POSE AND MOTION ESTIMATION FROM VISION USING DUAL QUATERNION-BASED EXTENDED KALMAN FILTERING

A Dissertation

Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

James Samuel Goddard, Jr.

December 1997

ACