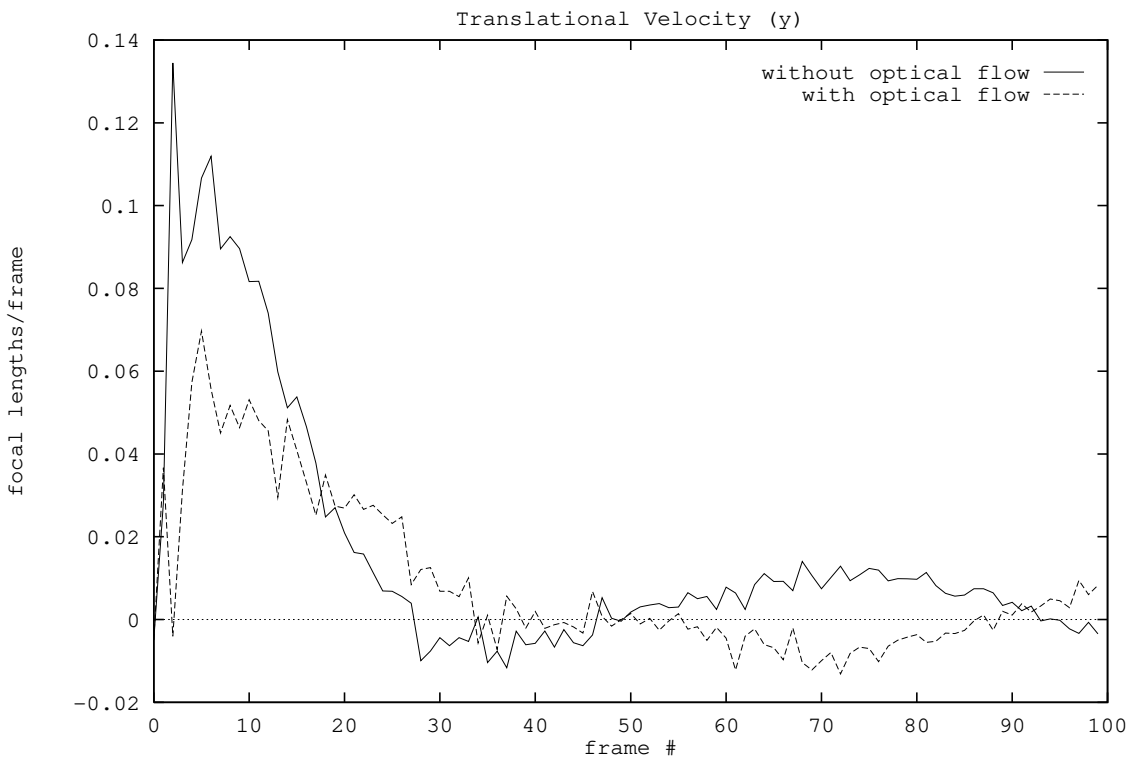


(b)

Figure 3.15: Experiment 5: FDM with correlated errors (bias)

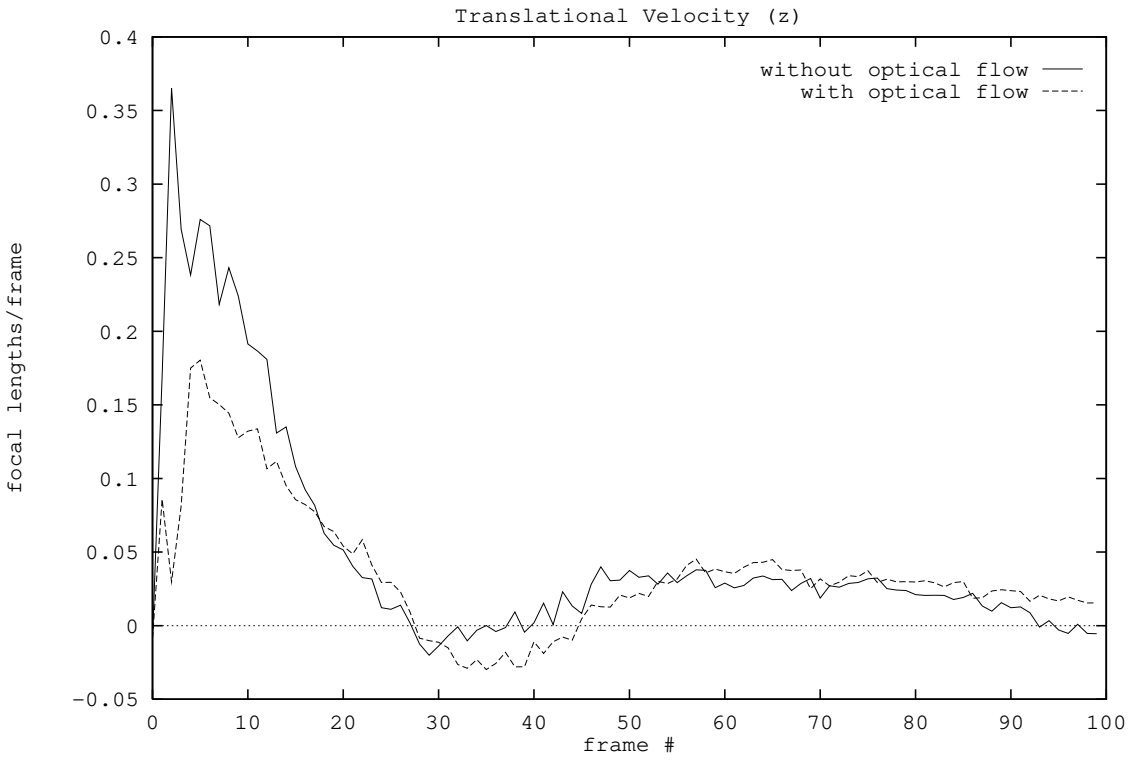


(c)

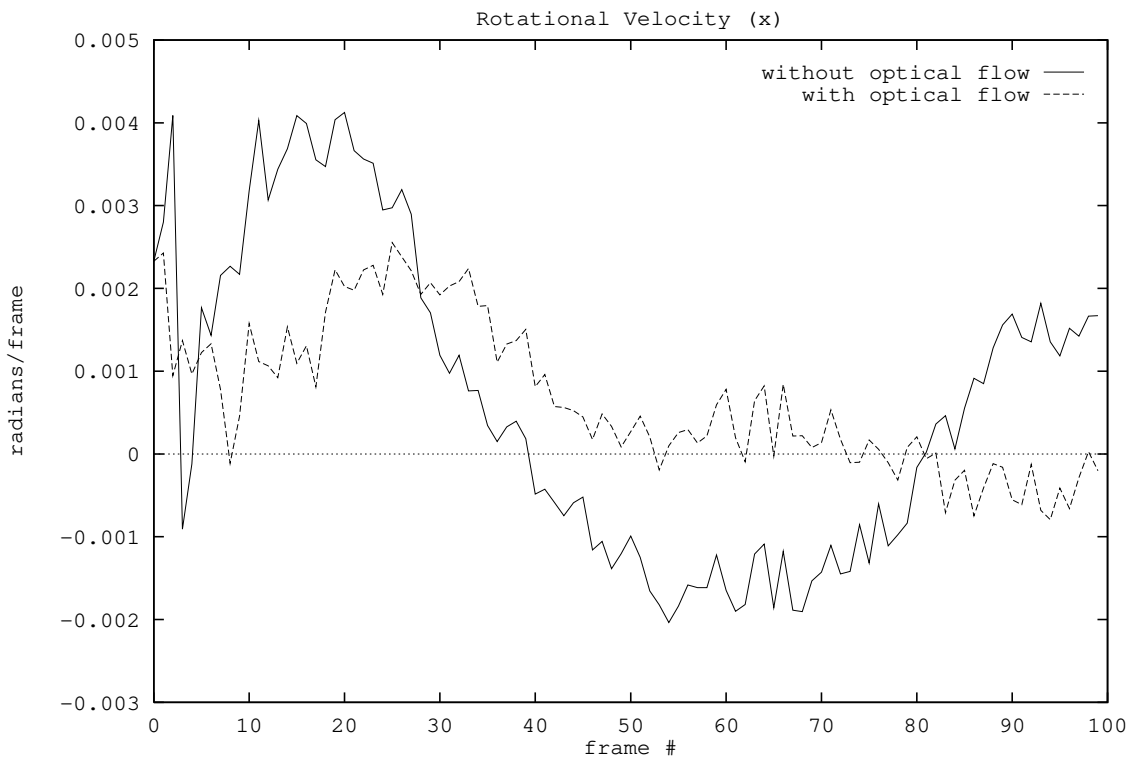


(d)

Figure 3.15: Experiment 5: FDM with correlated errors (bias)

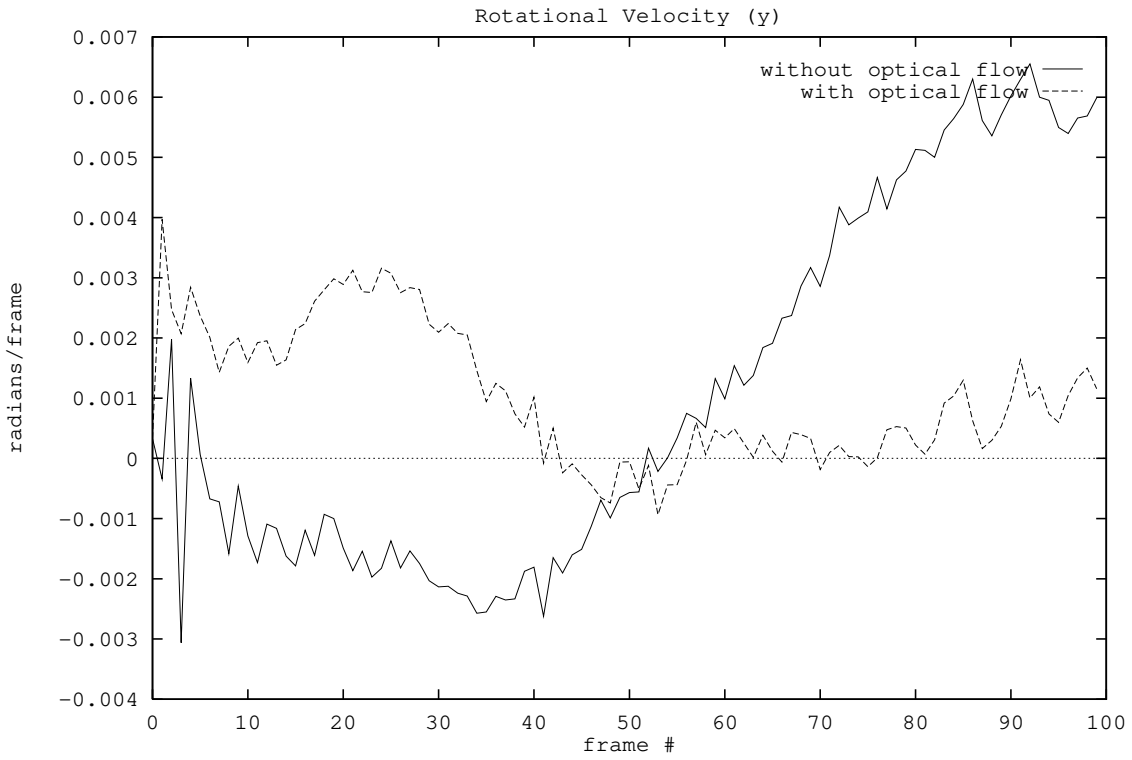


(e)

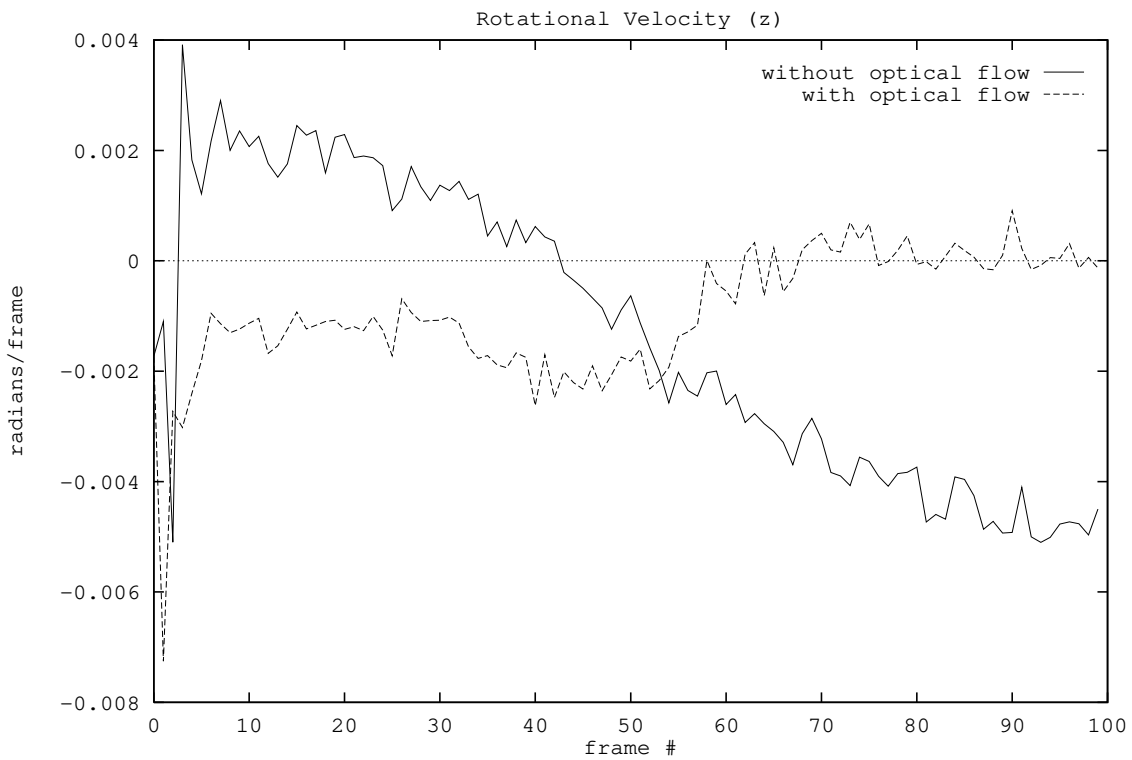


(f)

Figure 3.15: Experiment 5: FDM with correlated errors (bias)

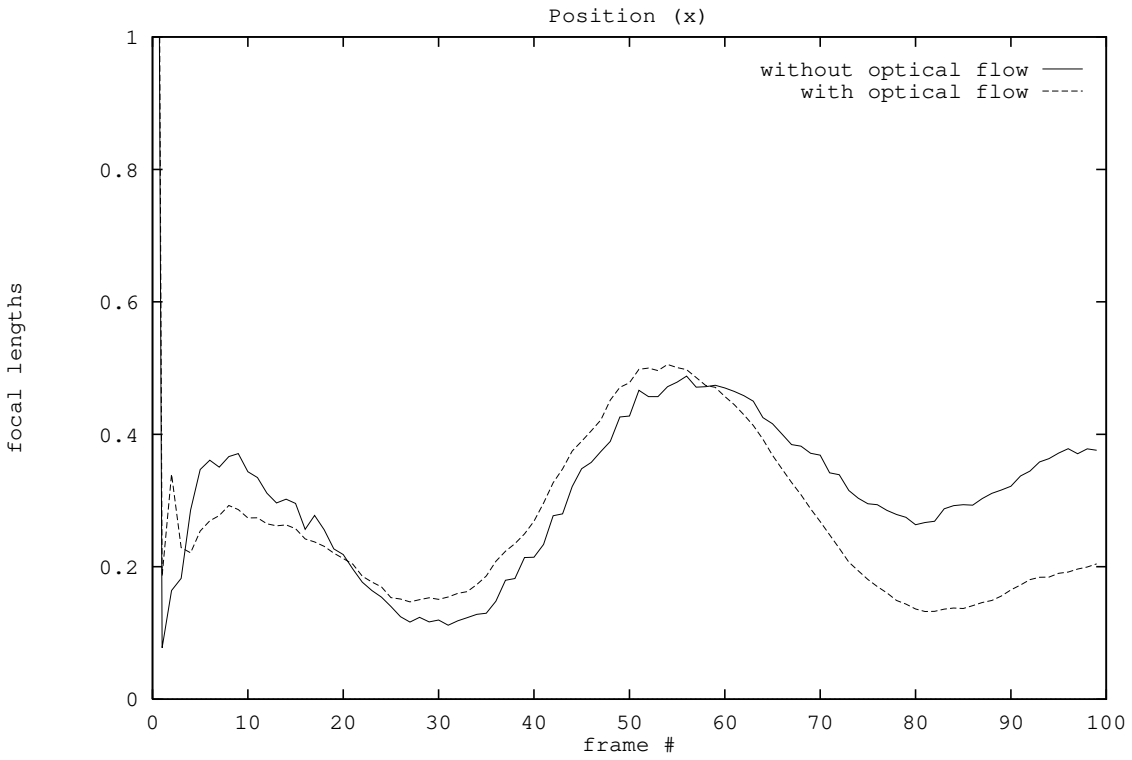


(g)

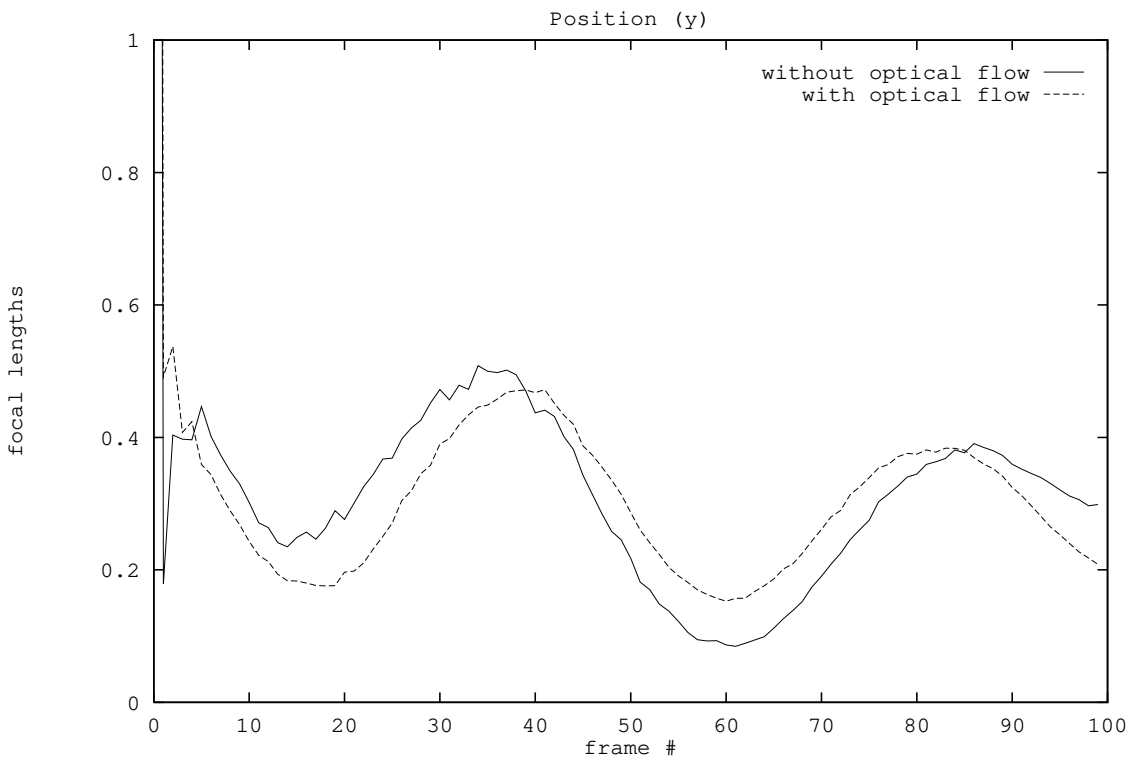


(h)

Figure 3.15: Experiment 5: FDM with correlated errors (bias)

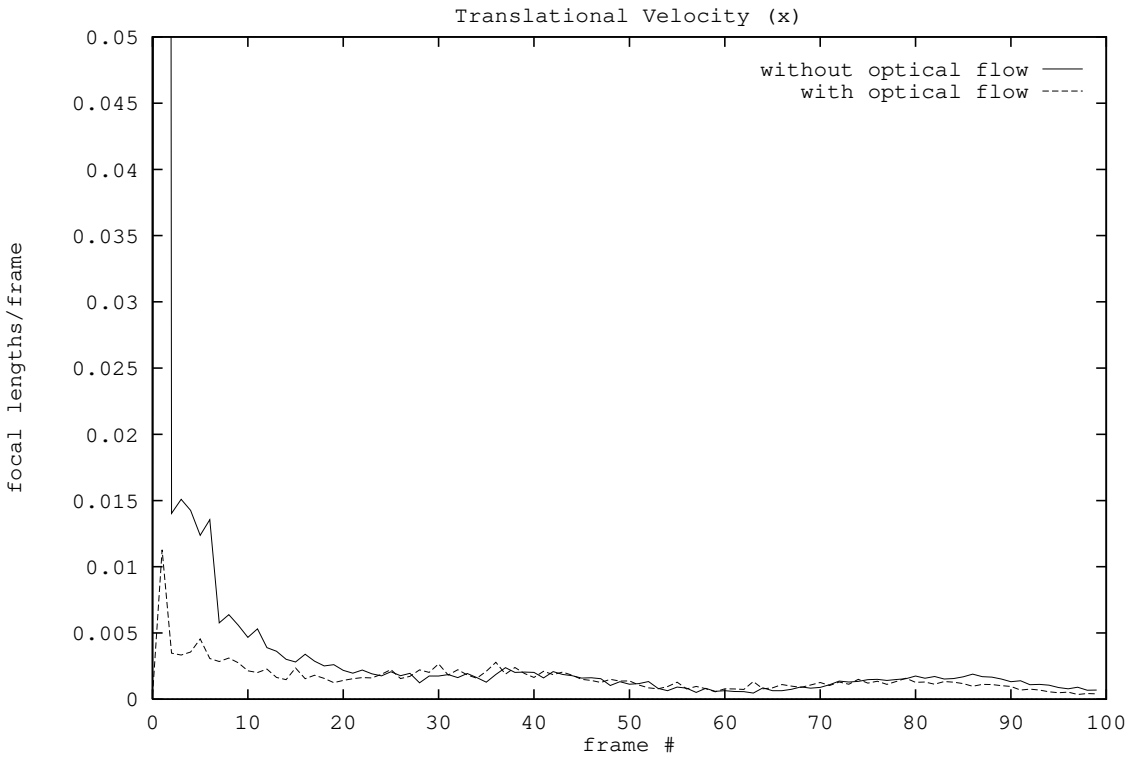


(a)

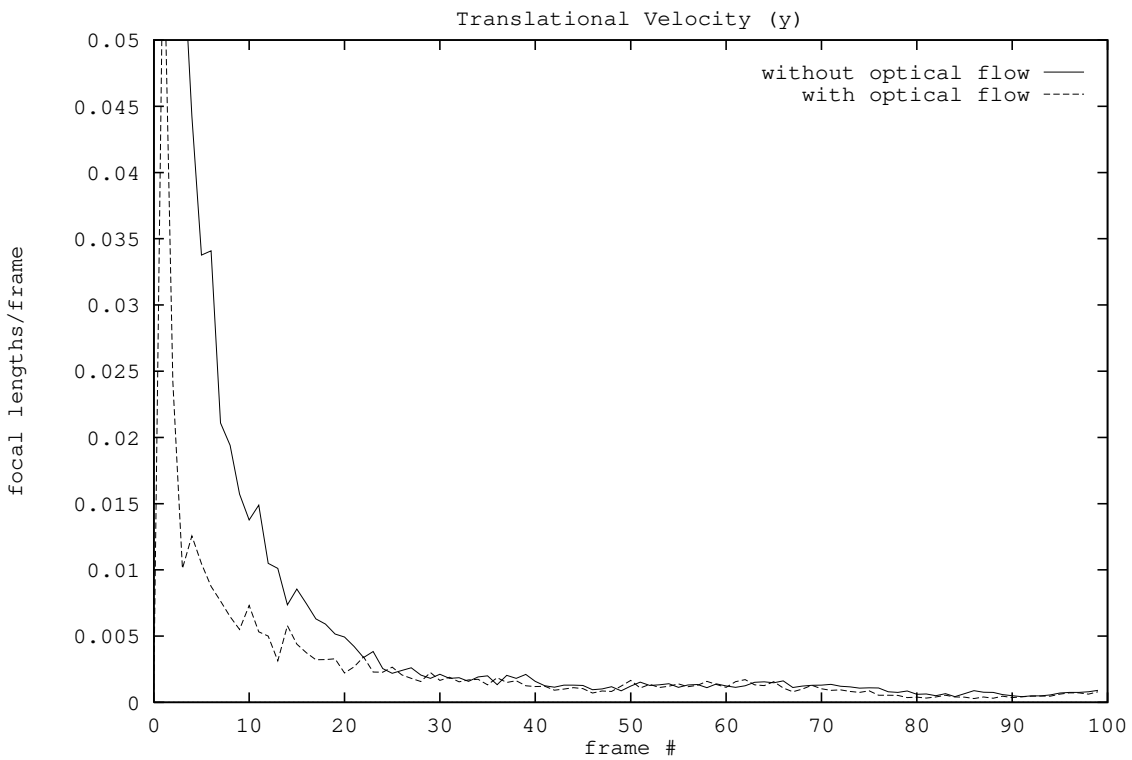


(b)

Figure 3.16: Experiment 5: FDM with correlated errors (variance)

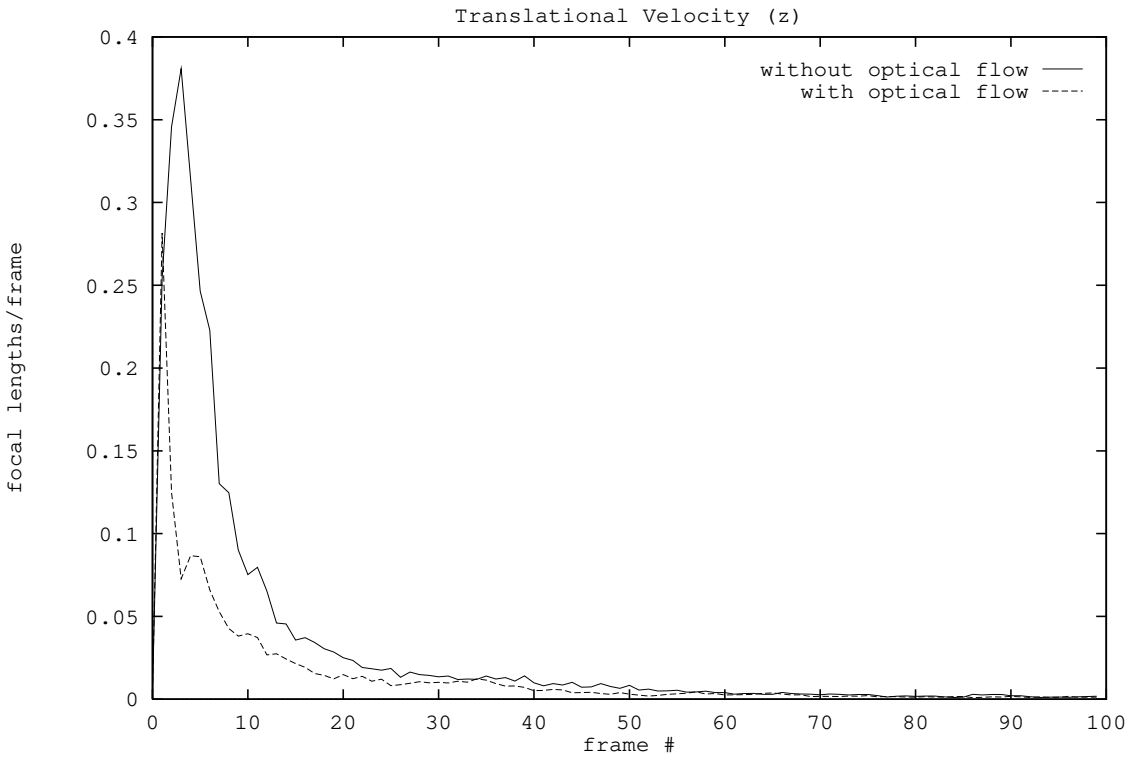


(c)

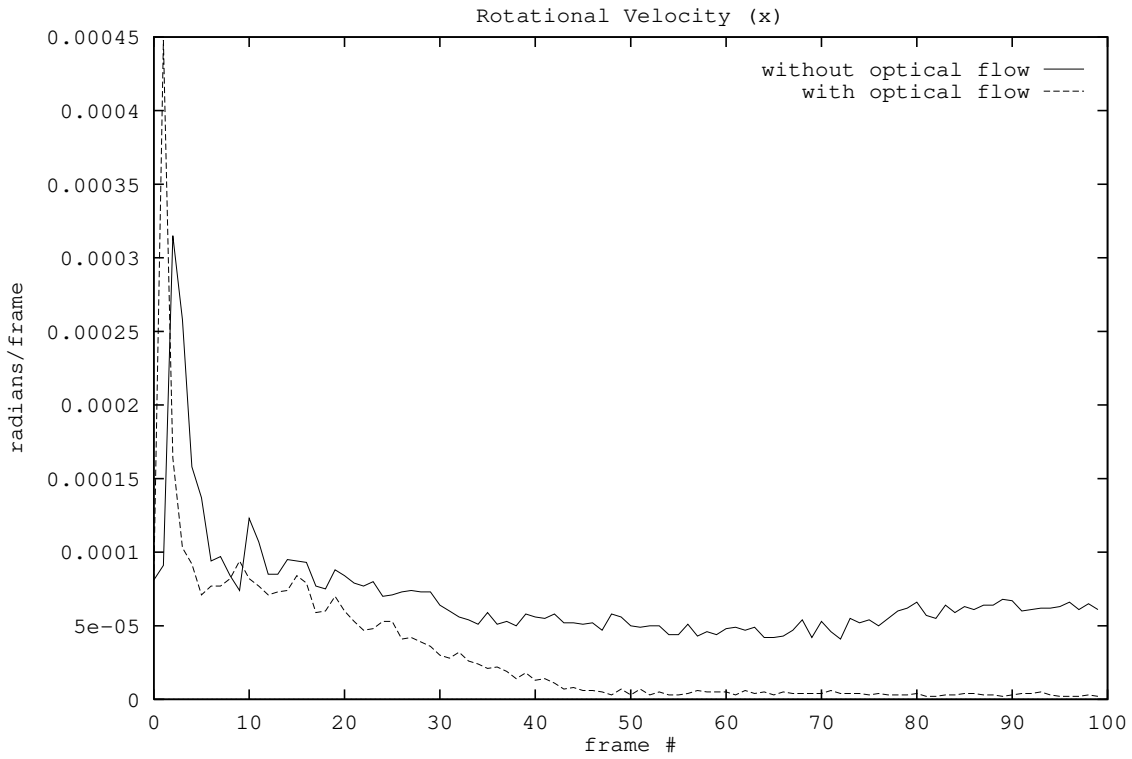


(d)

Figure 3.16: Experiment 5: FDM with correlated errors (variance)

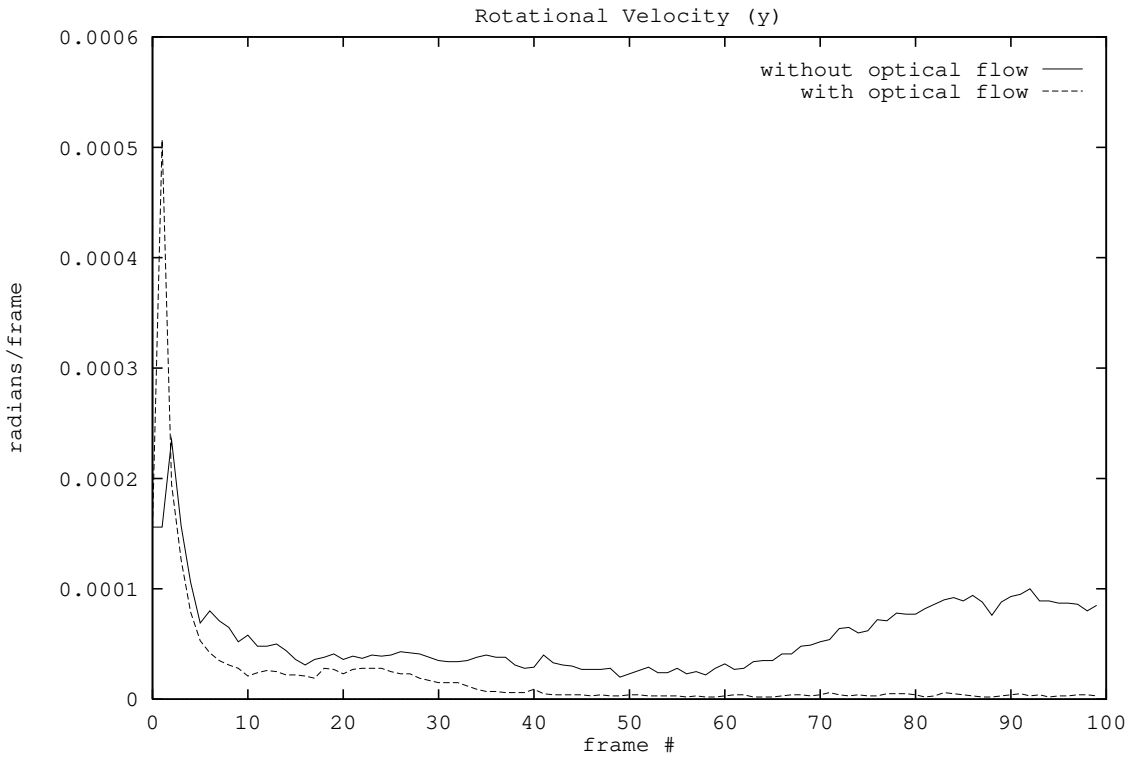


(e)

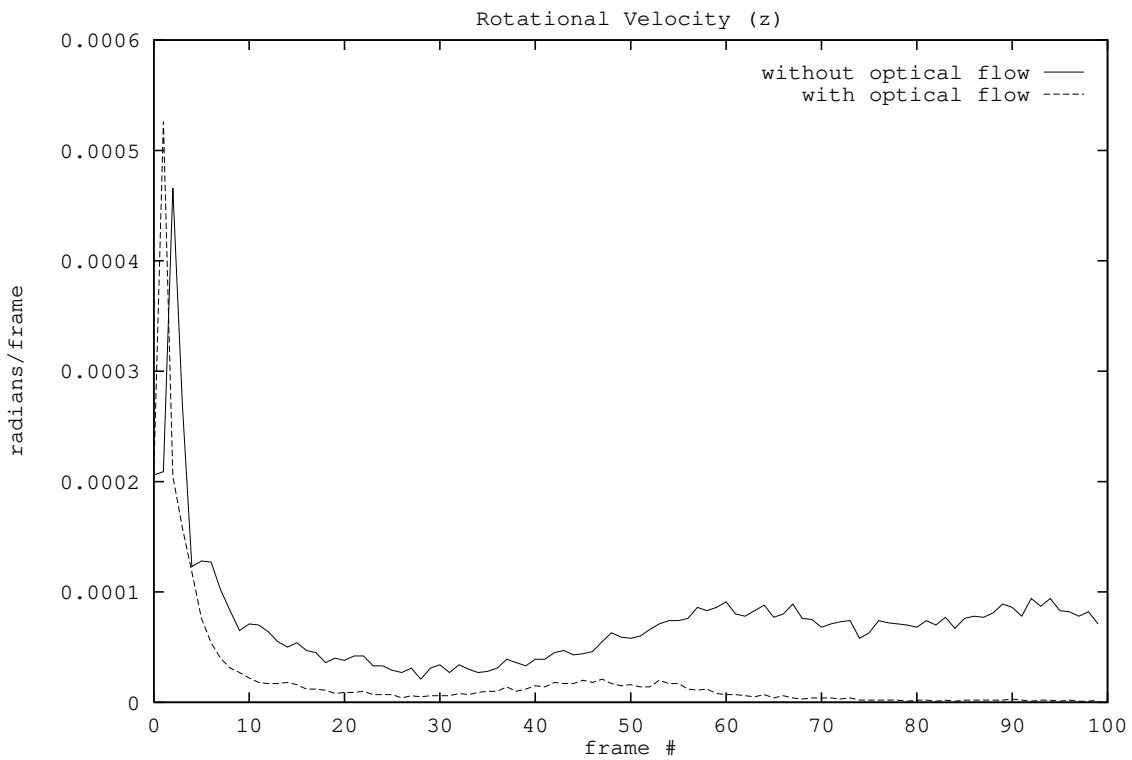


(f)

Figure 3.16: Experiment 5: FDM with correlated errors (variance)



(g)



(h)

Figure 3.16: Experiment 5: FDM with correlated errors (variance)

Rearranging (3.52), we get

$$\dot{x}(t) = \frac{x(t+\Delta t) - x(t)}{\Delta t} - \int \frac{1}{2}\Delta t^2 \ddot{x}(t)dt. \tag{3.53}$$

The feature displacement vector method is exact if and only if the second term on the right hand side, or the image plane acceleration $\ddot{x}(t)$, is zero. In other words, even if there is no error in the position measurements, the optical flow measurements obtained using the feature displacement method is still only an approximation to the true optical flow, unless the image plane velocity is constant, or equivalently, the image plane acceleration is zero.

Secondly, one of the basic assumptions of the Kalman filter states that the measurement noise should be temporally uncorrelated. The errors in the displacement vectors at t_k and t_{k+1}, which are dependent on the noisy position measurements at times t_k, t_{k+1}, and t_{k+2}, are

$$\frac{\delta_{X_1} - \delta_{X_2}}{\Delta t} \tag{3.54}$$

and

$$\frac{\delta_{X_2} - \delta_{X_3}}{\Delta t}. \tag{3.55}$$

If we examine the correlation between the optical flow measurement errors at times t_k and t_{k+1},

$$E\left\{\left(\frac{\delta_{X_2} - \delta_{X_1}}{\Delta t}\right)\left(\frac{\delta_{X_3} - \delta_{X_2}}{\Delta t}\right)\right\} = \frac{1}{4}E\{\delta_{X_2}^2\} \tag{3.56}$$

where we have made the assumption that $E\{\delta_{X_i}\delta_{X_j}\} = 0$ for all $i \neq j$, we will see that the result is non-zero unless the position measurements are error-free. It is worth investigating if the existence of temporal correlation in the measurement noise affects the performance of the FDM version of the hybrid algorithm. But, first, we need to devise a way to simulate measurement errors that are uncorrelated. Fortunately, uncorrelatedness in the measurement noise can be achieved by computing one estimate for every two frames. More precisely, the state estimate for frame k is produced using measurements from frames k and $k+1$ whereas frames $k+2$ and $k+3$ are used to arrive at the state estimate for frame $k+2$. Note that no estimate is produced

for frame $k+1$. We have repeated our experiment with this revised approach and the results, shown in Figures 3.17 and 3.18, suggest that the approximation error inherent in the feature displacement method is a much more dominant factor than the correlatedness in the measurement noise.

3.5 Discussions

One possible drawback of the hybrid feature/flow-based algorithm (as well as the original BCC algorithm) is that feature correspondence must be maintained throughout the entire image sequence. In our simulations, we avoided the problem of self-occlusion, or the disappearance of feature points due to object's rotational motion. We simply treated the object as if it were transparent. One way to tackle this problem is to minimize the number of image frames in which we need to establish feature correspondence. This will be discussed further in Chapter 4.

Moreover, due to certain computational constraints, it may be necessary to reduce the size of the state vector. One way to achieve this goal is to eliminate the structure and quaternion states. Eliminating the former is quite involved and will be discussed further in Chapter 4. If the rotational velocity is known to be constant a priori, then the quaternion states can be removed as follows: instead of propagating the rotation matrix using the unit quaternion, we simply use the rotational velocity vector. More specifically, the rotation matrix can be propagated using the relationship [98]

$$R(t) \;=\; R_0[n_1, n_2, n_3; \theta] \tag{3.57}$$

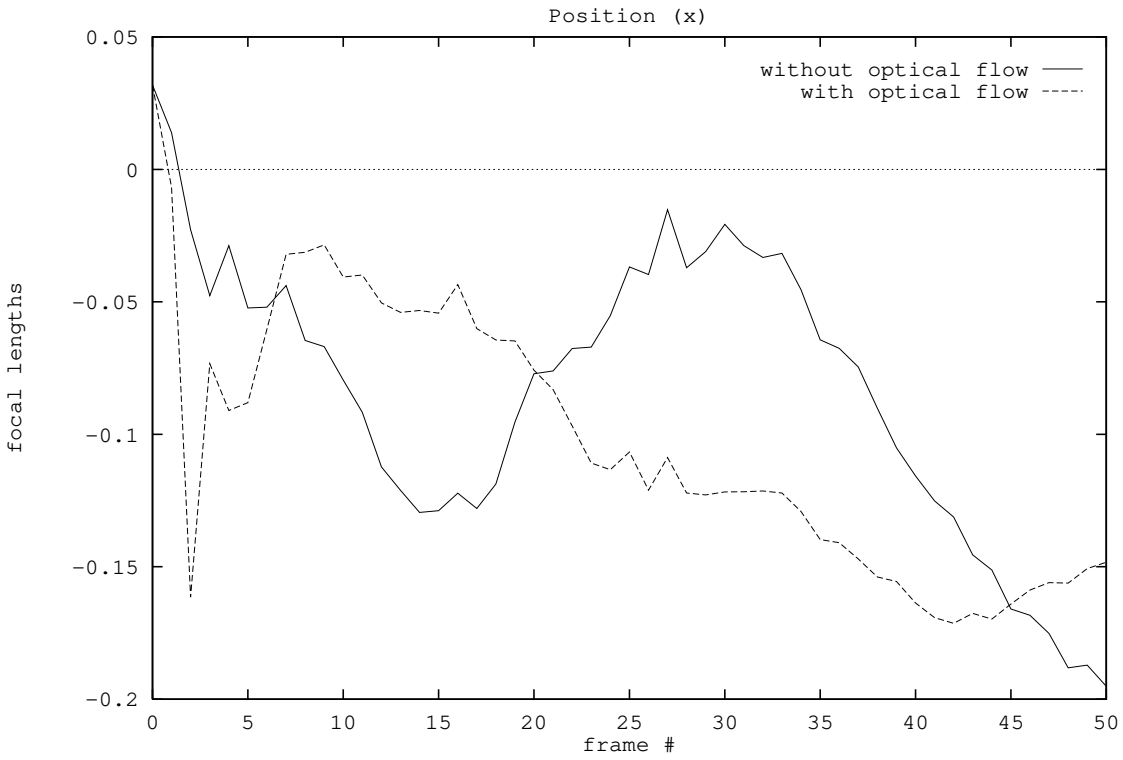
where

$$R_0[n_1, n_2, n_3; \theta] = \begin{bmatrix} n_1^2 + (1-n_1^2)c & n_1 n_2(1-c) - n_3 s & n_1 n_3(1-c) + n_2 s \\ n_1 n_2(1-c) + n_3 s & n_2^2 + (1-n_2^2)c & n_2 n_3(1-c) - n_1 s \\ n_1 n_3(1-c) - n_2 s & n_2 n_3(1-c) + n_1 s & n_3^2 + (1-n_3^2)c \end{bmatrix} \tag{3.58}$$

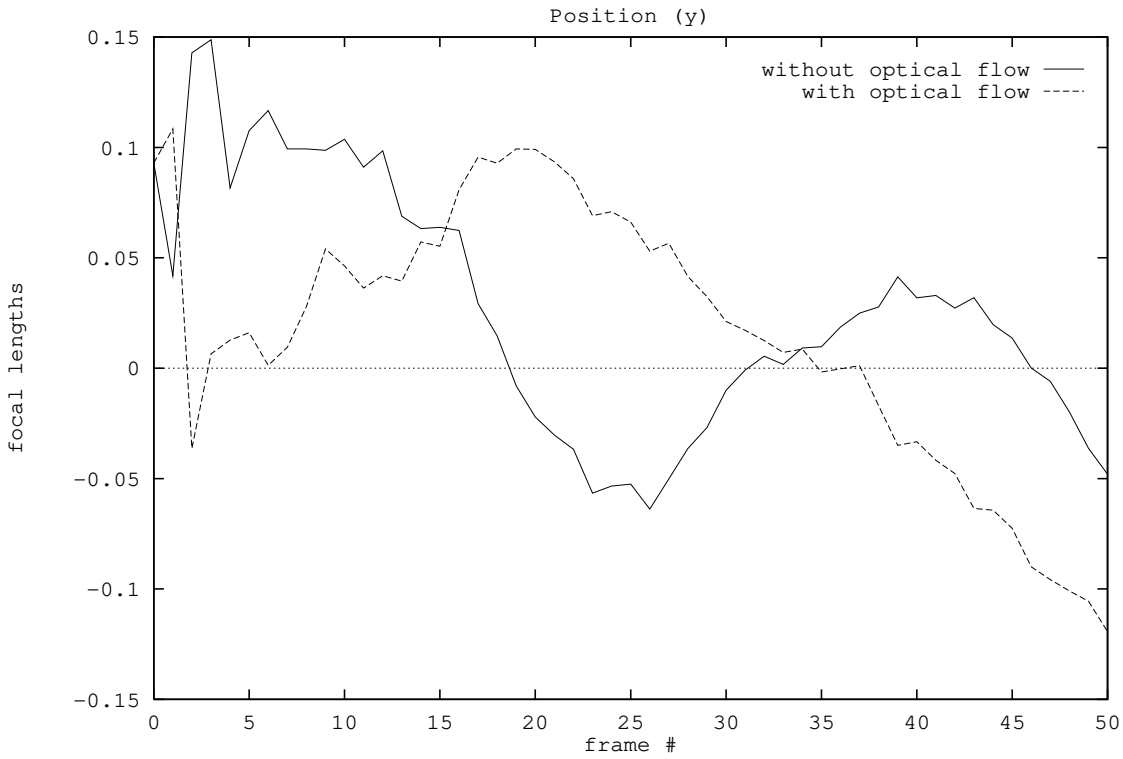
and

$$n_1 = \frac{\omega_x}{|\omega|}; \quad n_2 = \frac{\omega_y}{|\omega|}; \quad n_3 = \frac{\omega_z}{|\omega|} \tag{3.59}$$

$$c = \cos\theta; \quad s = \sin\theta; \quad \theta = |\omega|(t - t_0). \tag{3.60}$$

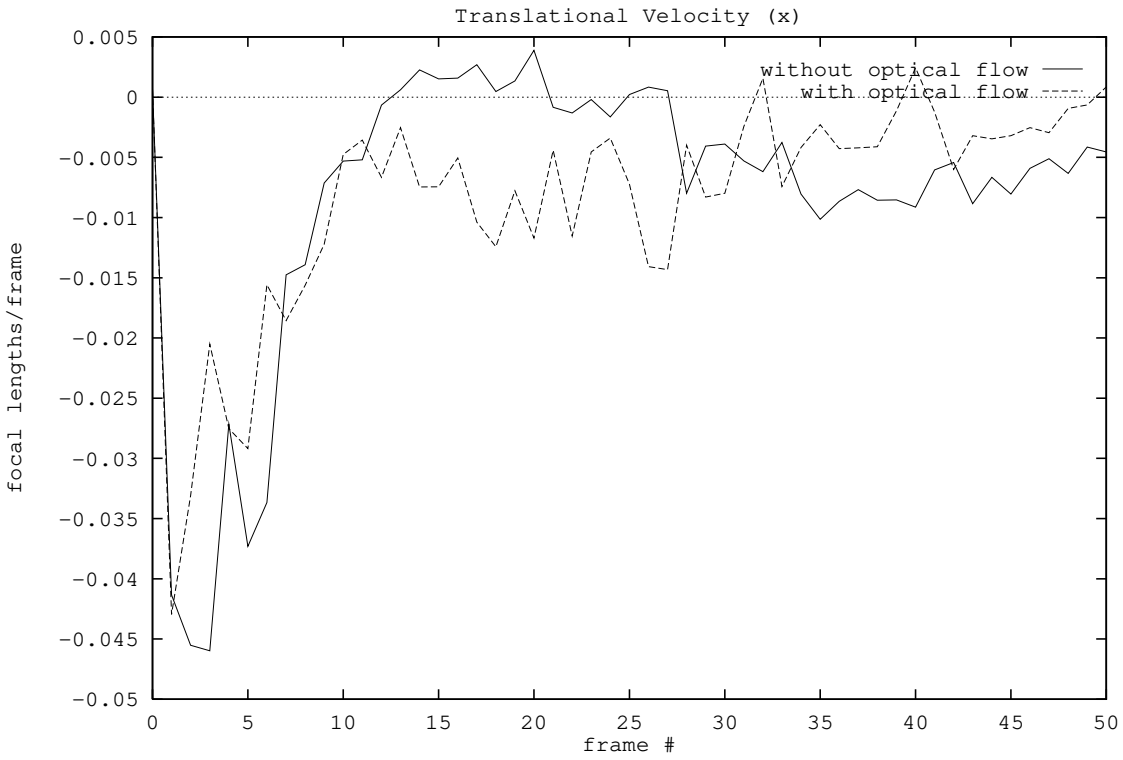


(a)

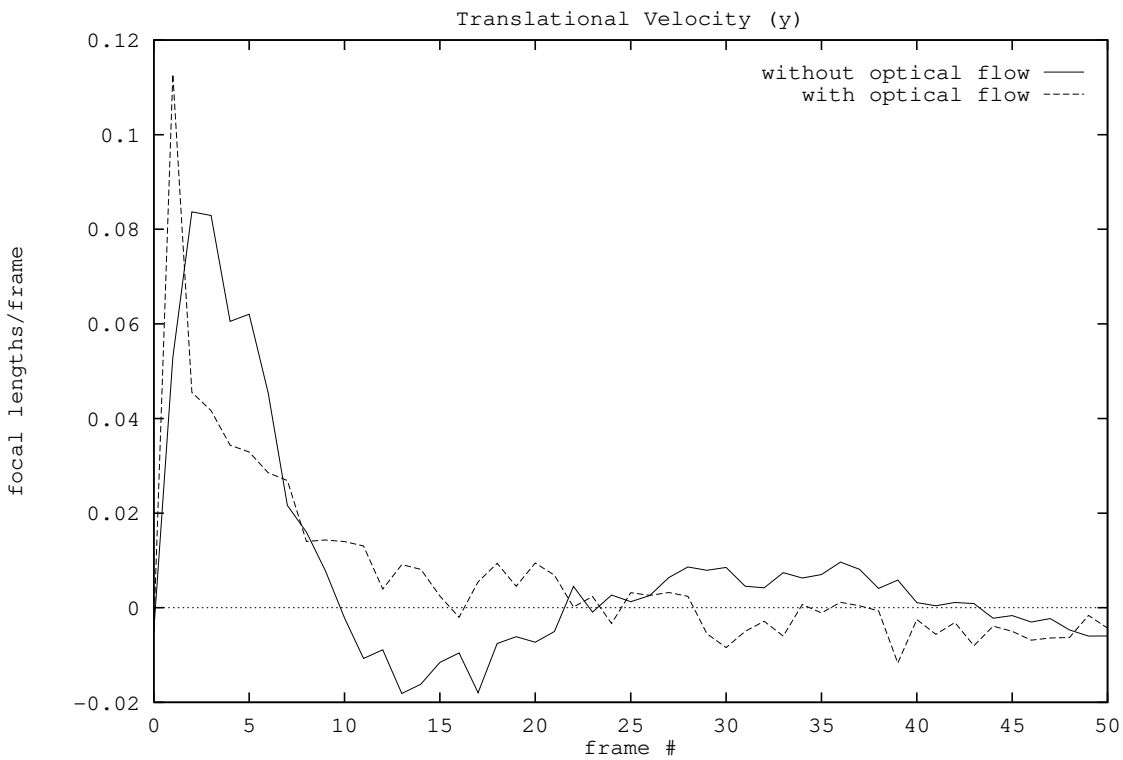


(b)

Figure 3.17: Experiment 5: FDM with uncorrelated errors (bias)

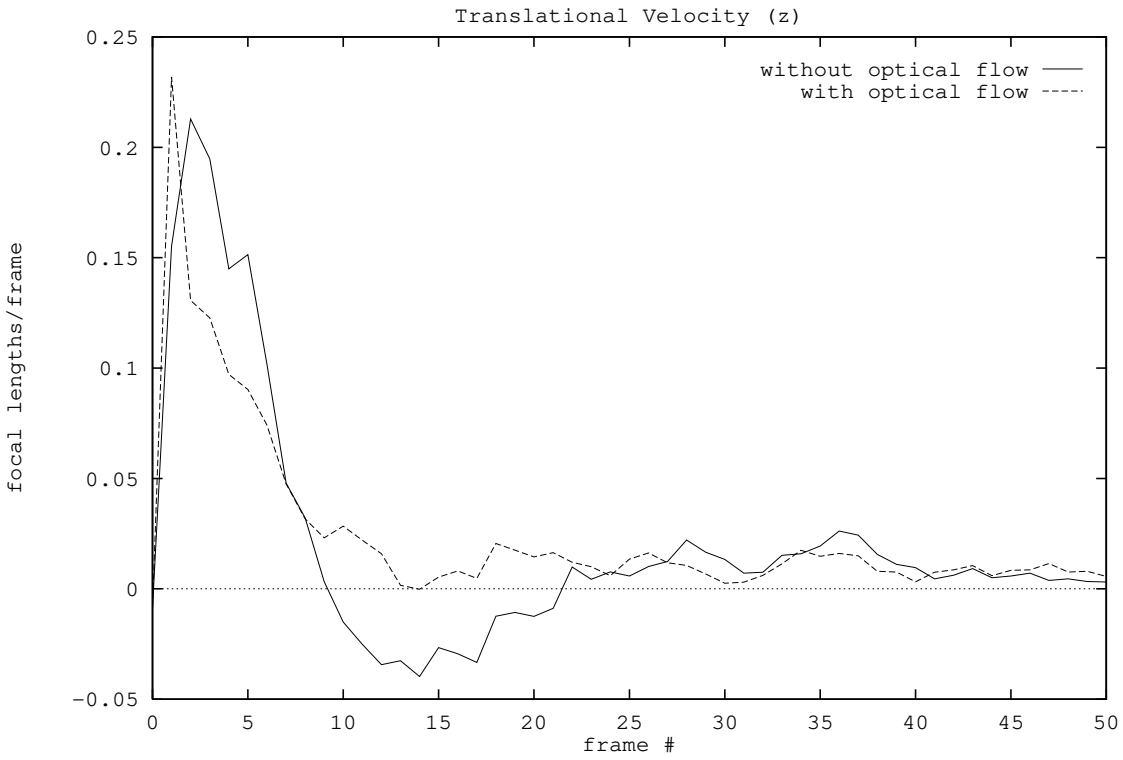


(c)

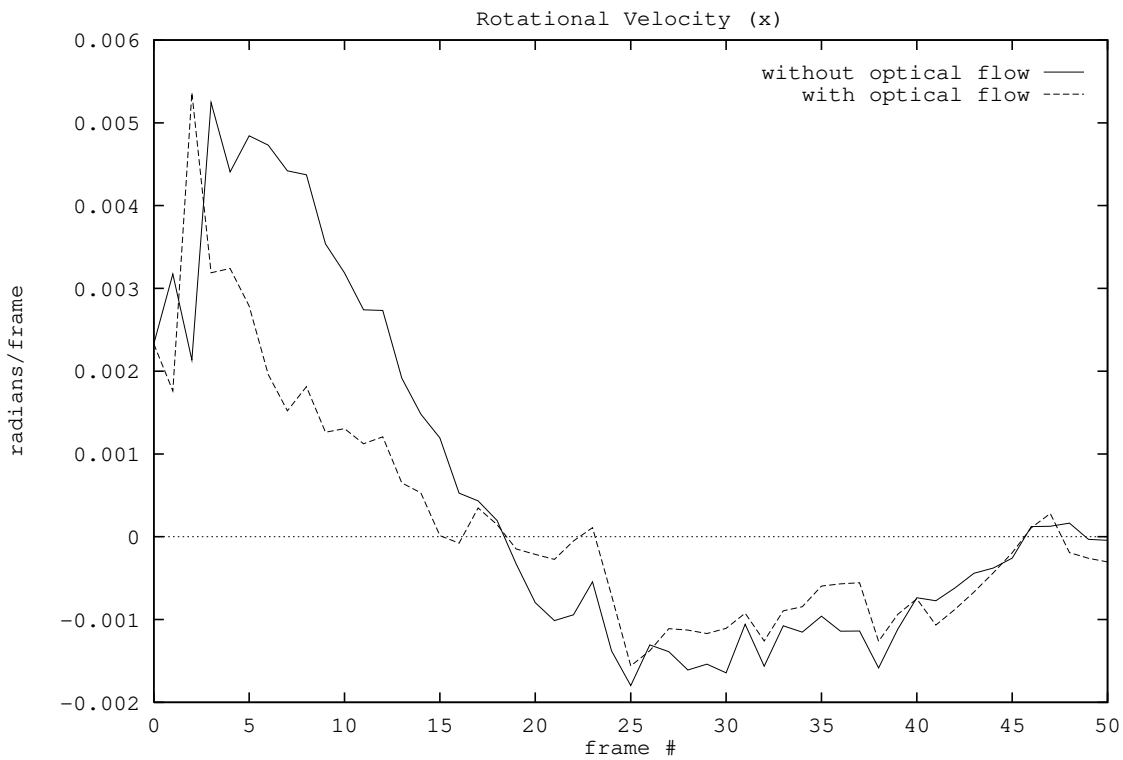


(d)

Figure 3.17: Experiment 5: FDM with uncorrelated errors (bias)

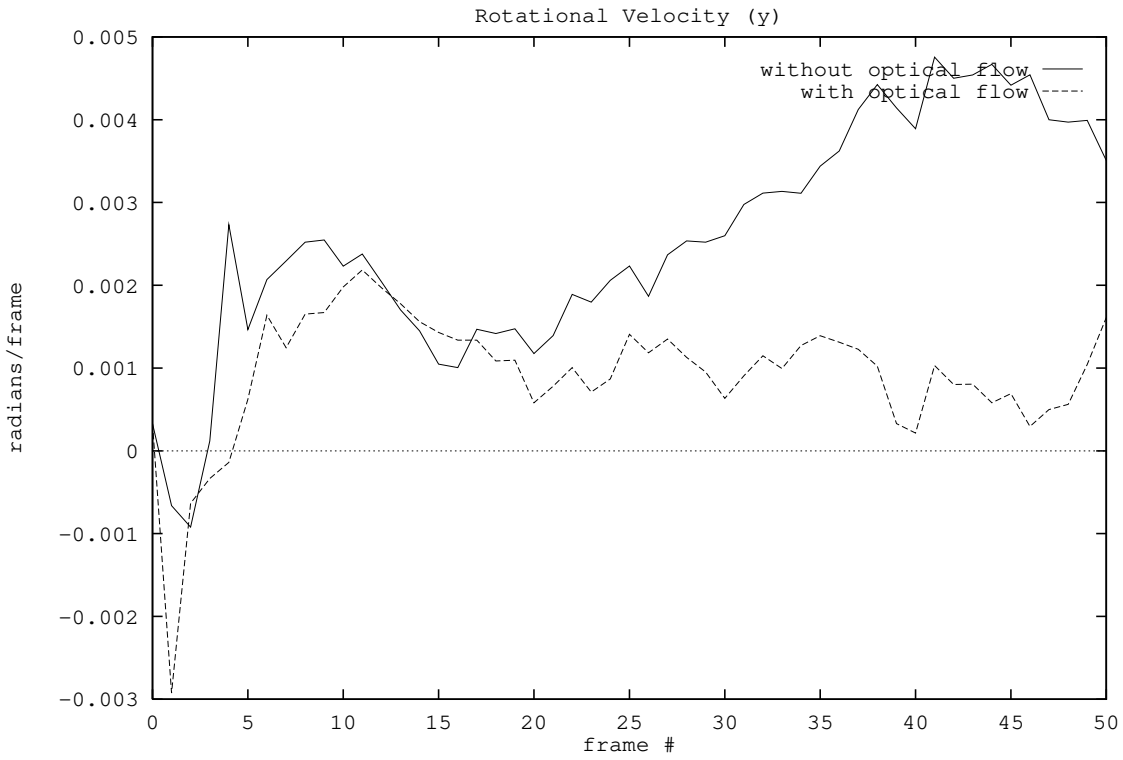


(e)

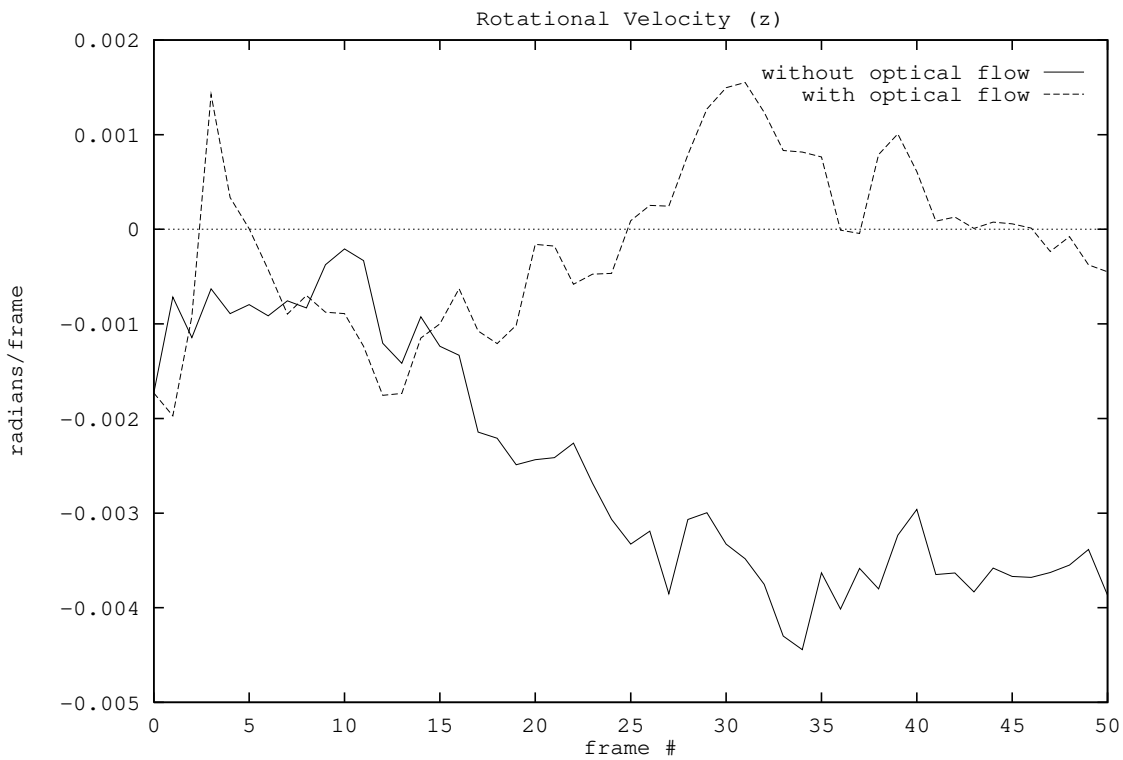


(f)

Figure 3.17: Experiment 5: FDM with uncorrelated errors (bias)

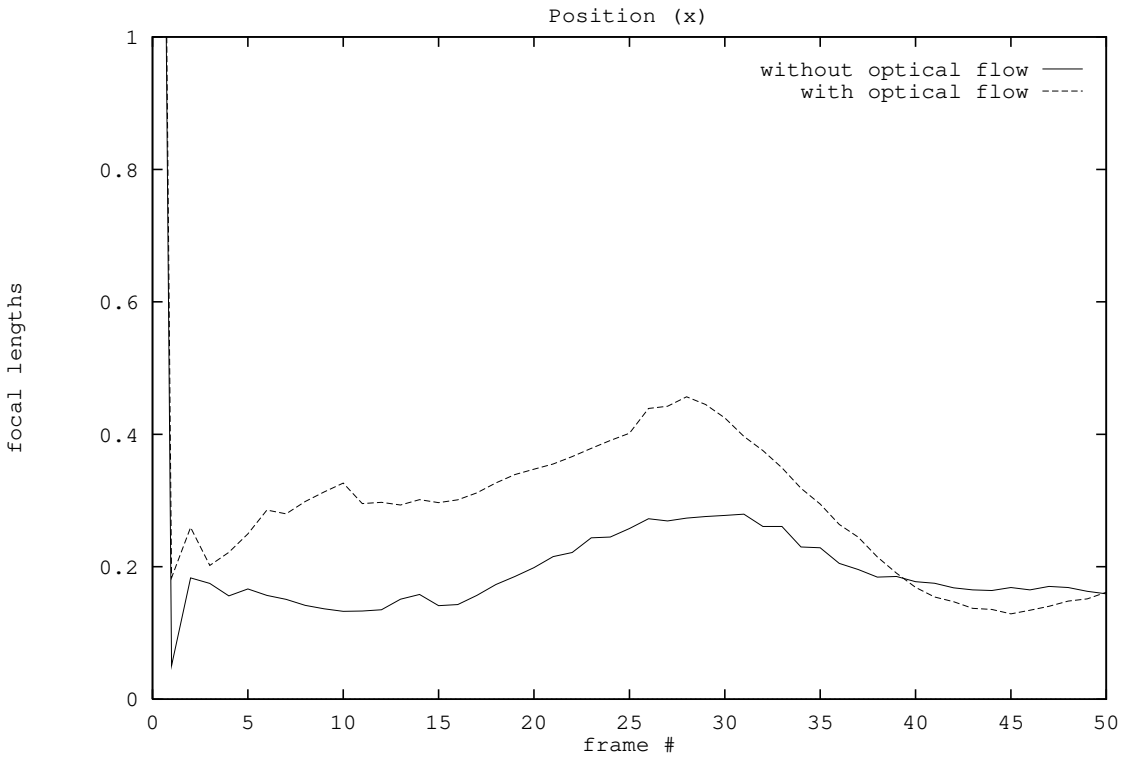


(g)

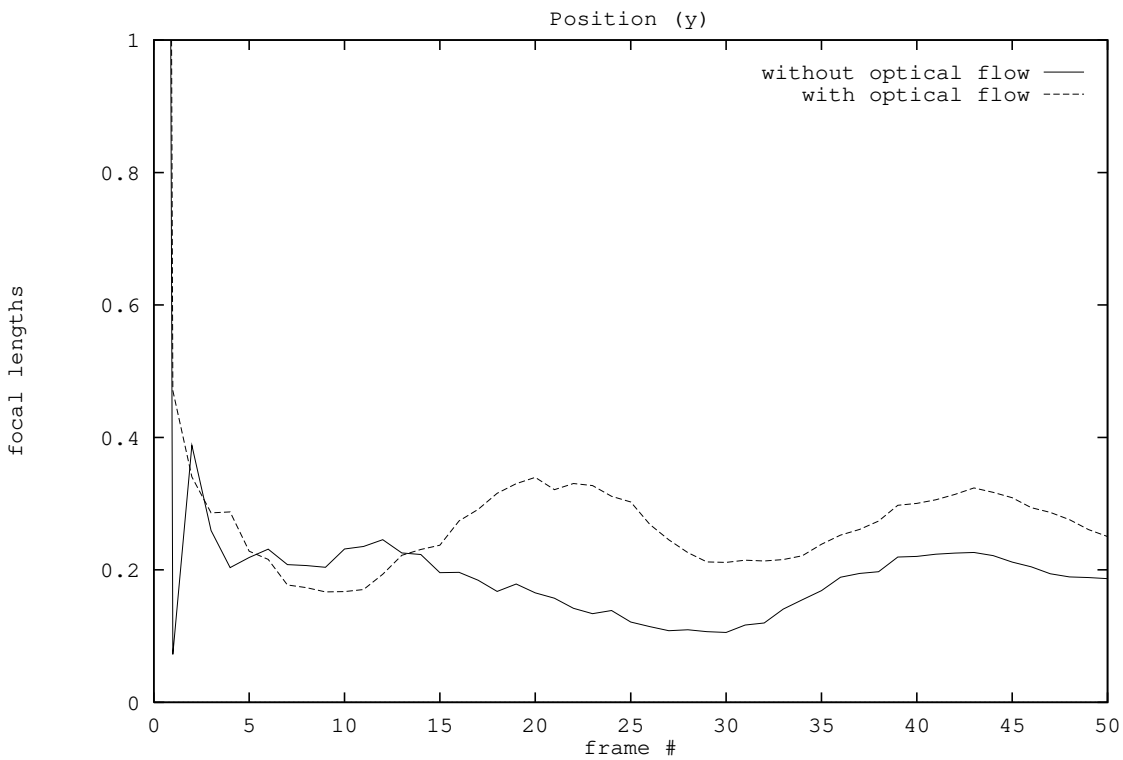


(h)

Figure 3.17: Experiment 5: FDM with uncorrelated errors (bias)

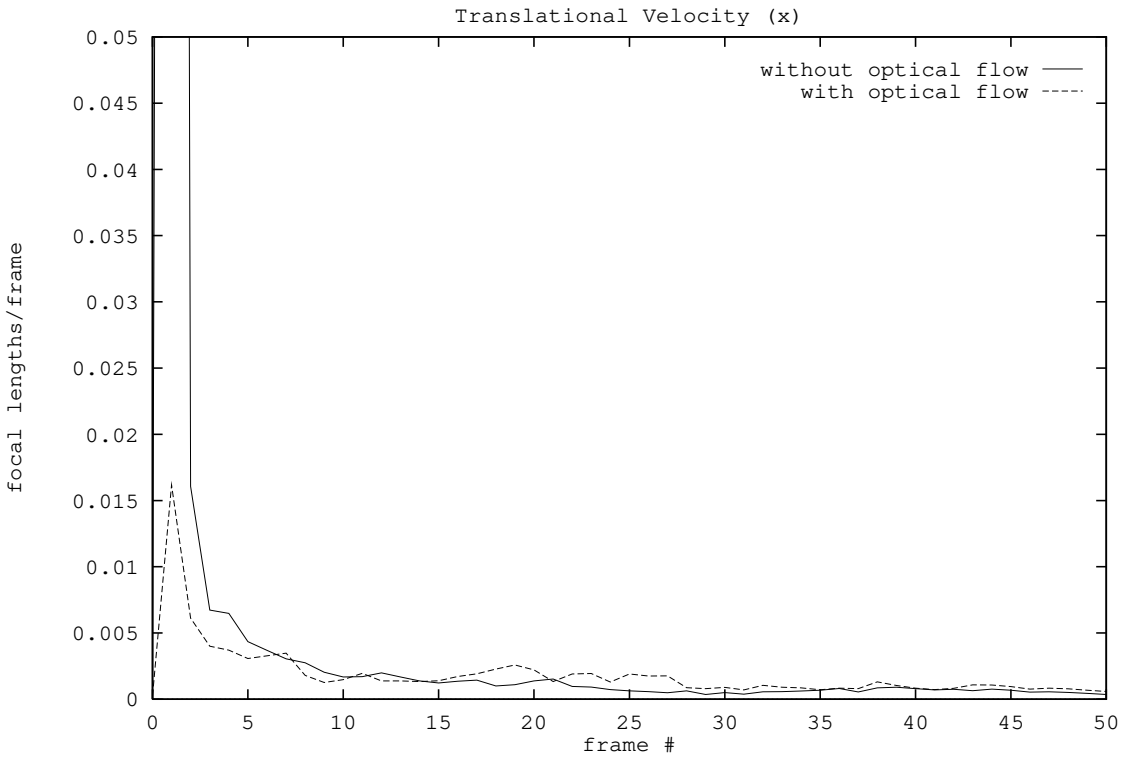


(a)

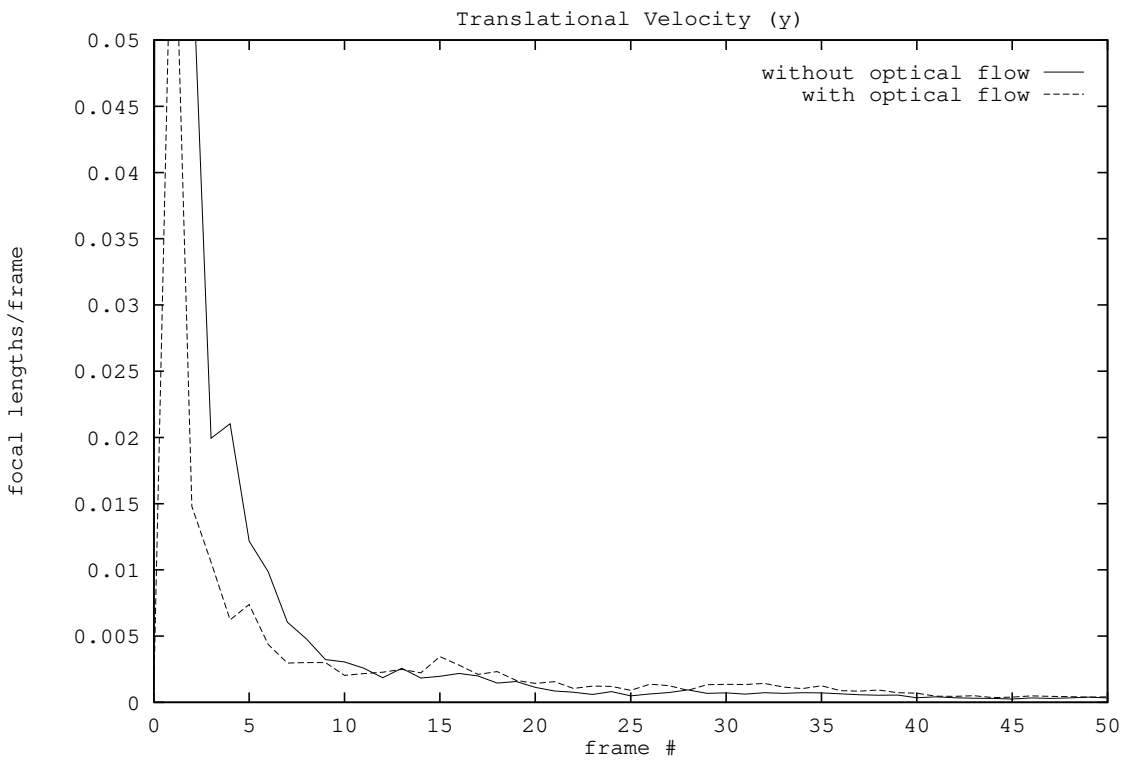


(b)

Figure 3.18: Experiment 5: FDM with uncorrelated errors (variance)

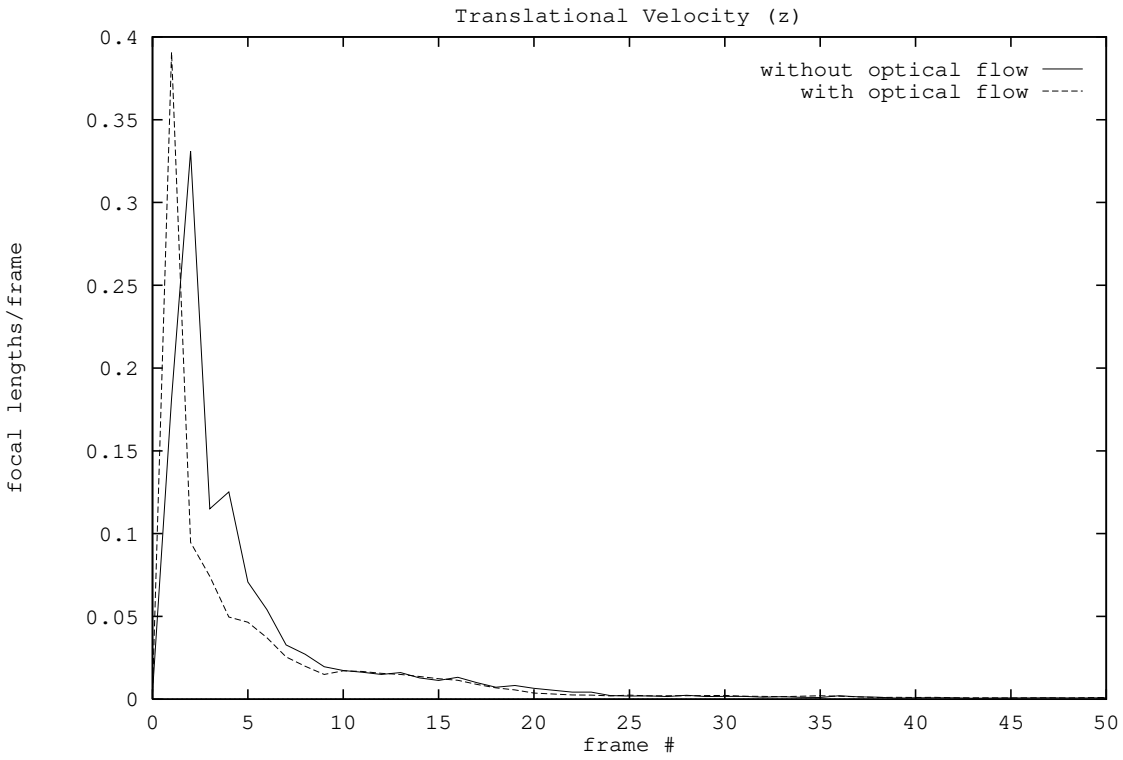


(c)

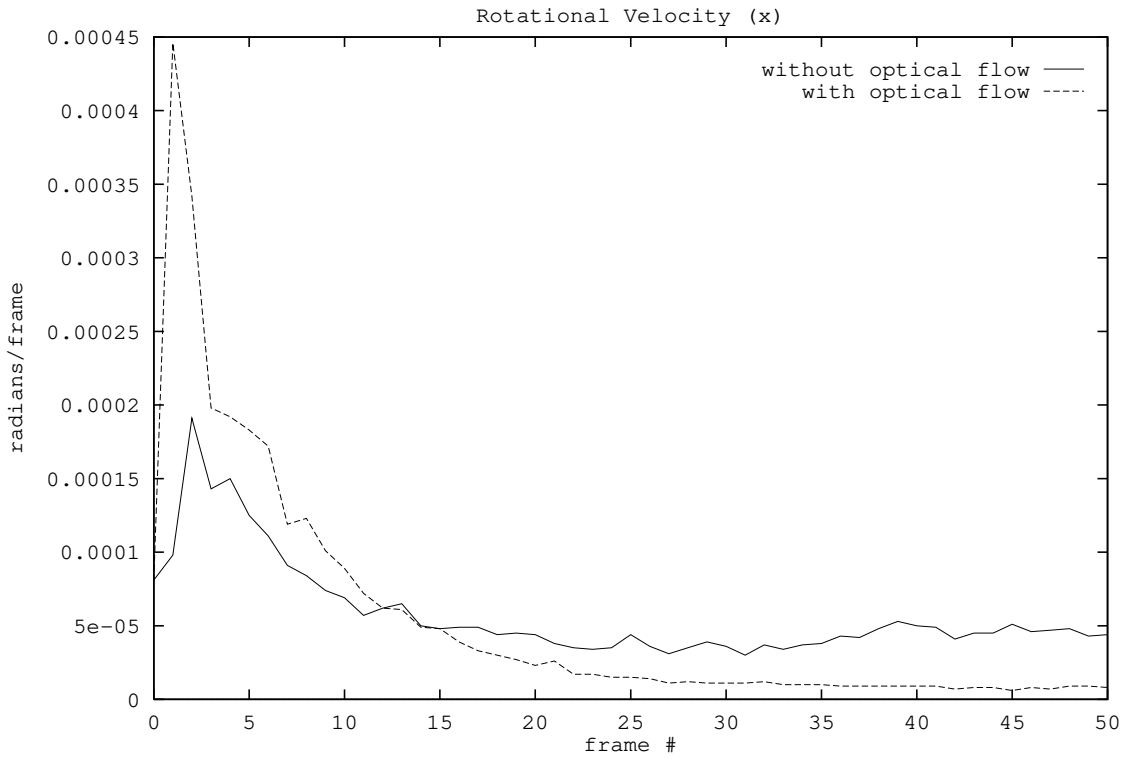


(d)

Figure 3.18: Experiment 5: FDM with uncorrelated errors (variance)

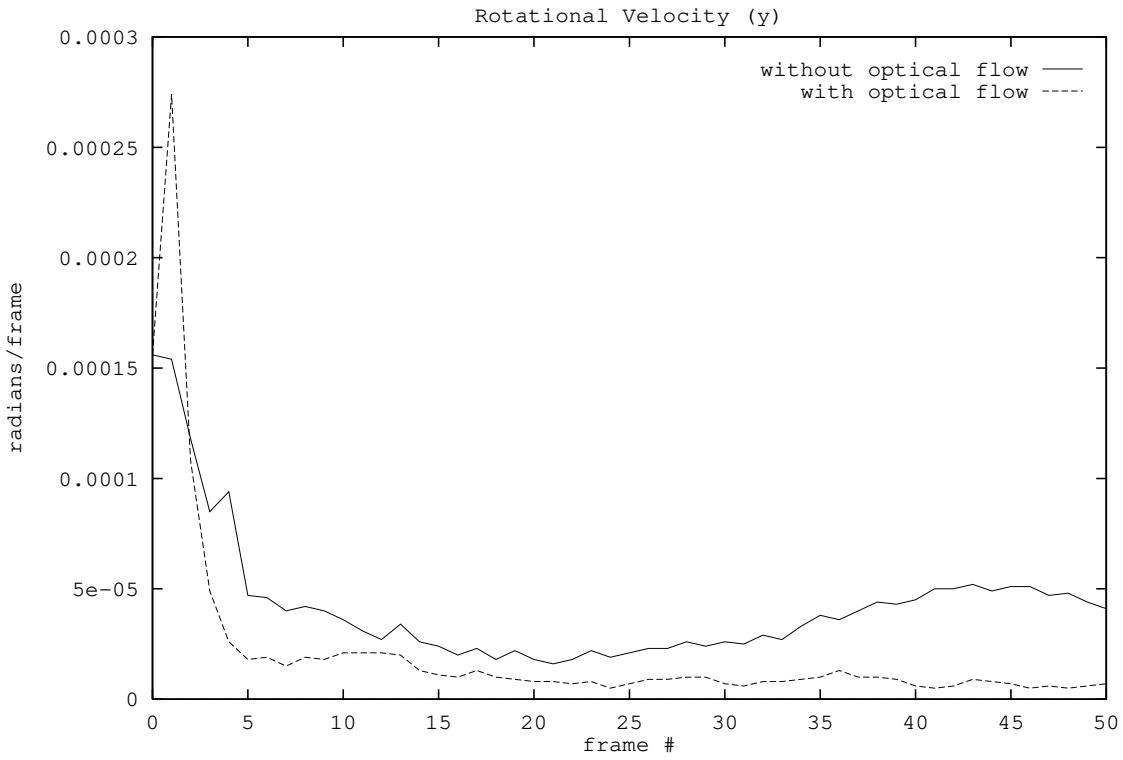


(e)

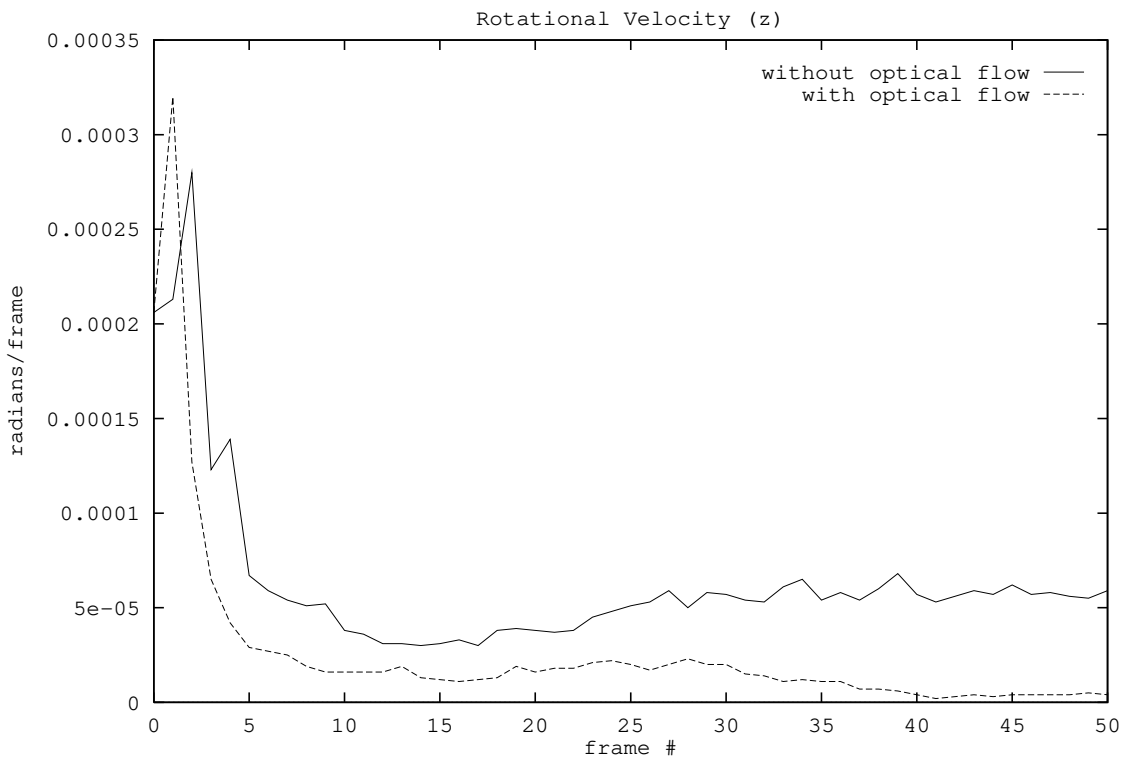


(f)

Figure 3.18: Experiment 5: FDM with uncorrelated errors (variance)



(g)



(h)

Figure 3.18: Experiment 5: FDM with uncorrelated errors (variance)

The resulting measurement function is fairly complex and the process of obtaining the corresponding H matrix is quite involved. Thus, no experimental results have been obtained.

3.6 Summary

We began this chapter by giving an overview of the BCC algorithm. Then we presented a new hybrid algorithm that is based on the same recursive framework but makes use of optical flow measurements in addition to feature correspondences. A simulation system designed to evaluate this new technique was discussed, as well as some of the implementation issues. Experimental results showed that as long as the relative error in the optical flow measurements is less than about 100%, significant improvements in the estimated trajectory can be observed. The feature displacement method, which serves mainly as a baseline for comparison, does not achieve such improvements. The hybrid algorithm also offers superior performance, compared to the original BCC algorithm, during filter initialization and abrupt changes in the motion model. Finally, we discussed the shortcomings of the hybrid algorithm and suggested a number of possible improvements.

Chapter 4

MEASUREMENT TRANSFORMATION

Despite the fact that the hybrid feature/flow-based algorithm introduced in Chapter 3 offers better performance than the algorithm proposed by Broida *et al.* [15], it has a number of disadvantages:

- feature points must be tracked over the entire image sequence which may have a large number of frames
- the motion model consists of a time-varying normalization factor
- due to the non-linearities inherent in the system and measurement models, an optimal linear Kalman filter cannot be used
- the size of the state vector grows as the number of feature points increases

In this chapter we will introduce the concept of measurement transformation to formulate an algorithm that is capable of eliminating all of these problems. Unfortunately, a complete implementation is not available and two new problems were discovered in the process of developing this new algorithm.

Consider the two most "basic" kinds of measurements in 3-D motion estimation discussed in Chapter 2: feature correspondences and optical flow fields. The basic idea behind measurement transformation is that these basic measurements could be "transformed" into more useful quantities. Recall that in the recursive framework proposed by Broida *et al.* , the 3-D motion and structure of an object is modelled

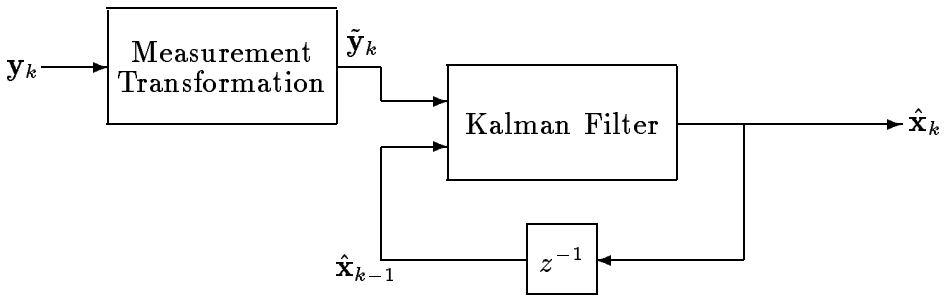


Figure 4.1: Kalman filter with measurement transformation

as a dynamical system and the state of this system is estimated recursively using an approximate non-linear Kalman filter. We propose an intermediate step to process the position measurements before using them as input to the Kalman filter (Figure 4.1). By carefully designing the transformation, certain states, such as the 3-D position and orientation, can be observed more directly. In fact, this is exactly what the two-view motion and structure estimation algorithm proposed by Weng *et al.* [91] allows us to do. The use of two-view motion estimation within a recursive framework eliminates the need to track feature points over many image frames and improves the quality of the estimated motion parameters over time.

4.1 Overview of the Weng-Ahuja-Huang Two-View Motion Estimation Algorithm

Consider the 3-D coordinates of a point on a rigid object at time t_k

$$\mathbf{P} = [X \;\; Y \;\; Z]^T \tag{4.1}$$

and the 3-D coordinates of the same point at time t_{k+1}

$$\mathbf{P}' = [X' \;\; Y' \;\; Z']^T. \tag{4.2}$$

The relationship between them can be expressed as

$$\mathbf{P}' = R\mathbf{P} + \mathbf{T} \tag{4.3}$$

where R is a rotation matrix and $\mathbf{T}$ is a translation vector. The projections of $\mathbf{P}$ and $\mathbf{P}'$ onto the image plane, via unity focal length perspective projection, are

$$\mathbf{p} = \left[\frac{X}{Z} \quad \frac{Y}{Z} \quad 1\right]^T \tag{4.4}$$

$$\mathbf{p}' = \left[\frac{X'}{Z'} \quad \frac{Y'}{Z'} \quad 1\right]^T. \tag{4.5}$$

Using (4.4) and (4.5) we can rewrite (4.3) as

$$Z'\mathbf{p}' = RZ\mathbf{p} + \mathbf{T}. \tag{4.6}$$

Dividing (4.6) by $\|\mathbf{T}\|$, we get

$$\tilde{Z}'\mathbf{p}' = R\tilde{Z}\mathbf{p} + \hat{\mathbf{T}} \tag{4.7}$$

where $\hat{\mathbf{T}} \triangleq \mathbf{T}/\|\mathbf{T}\|$, $\tilde{Z} \triangleq Z/\|\mathbf{T}\|$, and $\tilde{Z}' \triangleq Z'/\|\mathbf{T}\|$. The two-view motion and structure estimation algorithm developed by Weng *et al.* [91] (called the WAH algorithm hereafter) recovers the interframe motion parameters, R and $\hat{\mathbf{T}}$, as well as the relative depths, $\tilde{Z}$ and $\tilde{Z}'$.

Using the measured image plane coordinates of n feature points (where $n \geq 8$), an $n \times 9$ matrix A is first formed. The next step is to find a unit vector $\mathbf{h}$ such that $\|A\mathbf{h}\|$ is minimized. According to Rayleigh's principle [81], the solution is the eigenvector associated with the smallest eigenvalue of A^TA. The components of $\mathbf{h}$ are then used to form a 3×3 matrix E consisting of the so-called "essential parameters" [28, 92]. By computing the eigenvector associated with the smallest eigenvalue of EE^T, one obtains the direction of translation (except for the sign). Having computed the direction of translation, the rotation matrix can be calculated by solving another eigenvalue problem. Finally, the proper sign of the direction of translation is determined and the relative depths, $\mathbf{Z}_i$, are found by solving

$$\|[\mathbf{p}_i' \quad -R\mathbf{p}_i]\mathbf{Z}_i - \hat{\mathbf{T}}\| = \min, \qquad i = 1, \ldots, n \tag{4.8}$$

where $\mathbf{p}_i$ and $\mathbf{p}_i'$ are the image plane position measurements of the i-th feature, R is the interframe rotation matrix, $\hat{\mathbf{T}}$ is the unit vector representing the direction of translation, and $\mathbf{Z}_i = [\tilde{Z}_i' \quad \tilde{Z}_i]^T$.

In addition to an algorithm for recovering $\hat{\mathbf{T}}$, R, and $\mathbf{Z}_i$ from two frames, Weng *et al.* have also analyzed the errors in $\hat{\mathbf{T}}$ and R using classical perturbation theory. Given knowledge about the noise statistics of the feature point measurements, one can estimate the covariance matrices of the errors in $\hat{\mathbf{T}}$ and R. Unfortunately, an error analysis for the relative depths is not available in [91]. An error analysis for the relative depth scale factor is provided in Section 4.3.1.

4.2 Recursive Update Formulae

Let us begin with (3.3), which is restated here as (4.9),

$$\mathbf{s}_i(t) \;=\; R(t)\mathbf{s}_{oi} + \mathbf{s}_{\mathrm{R}}(t) \tag{4.9}$$

where $\mathbf{s}_{\mathrm{i}}(t)$ is the vector of 3-D coordinates of the i-th feature point on the object expressed in camera-centred coordinates, $\mathbf{s}_{\mathrm{oi}}$ is the same point expressed in object-centred coordinates, $R(t)$ is the rotation matrix that aligns the object-centred coordinates system with the camera-centred coordinates system, and $\mathbf{s}_{\mathrm{R}}(t)$ is the 3-D coordinate vector of the origin of the OCCS. Rearranging (4.9) and evaluating at t_k and t_{k+1} we get

$$\mathbf{s}_{\mathrm{oi}} \;=\; R^{-1}(t_k)\left[\mathbf{s}_{\mathrm{i}}(t_{\mathrm{k}}) - \mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}})\right] \tag{4.10}$$

and

$$\mathbf{s}_{\mathrm{oi}} \;=\; R^{-1}(t_{k+1})\left[\mathbf{s}_{\mathrm{i}}(t_{\mathrm{k+1}}) - \mathbf{s}_{\mathrm{R}}(t_{\mathrm{k+1}})\right]. \tag{4.11}$$

Equating (4.10) and (4.11) and rearranging we get

$$\mathbf{s}_{\mathrm{i}}(t_{\mathrm{k+1}}) \;=\; R(t_{k+1})R^{-1}(t_k)\mathbf{s}_{\mathrm{i}}(t_{\mathrm{k}}) - R(t_{k+1})R^{-1}(t_k)\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}}) + \mathbf{s}_{\mathrm{R}}(t_{\mathrm{k+1}}). \tag{4.12}$$

Now define the new interframe quantities

$$R_{k,k+1} \;\triangleq\; R(t_{k+1})R^{-1}(t_k) \tag{4.13}$$

and

$$\mathbf{T}_{k,k+1} \;\triangleq\; \mathbf{s}_{\mathrm{R}}(t_{\mathrm{k+1}}) - R_{k,k+1}\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}}), \tag{4.14}$$

so that (4.12) can be rewritten as

$$\mathbf{s}_i(t_{k+1}) \;=\; R_{\mathrm{k,k+1}}\,\mathbf{s}_i(t_k) + \mathbf{T}_{\mathrm{k,k+1}} \tag{4.15}$$

which has the same form as (4.3). If we rewrite (4.13) as

$$R(t_{k+1}) \;=\; R_{k,k+1}R(t_k), \tag{4.16}$$

then we have a recursive update formula for $R(t_k)$ in terms of the interframe rotation matrix $R_{k,k+1}$. Similarly, we can derive a recursive update formula for $\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}})$ in terms of the interframe rotation matrix $R_{k,k+1}$ and the translational vector $\mathbf{T}_{k,k+1}$ by rewriting (4.14) as

$$\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k+1}}) \;=\; R_{\mathrm{k,k+1}}\,\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}}) + \mathbf{T}_{\mathrm{k,k+1}}\,. \tag{4.17}$$

However, we must keep in mind that due to the "scale factor ambiguity" the WAH algorithm can only recover the direction of translation $\hat{\mathbf{T}}_{\mathrm{k,k+1}}$. So we divide (4.17) by $\|\mathbf{T}_{\mathrm{k,k+1}}\|$ and get

$$\frac{\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k+1}})}{\|\mathbf{T}_{\mathrm{k,k+1}}\|} \;=\; R_{\mathrm{k,k+1}}\,\frac{\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}})}{\|\mathbf{T}_{\mathrm{k,k+1}}\|} + \hat{\mathbf{T}}_{\mathrm{k,k+1}}\,. \tag{4.18}$$

It is important to realize that the recursive update formula for $R(t)$ is independent of $\mathbf{s}_{\mathrm{R}}(t)$ while the recursive update formula for $\mathbf{s}_{\mathrm{R}}(t)$ is independent of $R(t)$. In other words, we have separated the estimation of the translational motion parameters from the estimation of the rotational parameters. This *decoupling* of translational and rotational components suggests that translational and rotational motion parameters can be estimated in parallel, at least in principle. Furthermore, many investigators believe that if the translational motion parameters are available, then the determination of the rotational motion parameters can be greatly facilitated, and vice versa. In fact, a number of algorithms are based on this idea [35, 91].

Let us first examine the translation subsystem.

4.2.1 Translation Subsystem

To define the translation subsystem, we need to first identify the state variables (which will form the state vector), then derive the measurement equation and the plant equation. The state vector for the translation subsystem must contain quantities useful for

describing the translational motion of the object, *i.e.,* the position of the origin of the OCCS (with respect to the CCCS) and its translational velocity. Looking at (4.18) one might choose the following as the state vector for the translation subsystem:

$$\begin{bmatrix} X_{\mathrm{R}}(t_k)/\|\mathbf{T}_{\mathrm{k,k+1}}\| \\ Y_{\mathrm{R}}(t_k)/\|\mathbf{T}_{\mathrm{k,k+1}}\| \\ Z_{\mathrm{R}}(t_k)/\|\mathbf{T}_{\mathrm{k,k+1}}\| \\ \dot{X}_{\mathrm{R}}/\|\mathbf{T}_{\mathrm{k,k+1}}\| \\ \dot{Y}_{\mathrm{R}}/\|\mathbf{T}_{\mathrm{k,k+1}}\| \\ \dot{Z}_{\mathrm{R}}/\|\mathbf{T}_{\mathrm{k,k+1}}\| \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{bmatrix}. \tag{4.19}$$

However, due to the time-varying normalization factor $\|\mathbf{T}_{\mathrm{k,k+1}}\|$ the above choice is not satisfactory. A significant improvement can be made by simply multiplying (4.18) by

$$\mathcal{K}_{\mathrm{k}} \triangleq \frac{\|\mathbf{T}_{\mathrm{k,k+1}}\|}{\|\mathbf{T}_{\mathrm{0,1}}\|}, \tag{4.20}$$

resulting in

$$\frac{\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k+1}})}{\|\mathbf{T}_{\mathrm{0,1}}\|} = R_{\mathrm{k,k+1}} \frac{\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}})}{\|\mathbf{T}_{\mathrm{0,1}}\|} + \hat{\mathbf{T}}_{\mathrm{k,k+1}} \mathcal{K}_{\mathrm{k}}\,. \tag{4.21}$$

Note that $\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}})$ and $\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k+1}})$ are now normalized by $\|\mathbf{T}_{\mathrm{0,1}}\|$, a quantity that is not dependent on time. The improved state vector for the translation subsystem is

$$\mathbf{x}^{trans}(t) = \begin{bmatrix} X_{\mathrm{R}}(t)/\|\mathbf{T}_{\mathrm{0,1}}\| \\ Y_{\mathrm{R}}(t)/\|\mathbf{T}_{\mathrm{0,1}}\| \\ Z_{\mathrm{R}}(t)/\|\mathbf{T}_{\mathrm{0,1}}\| \\ \dot{X}_{\mathrm{R}}/\|\mathbf{T}_{\mathrm{0,1}}\| \\ \dot{Y}_{\mathrm{R}}/\|\mathbf{T}_{\mathrm{0,1}}\| \\ \dot{Z}_{\mathrm{R}}/\|\mathbf{T}_{\mathrm{0,1}}\| \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{bmatrix}. \tag{4.22}$$

With this model, the true 3-D position and velocity of the object are known if $\|\mathbf{T}_{\mathrm{0,1}}\|$ can be computed or measured, for instance, by a stereo vision system or a laser range finder during the initialization phase. On the other hand, if one adopts a model similar to the one described in [15], then a different normalization factor has to be

computed or measured for each image frame, which is undoubtedly a much more difficult task.

Now that we have defined the state vector, the next step is to derive the measurement equation, which describes the relationship between the state variables and the measurements. Because we can observe states s_1–s_3 directly using (4.21), we have a very simple and *linear* measurement equation:

$$\mathbf{y}_k^{trans} = H\,\mathbf{x}_k^{trans} + \mathbf{v}_k^{trans} \tag{4.23}$$

where $\mathbf{y}_k^{trans}$ is the measurement vector, $\mathbf{v}_k^{trans}$ is the measurement noise vector, and

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \tag{4.24}$$

As in Chapter 3, we will assume that the object is rigid and moves at constant translational and rotational velocities. The position of the object (or, more precisely, the normalized coordinates of the origin of the OCCS) is determined by

$$\frac{X_{\mathrm{R}}(t_{k+1})}{\|\mathbf{T}_{0,1}\|} = \frac{X_{\mathrm{R}}(t_k)}{\|\mathbf{T}_{0,1}\|} + \frac{\dot{X}_{\mathrm{R}}}{\|\mathbf{T}_{0,1}\|}\Delta t \tag{4.25}$$

$$\frac{Y_{\mathrm{R}}(t_{k+1})}{\|\mathbf{T}_{0,1}\|} = \frac{Y_{\mathrm{R}}(t_k)}{\|\mathbf{T}_{0,1}\|} + \frac{\dot{Y}_{\mathrm{R}}}{\|\mathbf{T}_{0,1}\|}\Delta t \tag{4.26}$$

$$\frac{Z_{\mathrm{R}}(t_{k+1})}{\|\mathbf{T}_{0,1}\|} = \frac{Z_{\mathrm{R}}(t_k)}{\|\mathbf{T}_{0,1}\|} + \frac{\dot{Z}_{\mathrm{R}}}{\|\mathbf{T}_{0,1}\|}\Delta t \tag{4.27}$$

where $\Delta t \triangleq t_{k+1} - t_k$. Written in terms of the state variables, (4.25)–(4.27) become

$$s_1(t_{k+1}) = s_1(t_k) + s_4\Delta t \tag{4.28}$$

$$s_2(t_{k+1}) = s_2(t_k) + s_5\Delta t \tag{4.29}$$

$$s_3(t_{k+1}) = s_3(t_k) + s_6\Delta t \tag{4.30}$$

which implies that the plant is linear as well:

$$\mathbf{x}_{k+1}^{trans} = F\,\mathbf{x}_{k}^{trans} + \mathbf{w}_{k}^{trans} \tag{4.31}$$

where $\mathbf{w}_{k}^{trans}$ is the process noise vector, and

$$F = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.32}$$

Due to the fact that both the plant and measurement equations are linear, a simple linear Kalman filter can be used to estimate the translational motion parameters recursively.

Propagating the Scale Factor

At this point we have completely defined the translation subsystem except for one quantity, the factor $\mathcal{K}_k$. First of all, (4.20) can be rewritten as a product of ratios:

$$\mathcal{K}_k = \left(\frac{\|\mathbf{T}_{k,k+1}\|}{\|\mathbf{T}_{k\text{-}1,k}\|}\right)\left(\frac{\|\mathbf{T}_{k\text{-}1,k}\|}{\|\mathbf{T}_{k\text{-}2,k\text{-}1}\|}\right)\cdots\left(\frac{\|\mathbf{T}_{1,2}\|}{\|\mathbf{T}_{0,1}\|}\right). \tag{4.33}$$

Suppose that Z'_{i-1} denotes the absolute depth of a point at time t_i estimated by the WAH algorithm using frames $i-1$ and i. Furthermore, suppose that Z_i denotes the absolute depth of a point at time t_i estimated by the WAH algorithm using frames i and $i+1$. To simplify the notation, we have chosen to omit the subscripts that indicate the feature point with which the absolute/relative depths are associated. Now each individual ratio can be expressed as

$$\frac{\|\mathbf{T}_{i,i+1}\|}{\|\mathbf{T}_{i\text{-}1,i}\|} = \left(\frac{Z'_{i\text{-}1,i}}{\|\mathbf{T}_{i\text{-}1,i}\|}\right)\left(\frac{\|\mathbf{T}_{i,i+1}\|}{Z_{i,i+1}}\right) = \frac{\tilde{Z}'_{i\text{-}1,i}}{\tilde{Z}_{i,i+1}}. \tag{4.34}$$

It is crucial to recognize that $Z'_{i\text{-}1,i}$ and $Z_{i,i+1}$ actually represent the absolute depths of the *same* 3-D point (see Figure 4.2). Under noise-free conditions, $Z'_{i\text{-}1,i} = Z_{i,i+1}$; therefore, they cancel each other out in (4.34). Substituting (4.34) into (4.33), we get

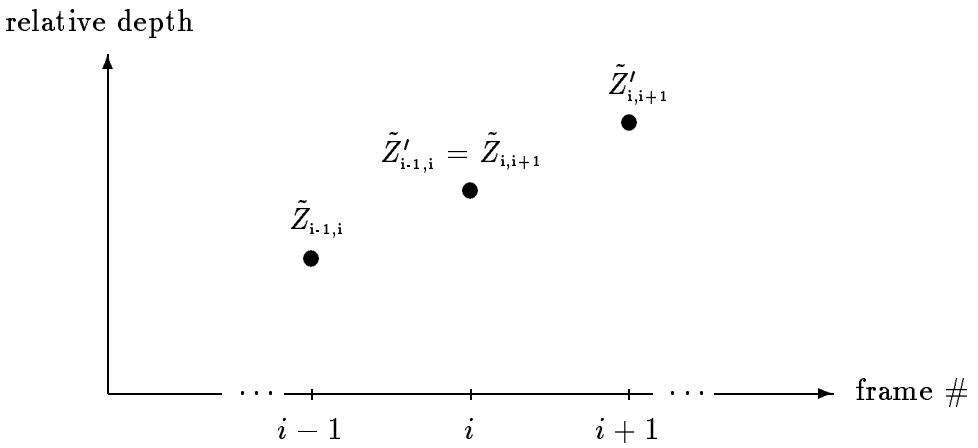


Figure 4.2: Relationship between Z_i's and $\mathcal{K}_k$

$$\mathcal{K}_{\mathrm{k}} = \left(\frac{\tilde{Z}'_{\mathrm{k\text{-}1,k}}}{\tilde{Z}_{\mathrm{k,k+1}}}\right)\cdots\left(\frac{\tilde{Z}'_{0,1}}{\tilde{Z}_{1,2}}\right) = \prod_{i=1}^{k}\frac{\tilde{Z}'_{\mathrm{i\text{-}1,i}}}{\tilde{Z}_{\mathrm{i,i+1}}}. \tag{4.35}$$

If we define

$$c_i \triangleq \frac{\|\mathbf{T}_{\mathrm{i,i+1}}\|}{\|\mathbf{T}_{\mathrm{i\text{-}1,i}}\|}, \tag{4.36}$$

then we can rewrite (4.35) using a more compact notation:

$$\mathcal{K}_{\mathrm{k}} = \prod_{i=1}^{k} c_i, \tag{4.37}$$

or, alternatively, in the form of a recursive relation:

$$\mathcal{K}_{\mathrm{k}} = \left(\frac{\|\mathbf{T}_{\mathrm{k,k+1}}\|}{\|\mathbf{T}_{\mathrm{k\text{-}1,k}}\|}\right)\mathcal{K}_{\mathrm{k\text{-}1}} = c_k\mathcal{K}_{\mathrm{k\text{-}1}}\,. \tag{4.38}$$

Three frames are needed to compute the ratio in (4.34); therefore, as long as one feature point can be tracked over three frames, we can "propagate" the unknown scale factor $\|\mathbf{T}_{\mathrm{k,k+1}}\|$ [23, 90]. One obvious advantage of this approach is that feature points no longer have to be tracked over an entire image sequence. A minimum of 8 points must be tracked over two frames and one of these 8 points must be be tracked for an additional frame.

4.2.2 Rotation Subsystem

To define the rotation subsystem, again, we need to first identify the state variables, then derive the measurement equation and the plant equation. The state vector for

the rotation subsystem must contain quantities useful for describing the rotational motion of the object. A unit quaternion is used to represent the orientation of the object (or, more precisely, the orientation of the OCCS with respect to the CCCS) and a 3-D vector is used to represent the rotational velocity:

$$\mathbf{x}^{rot}(t) = \begin{bmatrix} q_1(t) \\ q_2(t) \\ q_3(t) \\ q_4(t) \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{bmatrix}. \tag{4.39}$$

Having defined the state vector, we proceed to derive the measurement equation. A by-product of the WAH algorithm is the interframe quaternion $\boldsymbol{\eta}_{\mathrm{k,k+1}}$ [91]. In fact, the interframe rotation matrix $R_{\mathrm{k,k+1}}$ is formed by the components of $\boldsymbol{\eta}_{\mathrm{k,k+1}}$. Rewriting (4.16) using quaternions, we have

$$\mathbf{q}(t_{k+1}) \;=\; \boldsymbol{\eta}_{\mathrm{k,k+1}} \circ \mathbf{q}(t_k) \tag{4.40}$$

where $\circ$ denotes quaternion multiplication [39]. Using (4.40) we can observe states s_1–s_4 directly; thus, as in the translation subsystem, we have a *linear* measurement equation:

$$\mathbf{y}_{\mathrm{k}}^{rot} \;=\; H\,\mathbf{x}_{\mathrm{k}}^{rot} + \mathbf{v}_{\mathrm{k}}^{rot} \tag{4.41}$$

where $\mathbf{y}_{\mathrm{k}}^{rot}$ is the measurement vector, $\mathbf{v}_{\mathrm{k}}^{rot}$ is the measurement noise vector, and

$$H \;=\; \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}. \tag{4.42}$$

To derive the plant equation, recall that the unit quaternion is propagated forward in time using (3.7). Unfortunately, this leads to a non-linear plant equation:

$$\mathbf{x}_{\mathrm{k+1}}^{rot} \;=\; \mathbf{f}(\mathbf{x}_{\mathrm{k}}^{rot}) + \mathbf{w}_{\mathrm{k}}^{rot} \tag{4.43}$$

where

$$\mathbf{f}(\mathbf{x}_k^{rot}) = \begin{bmatrix} \exp[\Omega\,\Delta t] & 0_{4\times 3} \\ 0_{3\times 4} & I_{3\times 3} \end{bmatrix} \mathbf{x}_k^{rot}, \tag{4.44}$$

$I_{3\times 3}$ is a 3×3 identity matrix, $0_{4\times 3}$ is a 4×3 zero matrix, and $0_{3\times 4}$ is a 3×4 zero matrix. Because of the non-linearity in the plant equation, linearization of $\mathbf{f}(\cdot)$ is necessary and a linear Kalman filter cannot be used. An approximate non-linear filter, such as the extended Kalman filter, may be used instead.

To summarize this section, let us review the steps required in the proposed algorithm. Image plane position measurements (or "raw" measurements) are used as input to the WAH algorithm, producing the interframe motion and structure parameters, $\mathbf{T}_{k,k+1}$, $R_{k,k+1}$, and $\mathbf{Z}_{k,k+1}$. Using (4.16) and (4.21), these interframe parameters are combined with the state estimates from the previous frame to produce measurements for the Kalman filter. This arrangement has a number of potential advantages. First of all, feature points no longer have to be tracked over the entire image sequence. Instead, eight or more feature correspondences are required for every pair of consecutive frames and at least one of these point correspondences has to be maintained for three consecutive frames. Secondly, rather than having a time-varying normalization factor, a constant normalization factor is used in the translation subsystem state vector. This is an improvement over the state vector proposed by Broida *et al.* [15]. Moreover, as a result of the system decoupling, the estimation of translational motion can be carried out in parallel with the estimation of rotational motion. Furthermore, a simple linear Kalman filter is sufficient for the translation subsystem, although a more complex non-linear filter such as the EKF is still required by the rotation subsystem. Finally, the state of the overall system is represented by 13 state variables (in two state vectors) regardless of the number of feature points used by the WAH algorithm. In contrast, the BCC algorithm discussed in Chapter 3 has a state vector whose dimension is proportional to the number of feature points.

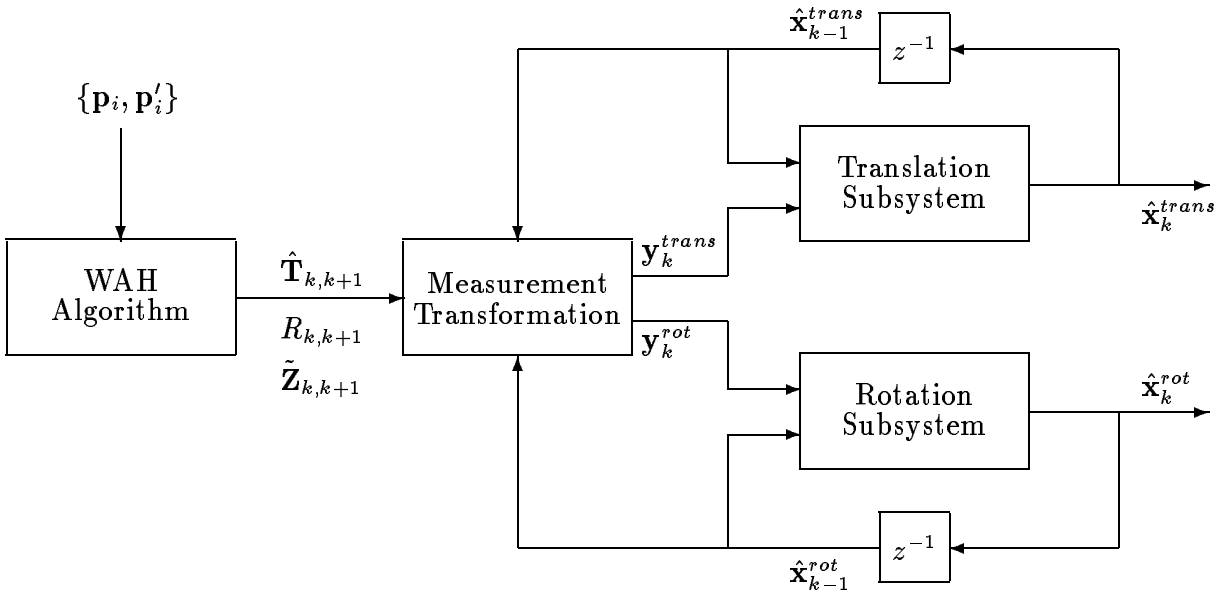


Figure 4.3: Measurement transformation: overall system diagram

4.3 Adaptive Estimation of the Measurement Error Covariance Matrix

Recall that the measurement noise covariance matrix in Chapter 3 is implemented as a constant diagonal matrix. Such a decision is justifiable since the correlation among image plane position measurements is negligible. Moreover, for a given image acquisition system and a specific feature detection/correspondence algorithm, the noise statistics are fixed (and often known). However, when measurement transformation is involved, matters become more complicated. First of all, feature correspondences over two frames are transformed by the WAH algorithm. Then they are combined with the previous state estimates to form the actual measurements used by the Kalman filter. Because errors introduced by the WAH algorithm are dependent on the motion of the object, which is not known a priori, an adaptive scheme for estimating the measurement noise covariance matrix is necessary. In this section, we shall develop such a scheme.

To be consistent with the notation in [91], we shall use Δ to denote matrix noise terms and δ to denote vector or scalar noise terms. For example, if R represents the

true interframe rotation matrix and $R(\epsilon)$ represents the noise-corrupted version of R, then

$$\Delta_R = R(\epsilon) - R. \tag{4.45}$$

Associated with each noise matrix consisting of column vectors $\mathbf{R}_1, \ldots, \mathbf{R}_n$

$$\Delta_R = [\mathbf{R}_1 \quad \mathbf{R}_2 \quad \cdots \quad \mathbf{R}_n] \tag{4.46}$$

is a noise vector formed by concatenating the columns of the noise matrix into a single column vector, *i.e.,*

$$\delta_R = \begin{bmatrix} \mathbf{R}_1(t) \\ \mathbf{R}_2(t) \\ \vdots \\ \mathbf{R}_n(t) \end{bmatrix}. \tag{4.47}$$

Similarly, if $\hat{\mathbf{T}}$ represents the true translation direction and $\hat{\mathbf{T}}(\epsilon)$ represents the noise-corrupted version of $\hat{\mathbf{T}}$, then

$$\delta_{\hat{\mathbf{T}}} = \hat{\mathbf{T}}(\epsilon) - \hat{\mathbf{T}}. \tag{4.48}$$

For scalar quantities, such as the relative depths, we have

$$\delta_{\tilde{Z}} = \tilde{Z}(\epsilon) - \tilde{Z}. \tag{4.49}$$

Finally, we shall use Γ to denote error covariance matrices. For example, $\Gamma_R = E\{\delta_R \delta_R^T\}$, where $E\{\cdot\}$ denotes the expectation operator.

4.3.1 Translation Subsystem

For simplicity, let us assume that an expression for the error covariance matrix of $\mathbf{Z}_i$ is available and that the correlations between $\mathbf{Z}_i$ and $\hat{\mathbf{T}}$ and between $\mathbf{Z}_i$ and R are known.

Error Estimation for c_k

We will begin by deriving the variance for the perturbation in the individual ratios. Combining (4.34) and 4.36), we get

$$c_i = \frac{\tilde{Z}'_{\text{i-1,i}}}{\tilde{Z}_{\text{i,i+1}}} \tag{4.50}$$

and in the presence of noise, (4.50) becomes

$$c_i + \delta_{c_i} = \frac{\tilde{Z}'_{\text{i-1,i}} + \delta_{\tilde{Z}'_{\text{i-1,i}}}}{\tilde{Z}_{\text{i,i+1}} + \delta_{\tilde{Z}_{\text{i,i+1}}}}. \tag{4.51}$$

Subtracting (4.50) from (4.51) and using a similar argument as found in [88], we find that the noise term δ_{c_i} is

$$\delta_{c_i} = \frac{\delta_{\tilde{Z}'_{\text{i-1,i}}} - \delta_{\tilde{Z}_{\text{i,i+1}}} \left(\frac{\tilde{Z}'_{\text{i-1,i}}}{\tilde{Z}_{\text{i,i+1}}} \right)}{\tilde{Z}_{\text{i,i+1}} + \delta_{\tilde{Z}_{\text{i,i+1}}}}. \tag{4.52}$$

If we assume that $\tilde{Z}_{\text{i,i+1}} \gg \delta_{\tilde{Z}_{\text{i,i+1}}}$, then we can approximate (4.52) as

$$\delta_{c_i} = \frac{\delta_{\tilde{Z}'_{\text{i-1,i}}} - \delta_{\tilde{Z}_{\text{i,i+1}}} \left(\frac{\tilde{Z}'_{\text{i-1,i}}}{\tilde{Z}_{\text{i,i+1}}} \right)}{\tilde{Z}_{\text{i,i+1}}}. \tag{4.53}$$

Squaring (4.53) and taking the expectation, we get

$$\sigma^2_{c_i} = \frac{1}{\tilde{Z}^2_{\text{i,i+1}}} E \left\{ \delta^2_{\tilde{Z}'_{\text{i-1,i}}} + \delta^2_{\tilde{Z}_{\text{i,i+1}}} \left(\frac{\tilde{Z}'_{\text{i-1,i}}}{\tilde{Z}_{\text{i,i+1}}} \right)^2 - 2\delta_{\tilde{Z}'_{\text{i-1,i}}} \delta_{\tilde{Z}_{\text{i,i+1}}} \left(\frac{\tilde{Z}'_{\text{i-1,i}}}{\tilde{Z}_{\text{i,i+1}}} \right) \right\}. \tag{4.54}$$

Error Estimation for $\mathcal{K}_k$

In the presence of noise, (4.37) becomes

$$\mathcal{K}_k + \delta_{\mathcal{K}_k} = \prod_{i=1}^{k} (c_i + \delta_{c_i}). \tag{4.55}$$

Subtracting (4.37) from (4.55) and retaining only the first-order terms results in

$$\delta_{\mathcal{K}_k} = \sum_{i=1}^{k} \delta_{c_i} \left(\prod_{j \neq i}^{k} c_j \right) \tag{4.56}$$

or, as a recursive formula,

$$\delta_{\mathcal{K}_k} \;=\; \delta_{\mathcal{K}_{k-1}} c_k + \delta_{c_k} \prod_{i=1}^{k-1} c_i. \tag{4.57}$$

Squaring (4.57), taking the expectation, and assuming that $E\{\delta_{\mathcal{K}_{k-1}} \delta_{c_k}\} = 0$, we obtain the variance of $\delta_{\mathcal{K}_k}$ as

$$\sigma^2_{\mathcal{K}_k} \;=\; E\{\delta^2_{\mathcal{K}_k}\} \;=\; \sigma^2_{\mathcal{K}_{k-1}} c_k^2 + \sigma^2_{c_k} \left(\prod_{i=1}^{k-1} c_i \right)^2 . \tag{4.58}$$

Error Estimation for $\mathbf{y}_k^{trans}$

To obtain an expression for the noise covariance matrix of the measurements, let us first rewrite (4.21) as

$$\frac{\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}})}{\|\mathbf{T}_{0,1}\|} \;=\; R_{\mathrm{k\text{-}1,k}} \, \frac{\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k\text{-}1}})}{\|\mathbf{T}_{0,1}\|} + \hat{\mathbf{T}}_{\mathrm{k\text{-}1,k}} \, \mathcal{K}_{\mathrm{k\text{-}1}} \, . \tag{4.59}$$

If we treat $\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k}})/\|\mathbf{T}_{0,1}\|$ as the current measurement and $\mathbf{s}_{\mathrm{R}}(t_{\mathrm{k\text{-}1}})/\|\mathbf{T}_{0,1}\|$ as the previous state estimate, then we can rewrite (4.59) as

$$\mathbf{y}_{\mathrm{k}}^{trans} \;=\; R_{\mathrm{k\text{-}1,k}} \, H \hat{\mathbf{x}}_{\mathrm{k\text{-}1}}^{trans} + \hat{\mathbf{T}}_{\mathrm{k\text{-}1,k}} \, \mathcal{K}_{\mathrm{k\text{-}1}} \, . \tag{4.60}$$

The matrix H, which is defined in (4.24), is used here to ensure that only the first three components of $\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}$ are used. Taking noise into consideration, (4.60) becomes

$$\begin{aligned} \mathbf{y}_{\mathrm{k}}^{trans} + \delta_{\mathbf{y}_{\mathrm{k}}} \;=\; & (R_{\mathrm{k\text{-}1,k}} + \Delta_{R_{\mathrm{k\text{-}1,k}}}) H (\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}^{trans} + \delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}}) \\ & + (\hat{\mathbf{T}}_{\mathrm{k\text{-}1,k}} + \delta_{\hat{\mathbf{T}}_{\mathrm{k\text{-}1,k}}}) (\mathcal{K}_{\mathrm{k\text{-}1}} + \delta_{\mathcal{K}_{\mathrm{k\text{-}1}}}). \end{aligned} \tag{4.61}$$

Subtracting (4.60) from (4.61) and applying first-order approximation, we get

$$\delta_{\mathbf{y}_{\mathrm{k}}} \;=\; R_{\mathrm{k\text{-}1,k}} \, H \delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}} + \Delta_{R_{\mathrm{k\text{-}1,k}}} \, H \hat{\mathbf{x}}_{\mathrm{k\text{-}1}}^{trans} + \mathcal{K}_{\mathrm{k\text{-}1}} \, \delta_{\hat{\mathbf{T}}_{\mathrm{k\text{-}1,k}}} + \delta_{\mathcal{K}_{\mathrm{k\text{-}1}}} \, \hat{\mathbf{T}}_{\mathrm{k\text{-}1,k}} \tag{4.62}$$

or, equivalently,

$$\delta_{\mathbf{y}_{\mathrm{k}}} \;=\; R_{\mathrm{k\text{-}1,k}} \, H \delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}} + M \delta_{R_{\mathrm{k\text{-}1,k}}} + \mathcal{K}_{\mathrm{k\text{-}1}} \, \delta_{\hat{\mathbf{T}}_{\mathrm{k\text{-}1,k}}} + \delta_{\mathcal{K}_{\mathrm{k\text{-}1}}} \, \hat{\mathbf{T}}_{\mathrm{k\text{-}1,k}} \tag{4.63}$$

where

$$M \;\triangleq\; \begin{bmatrix} s_1 & 0 & 0 & s_2 & 0 & 0 & s_3 & 0 & 0 \\ 0 & s_1 & 0 & 0 & s_2 & 0 & 0 & s_3 & 0 \\ 0 & 0 & s_1 & 0 & 0 & s_2 & 0 & 0 & s_3 \end{bmatrix} \tag{4.64}$$

and s_1–s_3 are the first three components of $\hat{\mathbf{x}}_{\text{k-1}}$. It is reasonable to assume that there is no correlation between the errors in $\hat{\mathbf{x}}_{\text{k-1}}$ and the errors in $\hat{\mathbf{T}}_{\text{k-1,k}}$, $R_{\text{k-1,k}}$, and $\mathcal{K}_{\text{k-1}}$, *i.e.*,

$$E\{\delta_{\hat{\mathbf{x}}_{k-1}}\delta^T_{\hat{\mathbf{T}}_{\text{k-1,k}}}\} = E\{\delta_{\hat{\mathbf{x}}_{k-1}}\delta^T_{R_{\text{k-1,k}}}\} = E\{\delta_{\hat{\mathbf{x}}_{k-1}}\delta^T_{\mathcal{K}_{\text{k-1}}}\} = 0. \tag{4.65}$$

Squaring (4.63) and taking the expectation, we arrive at the measurement noise covariance matrix

$$\begin{aligned}\Gamma^{trans}_{\mathbf{y}_\text{k}} &= R_{\text{k-1,k}}\, H\, \Gamma_{\hat{\mathbf{x}}_{\text{k-1}}} H^T R^T_{\text{k-1,k}} + M\, \Gamma_R\, M^T + \mathcal{K}^2_{\text{k-1}}\, \Gamma_{\hat{\mathbf{T}}} + \\ &\quad \sigma^2_{\mathcal{K}} \hat{\mathbf{T}}_{\text{k-1,k}}\, \hat{\mathbf{T}}^T_{\text{k-1,k}} + \Gamma_1 + \Gamma_2 + \Gamma_3 \end{aligned} \tag{4.66}$$

where

$$\Gamma_1 \triangleq M E\{\delta_R \delta^T_{\hat{\mathbf{T}}}\}\mathcal{K}_{\text{k-1}} + \mathcal{K}_{\text{k-1}}\, E\{\delta_{\hat{\mathbf{T}}}\delta^T_R\} M^T \tag{4.67}$$

$$\Gamma_2 \triangleq \hat{\mathbf{T}}_{\text{k-1,k}}\, E\{\delta_{\mathcal{K}}\delta^T_R\} M^T + M E\{\delta_R \delta_{\mathcal{K}}\}\hat{\mathbf{T}}^T_{\text{k-1,k}} \tag{4.68}$$

$$\Gamma_3 \triangleq \hat{\mathbf{T}}_{\text{k-1,k}}\, E\{\delta_{\mathcal{K}}\delta^T_{\hat{\mathbf{T}}}\}\mathcal{K}_{\text{k-1}} + \mathcal{K}_{\text{k-1}}\, E\{\delta^T_{\hat{\mathbf{T}}}\delta_{\mathcal{K}}\}\hat{\mathbf{T}}^T_{\text{k-1,k}}\,. \tag{4.69}$$

In passing, one should recognize that $\Gamma_{\hat{\mathbf{x}}_{k-1}}$ is equivalent to the error covariance matrix $\hat{P}(k-1|k-1)$ which is directly available from the translation subsystem's Kalman filter.

4.3.2 Rotation Subsystem

Compared to the translation subsystem, estimating the measurement noise covariance matrix for the rotation subsystem is much less involved. Let us first rewrite (4.40) in terms of vector-matrix multiplication rather than quaternion multiplication [91, 97], *i.e.*,

$$\mathbf{q}(t_k) = N_{\text{k-1,k}}\, \mathbf{q}(t_{k-1}). \tag{4.70}$$

where

$$N_{\text{k-1,k}} \triangleq \begin{bmatrix} \eta_0 & -\eta_1 & -\eta_2 & -\eta_3 \\ \eta_1 & \eta_0 & -\eta_3 & \eta_2 \\ \eta_2 & \eta_3 & \eta_0 & -\eta_1 \\ \eta_3 & -\eta_2 & \eta_1 & \eta_0 \end{bmatrix} \tag{4.71}$$

and $\eta_0 - \eta_3$ are the components of the interframe quaternion $\boldsymbol{\eta}_{\mathrm{k\text{-}1,k}}$. Treating $\mathbf{q}(t_k)$ as the current measurement and $\mathbf{q}(t_{k-1})$ as the previous state estimate, we can rewrite (4.70) as

$$\mathbf{y}_{\mathrm{k}}^{rot} = N_{\mathrm{k\text{-}1,k}}\, H \hat{\mathbf{x}}_{\mathrm{k\text{-}1}}^{rot}. \tag{4.72}$$

Premultiplying $\hat{\mathbf{x}}_{k-1}$ by H, which is defined in (4.42), is necessary because only the first four components of $\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}$ are used. Taking noise into consideration, (4.72) becomes

$$\mathbf{y}_{\mathrm{k}}^{rot} + \delta_{\mathbf{y}_{\mathrm{k}}} = (N_{\mathrm{k\text{-}1,k}} + \Delta_{N_{\mathrm{k\text{-}1,k}}}) H (\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}^{rot} + \delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}}). \tag{4.73}$$

Subtracting (4.72) from (4.73) and retaining only first-order terms, we get an expression for the noise term

$$\delta_{\mathbf{y}_{\mathrm{k}}} = N_{\mathrm{k\text{-}1,k}}\, H \delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}} + \Delta_{N_{\mathrm{k\text{-}1,k}}}\, H \hat{\mathbf{x}}_{\mathrm{k\text{-}1}}^{rot} \tag{4.74}$$

which can be rewritten as

$$\delta_{\mathbf{y}_{\mathrm{k}}} = N_{\mathrm{k\text{-}1,k}}\, H \delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}} + M \delta_{N_{\mathrm{k\text{-}1,k}}} \tag{4.75}$$

where M is a 4×16 matrix defined as

$$M \triangleq [s_1 I_4 \quad s_2 I_4 \quad s_3 I_4 \quad s_4 I_4], \tag{4.76}$$

$s_1 - s_4$ are the first four components of $\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}^{rot}$, and I_4 is a 4×4 identity matrix. Squaring (4.75) and taking the expectation, we get the measurement noise covariance matrix:

$$\Gamma_{\mathbf{y}_{\mathrm{k}}}^{rot} = NHE\left\{\delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}}\, \delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}}^T\right\} H^T N^T + ME\left\{\delta_N \delta_N^T\right\} M^T, \tag{4.77}$$

assuming that $E\{\delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}}\, \delta_N^T\} = E\{\delta_N \delta_{\hat{\mathbf{x}}_{\mathrm{k\text{-}1}}}^T\} = 0$. The matrix elements of $\Gamma_N \triangleq E\{\delta_N \delta_N^T\}$ are identical to those found in $\Gamma_{\boldsymbol{\eta}} \triangleq E\left\{\delta_{\boldsymbol{\eta}} \delta_{\boldsymbol{\eta}}^T\right\}$, which can be estimated using the method outlined in [91]. Finally, one should recognize that $\Gamma_{\hat{\mathbf{x}}_{k-1}}$ is equivalent to the error covariance matrix $\hat{P}(k-1|k-1)$ which can be obtained from the rotation subsystem's Kalman filter. Note that although a linear Kalman filter cannot be used in the rotation subsystem, the adaptive measurement noise covariance matrix estimation scheme is much simpler than the one used in the translation subsystem.

4.4 Discussions

One major weakness of the algorithm proposed in this chapter is that errors tend to accumulate in the factor $\mathcal{K}_k$, which is computed as a product of relative depths. From (4.8), we can see that $\mathbf{Z}_i$ is comupted using only two observations, $\mathbf{p}'_i$ and $\mathbf{p}_i$; therefore, it is unreasonable to expect the relative depths to be highly accurate. In fact, when the noise level in the image is high enough, the WAH algorithm can result in negative relative depths. Since we assume that the object is always in front of the camera, negative relative depths are unacceptable. These erroneous depth values can also cause $\mathcal{K}_k$ to become negative. This is problematic because $\mathcal{K}_k$ is defined as a product of translation vector magnitudes, which implies that it should always be positive. It should be pointed out that, unlike the state estimates, no mechanism is designed to improve the accuracy of $\mathcal{K}_k$ over time.

Another disadvantage of the proposed algorithm is that a fairly complicated adaptive measurement noise covariance matrix estimation procedure is required. It is important that this procedure provides accurate information; otherwise, the performance of the Kalman filter will suffer.

Finally, it should be pointed out that the recursive two-view motion estimation algorithm presented in this chapter is, in certain ways, quite similar to the algorithm proposed by Heel [37]. The algorithm can be decomposed into four main functional modules and they operate together in a recursive fashion (Figure 4.4). The exchange of information among these modules is a unique feature of Heel's algorithm.

4.5 Summary

In this chapter, we examined the concept of measurement transformation and presented the mathematical formulation of an algorithm based on this idea. The two-view motion and structure estimation algorithm proposed by Weng *et al.* [91] was used to provide the transformed measurements. A comparison between this algorithm and the one developed by Broida *et al.* was made. The relative merits and shortcomings were discussed.

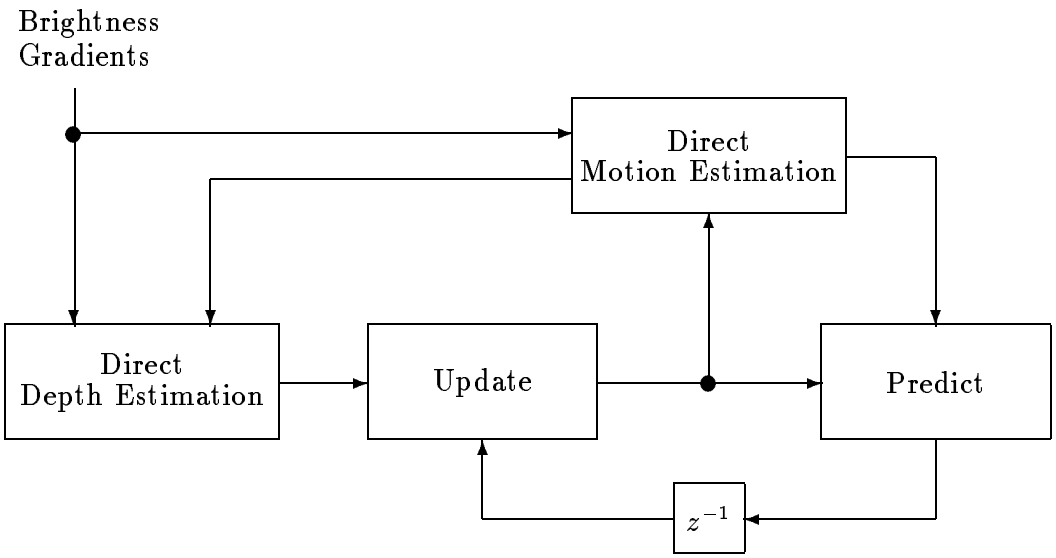


Figure 4.4: Block diagram of method proposed by Heel

Chapter 5

CONCLUSIONS AND FUTURE WORK

At the beginning of this thesis we discussed several recent approaches to estimating motion and structure from sequences of monocular images, with the emphasis on two distinct categories: the optical flow techniques and the feature-based methods. The optical flow approach involves computing the field of instantaneous velocities of brightness patterns in the 2-D image plane. The feature-based approach, on the other hand, is based on extracting a set of relatively sparse 2-D features and establishing inter-frame correspondence among them.

The first and main contribution of this thesis is the development of a hybrid 3-D trajectory and structure estimation scheme that is built upon the basic models and formulation of Broida *et al.* [15]. Broida *et al.* use the positions of a set of tracked feature points as input to an iterated extended Kalman filter (an approximate non-linear filter) which in turns recursively estimates the 3-D motion and structure of a rigid object for an arbitrarily long image sequence. We have developed in this thesis a hybrid algorithm that incorporates optical flow measurements into the basic recursive framework proposed by Broida *et al.* . Based on our simulation results, we are able to conclude that addition of optical flow measurements results in the reduction of large initial estimation errors, and significant improvements in the accuracy of the velocity state estimates and the tracking of abrupt object maneuvers. Improvements can be observed as long as the relative error in velocity measurements is less than about 100%, which is well within reach of existing methods for computing optical flow, although most of them are computationally intensive and are not suitable for

real-time implementation. We have also carried out a comparative study of three different filters for the purpose of 3-D trajectory estimation. Despite the common belief that the extended Kalman filter is inferior to the iterated extended Kalman filter or the iterated linear filter-smoother, our Monte Carlo simulation results suggest otherwise. Taking into consideration both the accuracy of the state estimates and the computational complexity, there is no concrete evidence to support that the claim (previously made by a number of investigators) that the two "more sophisticated" filters can offer better overall performance than the EKF.

The second contribution of this thesis is the mathematical formulation of an approach designed to address a number of problems inherent in the recursive framework proposed by Broida *et al.* . This approach, which is based on the concept of measurement transformation, eliminates the need to track the same features for a large number of frames and allows parameters of interest to be observed more directly. Moreover, the state vector of this proposed recursive motion estimation system has a fixed number of state variables. In other words, the size of the state vector is independent of the number of features being tracked. Furthermore, the estimation of the translational motion parameters is decoupled from the estimation of the rotational motion parameters, allowing an efficient implementation of the system. The most noticeable drawback is that a reliable adaptive measurement noise estimation algorithm is crucial to the performance of this algorithm. Also, it is confirmed by a preliminary implementation that the propagation of the unknown scale factor tends to accumulate errors.

Suggestions for Future Research

All of the experiments described in this thesis were performed using simulated feature points and optical flow measurements, with artificial noise added. While these simulations are useful in obtaining preliminary data regarding the usefulness and robustness of the hybrid algorithm, the algorithm's true capability to handle noise can only be tested using real imagery. However, obtaining motion data with ground truth of known accuracy is not a simple task. Fortunately, Dutta *et al.* have collected a total of eight real image sequences of about 30 frames each at five different outdoor

sites using both the "move and shoot" and the "stop and shoot" methods [26]. One of these sequences has been used by Chandrashekhar and Chellappa in [18]. In addition to testing an algorithm with real image sequences, one should also strive for highly efficient implementation as most real-world applications demand real-time or near real-time response.

With real image sequences, one needs to solve several problems that are not addressed in this thesis, such as feature selection, feature extraction, feature correspondence, and optical flow computation. However, because of the complexity of many of the existing algorithms and the large amount of data involved, most general-purpose computers are simply not powerful enough. A high-speed pipelined image processor, such as the Datacube MaxVideo 200, can be used to perform most of the low-level image processing operations. To further increase the throughput one could employ an expansion board equipped with a high-speed floating-point processor, such as the Intel i860. Depending on the application, one could even take advantage of special-purpose hardware designed to perform specific computer vision tasks. However, one must consider the trade-off between generality and performance. The best approach is to develop computationally efficient algorithms. Recent examples include a robust feature matching/tracking algorithm by Richardson and Blostein [67, 68] and a method for computing optical flow in real-time by Fleet and Langley [30].

In Chapter 4 we introduced the idea of measurement transformation and presented an approach based on feature correspondences. The same principles can be applied to optical flow fields. In other words, using a recursive framework, one can transform the optical flow measurements in such a way that some or all of the system state variables can be observed directly. One algorithm that is worthy of consideration is the subspace algorithm proposed by Heeger and Jepson [35]. It can be used to recover the direction of translation and the rotational velocity. Once the motion parameters have been computed relative depths can be recovered using, for example, a Kalman-filter-based method [38, 59].

Appendix A

THE JACOBIAN MATRIX

Recall from Chapter 3 that the Jacobian matrix H_k is obtained by differentiating the measurement equations (both position and optical flow) with respect to the state variables. From (3.34), we know that H_k is a $4M \times 12+3M$ matrix. In this appendix, the individual elements of this matrix will be shown. First of all, let us restate (3.6), (3.21), (3.22), (3.23), (3.31), and (3.32) here for convenience:

$$\begin{aligned} R &= \begin{bmatrix} s_6^2 - s_7^2 - s_8^2 + s_9^2 & 2(s_6 s_7 - s_8 s_9) & 2(s_6 s_8 + s_7 s_9) \\ 2(s_6 s_7 + s_8 s_9) & -s_6^2 + s_7^2 - s_8^2 + s_9^2 & 2(s_7 s_8 - s_6 s_9) \\ 2(s_6 s_8 - s_7 s_9) & 2(s_7 s_8 + s_6 s_9) & -s_6^2 - s_7^2 + s_8^2 + s_9^2 \end{bmatrix} \\ &= \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \end{aligned} \tag{A.1}$$

$$\begin{aligned} a_i &= (s_6^2 - s_7^2 - s_8^2 + s_9^2) s_{10+3i} + 2(s_6 s_7 - s_8 s_9) s_{11+3i} + \\ &\quad 2(s_6 s_8 + s_7 s_9) s_{12+3i} \end{aligned} \tag{A.2}$$

$$\begin{aligned} b_i &= 2(s_6 s_7 + s_8 s_9) s_{10+3i} + (-s_6^2 + s_7^2 - s_8^2 + s_9^2) s_{11+3i} + \\ &\quad 2(s_7 s_8 - s_6 s_9) s_{12+3i} \end{aligned} \tag{A.3}$$

$$\begin{aligned} c_i &= 2(s_6 s_8 - s_7 s_9) s_{10+3i} + 2(s_7 s_8 + s_6 s_9) s_{11+3i} + \\ &\quad (-s_6^2 - s_7^2 + s_8^2 + s_9^2) s_{12+3i} \end{aligned} \tag{A.4}$$

$$u_i = \frac{s_3 + c_i\, s_{11} - b_i\, s_{12}}{1 + c_i} - \frac{(s_5 + b_i\, s_{10} - a_i\, s_{11})(s_1 + a_i)}{(1 + c_i)^2} \tag{A.5}$$

$$v_i = \frac{s_4 - c_i\, s_{10} + a_i\, s_{12}}{1 + c_i} - \frac{(s_5 + b_i\, s_{10} - a_i\, s_{11})(s_2 + b_i)}{(1 + c_i)^2} \tag{A.6}$$

$$\tag{A.7}$$

The derivatives of a_i, b_i, and c_i with respect to the quaternion states (s_6–s_9) are:

$$\frac{da_i}{ds_6} = 2(s_6 s_{10+3i} + s_7 s_{11+3i} + s_8 s_{12+3i}) \tag{A.8}$$

$$\frac{da_i}{ds_7} = 2(-s_7 s_{10+3i} + s_6 s_{11+3i} + s_9 s_{12+3i}) \tag{A.9}$$

$$\frac{da_i}{ds_8} = 2(-s_8 s_{10+3i} - s_9 s_{11+3i} + s_6 s_{12+3i}) \tag{A.10}$$

$$\frac{da_i}{ds_9} = 2(s_9 s_{10+3i} - s_8 s_{11+3i} + s_7 s_{12+3i}) \tag{A.11}$$

$$\frac{db_i}{ds_6} = 2(s_7 s_{10+3i} - s_6 s_{11+3i} - s_9 s_{12+3i}) \tag{A.12}$$

$$\frac{db_i}{ds_7} = 2(s_6 s_{10+3i} + s_7 s_{11+3i} + s_8 s_{12+3i}) \tag{A.13}$$

$$\frac{db_i}{ds_8} = 2(s_9 s_{10+3i} - s_8 s_{11+3i} + s_7 s_{12+3i}) \tag{A.14}$$

$$\frac{db_i}{ds_9} = 2(s_8 s_{10+3i} + s_9 s_{11+3i} - s_6 s_{12+3i}) \tag{A.15}$$

$$\frac{dc_i}{ds_6} = 2(s_8 s_{10+3i} + s_9 s_{11+3i} - s_6 s_{12+3i}) \tag{A.16}$$

$$\frac{dc_i}{ds_7} = 2(-s_9 s_{10+3i} + s_8 s_{11+3i} - s_7 s_{12+3i}) \tag{A.17}$$

$$\frac{dc_i}{ds_8} = 2(s_6 s_{10+3i} + s_7 s_{11+3i} + s_8 s_{12+3i}) \tag{A.18}$$

$$\frac{dc_i}{ds_9} = 2(-s_7 s_{10+3i} + s_6 s_{11+3i} + s_9 s_{12+3i}) \tag{A.19}$$

The Jacobian matrix H_k can be treated as a concatenation of two sub-matrices, one of which is associated with the position measurements while the other one is associated with the optical flow measurements. Details about the former can be found in [15] and will not be discussed here. Differentiating (A.5) with respect to the state variables, we get

$$\frac{du_i}{ds_1} = -\frac{s_5 + b_i\, s_{10} - a_i\, s_{11}}{(1 + c_i)^2} \tag{A.20}$$

$$\frac{du_i}{ds_2} = 0 \tag{A.21}$$

$$\frac{du_i}{ds_3} = \frac{1}{1+c_i} \tag{A.22}$$

$$\frac{du_i}{ds_4} = 0 \tag{A.23}$$

$$\frac{du_i}{ds_5} = -\frac{s_1+a_i}{(1+c_i)^2} \tag{A.24}$$

$$\begin{aligned}\frac{du_i}{ds_j} = {} & \frac{s_{11}\frac{dc_i}{ds_6} - s_{12}\frac{db_i}{ds_6}}{1+c_i} - \frac{s_3 + c_i s_{11} - b_i s_{12}}{(1+c_i)^2}\frac{dc_i}{ds_6} - \\ & \frac{(s_{10}\frac{db_i}{ds_6} - s_{11}\frac{da_i}{ds_6})(s_1+a_i)}{(1+c_i)^2} - \frac{da_i}{ds_6}\frac{s_5 + b_i s_{10} - a_i s_{11}}{(1+c_i)^2} + \\ & 2\frac{dc_i}{ds_6}\frac{(s_5 + b_i s_{10} - a_i s_{11})(s_1+a_i)}{(1+c_i)^3}, \qquad j = 6\ldots 9\end{aligned} \tag{A.25}$$

$$\frac{du_i}{ds_{10}} = -\frac{b_i(s_1+a_i)}{(1+c_i)^2} \tag{A.26}$$

$$\frac{du_i}{ds_{11}} = \frac{c_i}{1+c_i} + \frac{a_i(s_1+a_i)}{(1+c_i)^2} \tag{A.27}$$

$$\frac{du_i}{ds_{12}} = -\frac{b_i}{1+c_i} \tag{A.28}$$

$$\begin{aligned}\frac{du_i}{s_{10+3i}} = {} & \frac{s_{11}r_{31} - s_{12}r_{21}}{1+c_i} - \frac{s_3 - c_i s_{11} - b_i s_{12}}{(1+c_i)^2}r_{31} - \\ & \frac{(s_{10}r_{21} - s_{11}r_{11})(s_1+a_i)}{(1+c_i)^2} - \frac{s_5 + b_i s_{10} - a_i s_{11}}{(1+c_i)^2}r_{11} + \\ & 2\frac{(s_5 + b_i s_{10} - a_i s_{11})(s_1+a_i)}{(1+c_i)^3}r_{31}\end{aligned} \tag{A.29}$$

$$\begin{aligned}\frac{du_i}{s_{11+3i}} = {} & \frac{s_{11}r_{32} - s_{12}r_{22}}{1+c_i} - \frac{s_3 - c_i s_{11} - b_i s_{12}}{(1+c_i)^2}r_{32} - \\ & \frac{(s_{10}r_{22} - s_{11}r_{12})(s_1+a_i)}{(1+c_i)^2} - \frac{s_5 + b_i s_{10} - a_i s_{11}}{(1+c_i)^2}r_{12} + \\ & 2\frac{(s_5 + b_i s_{10} - a_i s_{11})(s_1+a_i)}{(1+c_i)^3}r_{32}\end{aligned} \tag{A.30}$$

$$\begin{aligned}\frac{du_i}{s_{12+3i}} = {} & \frac{s_{11}r_{33} - s_{12}r_{23}}{1+c_i} - \frac{s_3 - c_i s_{11} - b_i s_{12}}{(1+c_i)^2}r_{33} - \\ & \frac{(s_{10}r_{23} - s_{11}r_{13})(s_1+a_i)}{(1+c_i)^2} - \frac{s_5 + b_i s_{10} - a_i s_{11}}{(1+c_i)^2}r_{13} + \\ & 2\frac{(s_5 + b_i s_{10} - a_i s_{11})(s_1+a_i)}{(1+c_i)^3}r_{33}\end{aligned} \tag{A.31}$$

Similarly, if we differentiate (A.6) with respect to the state variables, we get

$$\frac{dv_i}{ds_1} = 0 \tag{A.32}$$

$$\frac{dv_i}{ds_2} = -\frac{s_5 + b_i\, s_{10} - a_i\, s_{11}}{(1+c_i)^2} \tag{A.33}$$

$$\frac{dv_i}{ds_3} = 0 \tag{A.34}$$

$$\frac{dv_i}{ds_4} = \frac{1}{1+c_i} \tag{A.35}$$

$$\frac{dv_i}{ds_5} = -\frac{s_2 + b_i}{(1+c_i)^2} \tag{A.36}$$

$$\begin{aligned}\frac{dv_i}{ds_j} = {} & \frac{-s_{10}\frac{dc_i}{ds_6} + s_{12}\frac{da_i}{ds_6}}{1+c_i} - \frac{s_4 - c_i s_{10} + a_i s_{12}}{(1+c_i)^2}\frac{dc_i}{ds_6} - \\ & \frac{(s_{10}\frac{db_i}{ds_6} - s_{11}\frac{da_i}{ds_6})(s_2+b_i)}{(1+c_i)^2} - \frac{db_i}{ds_6}\frac{s_5 + b_i s_{10} - a_i s_{11}}{(1+c_i)^2} + \\ & 2\frac{dc_i}{ds_6}\frac{(s_5 + b_i s_{10} - a_i s_{11})(s_2+b_i)}{(1+c_i)^3} \qquad j = 6\ldots 9\end{aligned} \tag{A.37}$$

$$\frac{dv_i}{ds_{10}} = -\frac{c_i}{1+c_i} - \frac{b_i(s_2+b_i)}{(1+c_i)^2} \tag{A.38}$$

$$\frac{dv_i}{ds_{11}} = \frac{a_i(s_2+b_i)}{(1+c_i)^2} \tag{A.39}$$

$$\frac{dv_i}{ds_{12}} = \frac{a_i}{1+c_i} \tag{A.40}$$

$$\begin{aligned}\frac{dv_i}{s_{10+3i}} = {} & \frac{-s_{10}r_{31} + s_{12}r_{11}}{1+c_i} - \frac{s_4 - c_i s_{10} - a_i s_{12}}{(1+c_i)^2} r_{31} - \\ & \frac{(s_{10}r_{21} - s_{11}r_{11})(s_2+b_i)}{(1+c_i)^2} - \frac{s_5 + b_i s_{10} - a_i s_{11}}{(1+c_i)^2} r_{21} + \\ & 2\frac{(s_5 + b_i s_{10} - a_i s_{11})(s_2+b_i)}{(1+c_i)^3} r_{31}\end{aligned} \tag{A.41}$$

$$\begin{aligned}\frac{dv_i}{s_{11+3i}} = {} & \frac{-s_{10}r_{32} + s_{12}r_{12}}{1+c_i} - \frac{s_4 - c_i s_{10} - a_i s_{12}}{(1+c_i)^2} r_{32} - \\ & \frac{(s_{10}r_{22} - s_{11}r_{12})(s_2+b_i)}{(1+c_i)^2} - \frac{s_5 + b_i s_{10} - a_i s_{11}}{(1+c_i)^2} r_{22} + \\ & 2\frac{(s_5 + b_i s_{10} - a_i s_{11})(s_2+b_i)}{(1+c_i)^3} r_{32}\end{aligned} \tag{A.42}$$

$$\begin{aligned}\frac{dv_i}{s_{12+3i}} = {} & \frac{-s_{10}r_{33} + s_{12}r_{13}}{1+c_i} - \frac{s_4 - c_i s_{10} - a_i s_{12}}{(1+c_i)^2} r_{33} - \\ & \frac{(s_{10}r_{23} - s_{11}r_{13})(s_2+b_i)}{(1+c_i)^2} - \frac{s_5 + b_i s_{10} - a_i s_{11}}{(1+c_i)^2} r_{23} + \\ & 2\frac{(s_5 + b_i s_{10} - a_i s_{11})(s_2+b_i)}{(1+c_i)^3} r_{33}\end{aligned} \tag{A.43}$$

Bibliography

[1] J. K. Aggarwal and N. Nandhakumar. On the Computation of Motion from Sequences of Images – A Review. *Proceedings of the IEEE*, 76(8):917–935, 1988.

[2] J. Aloimonos and D. Tsakiris. On the Visual Mathematics of Tracking. *Image and Vision Computing*, 9(4):235–251, 1991.

[3] J. Arnspang. Optic Acceleration. In *Proceedings of the Second International Conference on Computer Vision*, pages 364–373. Institute of Electrical and Electronic Engineers, 1988.

[4] A. Azarbayejani, B. Horowitz, and A. Pentland. Recursive Estimation of Structure and Motion using Relative Orientation Constraints. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 294–299. Institute of Electrical and Electronic Engineers, 1993.

[5] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association.* Academic Press, 1988.

[6] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of Optical Flow Techniques. Technical Report RPL-TR-9107, Department of Computing and Information Science, Queen's University, 1992.

[7] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of Optical Flow Techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[8] J. L. Barron, A. D. Jepson, and J. K. Tsotsos. The Feasibility of Motion and Structure Computations. In *Proceedings of the Second International Conference on Computer Vision*, pages 651–657. Institute of Electrical and Electronic Engineers, 1988.

[9] J. L. Barron, A. D. Jepson, and J. K. Tsotsos. The Feasibility of Motion and Structure from Noisy Time-Varying Image Velocity Information. *International Journal of Computer Vision*, 5(3):239–269, 1990.

[10] P. J. Besl. Active, Optical Range Imaging Sensors. *Machine Vision and Applications*, 1(2):127–152, 1988.

[11] S. S. Blackman. *Multiple-Target Tracking with Radar Application.* Artech House, 1986.

[12] S. D. Blostein and R. M. Chann. Hybrid Feature/Flow-Based 3-D Trajectory Estimation. In *Proceedings of Vision Interface '93*, pages 125–130, 1993.

[13] S. D. Blostein and R. M. Chann. The Use of Image Plane Velocity Measurements in Recursive 3-D Motion Estimation from a Monocular Image Sequence. In *Proceedings of the Canadian Conference on Electrical and Computer Engineering.* Canadian Society for Electrical and Computer Engineering, 1994. To appear.

[14] S. D. Blostein and T. S. Huang. Error Analysis in Stereo Determination of 3-D Point Positions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(6):752–765, 1987.

[15] T. J. Broida, S. Chandrashekhar, and R. Chellappa. Recursive 3-D Motion Estimation from a Monocular Image Sequence. *IEEE Transactions on Aerospace and Electronic Systems*, 26(4):639–656, 1990.

[16] T. J. Broida and R. Chellappa. Estimating the Kinematics and Structure of a Rigid Object from a Sequence of Monocular Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):497–513, 1991.

[17] R. G. Brown and P. Y. C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering (Second Edition).* John Wiley & Sons, 1992.

[18] S. Chandrashekhar and R. Chellappa. Passive Navigation in a Partially Known Environment. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 2–7. Institute of Electrical and Electronic Engineers, 1991.

[19] K. Chang, H. Lee, and C. Chung. Adaptively Estimating Motion Parameters for Tracking a Maneuvering Target Using Images. *International Journal of Robotics and Automation*, 4(1):43–52, 1989.

[20] B. W. Char, K. O. Geddes, G. H. Gonnet, B. L. Leong, M. B. Monagan, and S. M. Watt. *First Leaves: A Tutorial Introduction to Maple V.* Springer-Verlag, 1992.

[21] C. K. Chui and G. Chen. *Kalman Filtering with Real-Time Applications.* Springer-Verlag, 1987.

[22] I. J. Cox. A Review of Statistical Data Association Techniques for Motion Correspondence. *International Journal of Computer Vision*, 10(1):53–66, 1993.

[23] N. Cui, J. Weng, and P. Cohen. Extended Structure and Motion Analysis from Monocular Image Sequences. In *Proceedings of the Third International Conference on Computer Vision*, pages 222–229. Institute of Electrical and Electronic Engineers, 1990.

[24] E. De Micheli, V. Torre, and S. Uras. The Accuracy of the Computation of Optical Flow and of the Recovery of Motion Parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):434–447, 1993.

[25] U. R. Dhond and J. K. Aggarwal. Structure from Stereo – A Review. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6):1489–1510, 1989.

[26] R. Dutta, R. Manmatha, L. R. Williams, and E. M. Riseman. A Data Set for Quantitative Motion Analysis. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 159–164. Institute of Electrical and Electronic Engineers, 1989.

[27] A. M. Earnshaw and S. D. Blostein. Unbiased Estimation of Camera Translation Direction from Optical Flow using Linear Constraints. In *Proceedings of the Canadian Conference on Electrical and Computer Engineering*. Canadian Society for Electrical and Computer Engineering, 1994. To appear.

[28] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. The MIT Press, 1993.

[29] D. J. Fleet and A. D. Jepson. Computation of Component Image Velocity from Local Phase Information. *International Journal of Computer Vision*, 5(1):77–104, 1990.

[30] D. J. Fleet and K. Langley. Toward Real-Time Optical Flow. In *Proceedings of Vision Interface '93*, pages 116–124, 1993.

[31] B. Friedland. Analysis Strapdown Navigation Using Quaternions. *IEEE Transactions on Aerospace and Electronic Systems*, 14(5):764–768, 1978.

[32] J. Funda, R. H. Taylor, and R. P. Paul. On Homogeneous Transforms, Quaternions, and Computational Efficiency. *IEEE Transactions on Robotics and Automation*, 6(3):382–388, 1990.

[33] A. Gelb. *Applied Optimal Estimation*. M.I.T. Press, 1974.

[34] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1989.

[35] D. J. Heeger and A. D. Jepson. Subspace Methods for Recovering Rigid Motion I:Algorithm and Implementation. *International Journal of Computer Vision*, 7(2):95–117, 1992.

[36] J. Heel. Direct Dynamic Motion Vision. In *IEEE International Conference on Robotics and Automation*, pages 1142–1147. Institute of Electrical and Electronic Engineers, 1990.

[37] J. Heel. Direct Estimation of Structure and Motion from Multiple Frames. Technical Report AI Memo 1190, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1990.

[38] J. Heel. Temporal Surface Reconstruction. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 607–612. Institute of Electrical and Electronic Engineers, 1991.

[39] B. K. P. Horn. *Robot Vision.* The MIT Press, 1986.

[40] R. Hummel and V. Sundareswaran. Motion Parameter Estimation from Global Flow Field Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):459–476, 1993.

[41] S. Iu and K. Wohn. Recovery of 3-D Motion of a Single Particle. In *SPIE Vol. 1192 Intelligent Robots and Computer Vision VIII: Algorithms and Techniques*, pages 746–757. International Society for Optical Engineering, 1989.

[42] S. Iu and K. Wohn. Estimation of General Rigid Body Motion From a Long Sequence of Images. In *International Conference on Pattern Recognition*, pages 217–219. Institute of Electrical and Electronic Engineers, 1990.

[43] R. C. Jain and A. K. Jain. Report: 1988 NSF Range Image Understanding Workshop. In R. C. Jain and A. K. Jain, editors, *Analysis and Interpretation of Range Images*, pages 1–31. Springer-Verlag, 1990.

[44] A. H. Jazwinski. *Stochastic Processes and Filtering Theory.* Academic Press, 1970.

[45] A. D. Jepson and D. J. Heeger. A Fast Subspace Algorithm for Recovering Rigid Motion. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 124–131. Institute of Electrical and Electronic Engineers, 1991.

[46] A. D. Jepson and D. J. Heeger. Linear Subspace Methods for Recovering Translational Direction. Technical Report RBCV-TR-92-40, Department of Computer Science, University of Toronto, 1992.

[47] A. D. Jepson and Heeger D. J. Linear Subspace Methods for Recovering Translational Direction. In L. Harris and M. Jenkin, editors, *Spatial Vision in Humans and Robots*, pages 39–62. Cambridge University Press, 1993.

[48] B. Kamgar-Parsi and B. Kamgar-Parsi. Evaluation of Quantization Errors in Computer Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(9):929–940, 1989.

[49] K. Kanatani. Unbiased Estimation and Statistical Analysis of 3-D Rigid Motion from Two Views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(1):37–50, 1993.

[50] N. Kehtarnavaz, N. C. Griswold, and J. K. Eem. Comparison of Mono and Stereo Camera Systems for Autonomous Vehicle Tracking. In *SPIE Vol. 1468 Applications of Artificial Intelligence IX*, pages 467–478. International Society for Optical Engineering, 1991.

[51] D. Koller, K. Daniilidis, , and H.-H. Nagel. Model-Based Ojbect Tracking in Monocular Image Sequences of Road Traffic Scenes. *International Journal of Computer Vision*, 10(3):257–281, 1993.

[52] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems.* Prentice-Hall, 1974.

[53] C.-H. Lee. Time-Varying Images: The Effect of Finite Resolution on Uniqueness. *CVGIP: Image Understanding*, 54(3):325–332, 1991.

[54] A. Leon-Garcia. *Probability and Random Processes for Electrical Engineering.* Addison-Wesley, 1989.

[55] M. K. Leung, Y. Liu, and T. S. Huang. Estimating 3D Vehicle Motion in an Outdoor Scene from Monocular and Stereo Image Sequences. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 62–68. Institute of Electrical and Electronic Engineers, 1991.

[56] P.-G. Maillot. Using Quaternions for Coding 3D Transformations. In A. S. Glassner, editor, *Graphics Gems*, pages 498–515. Academic Press, 1990.

[57] R. R. Martin. Rotation by Quaternions. *Mathematical Spectrum*, 17(2):42–48, 1985.

[58] The Math Works, Inc. *MATLAB Reference Guide*, 1992.

[59] L. Matthies, T. Kanade, and R. Szeliski. Kalman Filter-based Algorithms for Estimating Depth from Image Sequences. *International Journal of Computer Vision*, 3(?):209–236, 1989.

[60] S. J. Maybank. Filter Based Estimates of Depth. In *British Machine Vision Conference*, pages 349–354. British Machine Vision Association, 1990.

[61] P. S. Maybeck. *Stochastic Models, Estimation, and Control (Volume 1)*. Academic Press, 1979.

[62] P. S. Maybeck. *Stochastic Models, Estimation, and Control (Volume 2)*. Academic Press, 1979.

[63] P. S. Maybeck. The Kalman Filter: An Introduction to Concepts. In I. J. Cox and G. T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 194–204. Springer-Verlag, 1990.

[64] J. Oliensis and J. I. Thomas. Incorporating Motion Error in Multi-Frame Structure from Motion. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 8–13. Institute of Electrical and Electronic Engineers, 1991.

[65] N. P. Papanikolopoulos and P. K. Khosla. Feature Based Robotic Visual Tracking of 3-D Translational Motion. In *IEEE Conference on Decision and Control*, pages 1877–1882. Institute of Electrical and Electronic Engineers, 1991.

[66] B. Rao. Data Association Methods for Tracking Systems. In A. Blake and A. Yuille, editors, *Active Vision*, pages 91–105. The MIT Press, 1992.

[67] H. S. Richardson. Multiple multistage hypothesis tests: A sequential detection approach to target tracking. Master's thesis, Queen's University, 1992.

[68] H. S. Richardson and S. D. Blostein. A Sequential Detection Framework for Feature Tracking within Computational Constraints. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 861–864. Institute of Electrical and Electronic Engineers, 1992.

[69] J. J. Rodriguez and J. K. Aggarwal. Stochastic Analysis of Stereo Quantization Error. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):467–470, 1990.

[70] B. Sabata and J. K. Aggarwal. Estimation of Motion from a Pair of Range Images: A Review. *CVGIP: Image Understanding*, 54(3):309–324, 1991.

[71] I. K. Sethi and R. Jain. Finding Trajectories of Features Points in a Monocular Image Sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):56–73, 1987.

[72] I. H. Shames. *Engineering Mechanics (Volume II).* Prentice-Hall, 1980.

[73] H. Shariat and K. E. Price. Motion Estimation with More Than Two Frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):417–434, 1990.

[74] K. Shoemaker. Animating Rotation with Quaternion Curves. In *SIGGRAPH '85*, pages 245–254. Association for Computing Machinery, 1985.

[75] O. Silvén and T. Repo. Solutions for Real-Time Visual Tracking. In *SPIE Vol. 1708 Applications of Artificial Intelligence X: Machine Vision and Robotics*, pages 184–195. International Society for Optical Engineering, 1992.

[76] O. Silvén and T. Repo. Experiments with Monocular Visual Tracking and Environment Modeling. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 84–92. Institute of Electrical and Electronic Engineers, 1993.

[77] H. W. Sorenson. Least-squares Estimation: from Gauss to Kalman. *IEEE Spectrum*, pages 63–68, 1970.

[78] M. E. Spetsakis and J. Aloimonos. Structure from Motion Using Line Correspondences. *International Journal of Computer Vision*, 4(3):171–183, 1990.

[79] M. W. Spong and M. Vidyasagar. *Robot Dynamics and Control.* John Wiley & Sons, 1989.

[80] G. W. Stewart and J. Sun. *Matrix Perturbation Theory.* Academic Press, 1990.

[81] G. Strang. *Linear Algebra and its Applications.* Harcourt Brace Jovanovich, 1988.

[82] M. Suk and S. M. Bhandarkar. *Three-Dimensional Object Recognition from Range Images.* Springer-Verlag, 1992.

[83] V. Sundareswaran. Egomotion from Global Flow Field Data. In *Proceedings of the IEEE Workshop on Visual Motion*, pages 140–145. Institute of Electrical and Electronic Engineers, 1991.

[84] V. Sundareswaran. A Fast Method to Estimate Sensor Translation. In *Proceedings of the Second European Conference on Computer Vision*, pages 253–257, 1992.

[85] M. A. Taalebinezhaad. Partial Implementation of the Fixation Method on Real Images: Direct Recovery of Motion and Shape in the General Case. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 400–405. Institute of Electrical and Electronic Engineers, 1991.

[86] M. A. Taalebinezhaad. Autonomous Fixation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 744–747. Institute of Electrical and Electronic Engineers, 1992.

[87] M. A. Taalebinezhaad. Direct Recovery of Motion and Shape in the General Space by Fixation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):847–853, 1992.

[88] E. J. Weldon, Jr. and H. Liu. How Accurately Can Direct Motion Vision Determine Depth? In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 613–618. Institute of Electrical and Electronic Engineers, 1991.

[89] J. Weng, P. Cohen, and N. Rebibo. Motion and Structure Estimation from Stereo Image Sequences. *IEEE Transactions on Robotics and Automation*, 8(3):362–382, 1992.

[90] J. Weng, T. S. Huang, and N. Ahuja. 3-D Motion Estimation, Understanding, and Prediction from Noisy Image Sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(3):370–389, 1987.

[91] J. Weng, T. S. Huang, and N. Ahuja. Motion and Structure from Two Perspective Views: Algorithms, Error Analysis, and Error Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5):451–476, 1989.

[92] J. Weng, T. S. Huang, and N. Ahuja. *Motion and Structure from Image Sequences.* Springer-Verlag, 1993.

[93] R. P. Wishner, J. A. Tabaczynski, and M. Athans. A Comparison of Three Non-Linear Filters. *Automatica*, 5:487–496, 1969.

[94] P. W. Wong. On Quantization Errors in Computer Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):951–956, 1991.

[95] J. J. Wu, R. E. Rink, T. M. Caelli, and V. G. Gourishankar. Recovery of the 3-D Location and Motion of a Rigid Object Through Camera Image (An Extended Kalman Filter Approach). *International Journal of Computer Vision*, 2(4):373–394, 1989.

[96] C. R. Wylie and L. C. Barrett. *Advanced Engineering Mathematics.* McGraw-Hill, 1982.

[97] G. J. Young and R. Chellappa. 3-D Motion Estimation Using a Sequence of Noisy Stereo Images: Models, Estimation, and Uniqueness Results. *IEEE*

Transactions on Pattern Analysis and Machine Intelligence, 12(8):735–759, 1990.

[98] G. S. Young, R. Chellappa, and T. H. Wu. Monocular Motion Estimation Using a Long Sequence of Noisy Images. In *International Conference on Acoustics, Speech and Signal Processing*, pages 2437–2440. Institute of Electrical and Electronic Engineers, 1991.

[99] Z. Zhang and O. D. Faugeras. *3-D Dynamic Scene Analysis: A Stereo-Based Approach*. Springer-Verlag, 1992.

[100] Z. Zhang and O. D. Faugeras. Three-Dimensional Motion Computation and Object Segmentation in a Long Sequence of Stereo Frames. *International Journal of Computer Vision*, 7(3):211–242, 1992.

Vita

Robert M. Chann

EDUCATION

Queen's University	M.Sc.	Electrical Engineering	1991–94
University of Waterloo	B.A.Sc.	Computer Engineering	1986–91

PUBLICATIONS AND PRESENTATIONS

Steven Blostein and Robert Chann (1993), "Hybrid Feature/Flow-Based 3-D Trajectory Estimation", *Proceedings of Vision Interface '93.*

Steven Blostein and Robert Chann (1994), "The Use of Image Plane Velocity Measurements in Recursive 3-D Motion Estimation from a Monocular Image Sequence", *Proceedings of the Canadian Conference on Electrical and Computer Engineering.*

K.L. Ambach, P. Esfahani, R.M. Chann, J.S. Burton, and A.W. Toga (1994), "Digital Correlation of High Resolution Anatomy and Histology from Cryosectioned Human Brain", *Society for Neuroscience Abstracts, 24th Annual Meeting.*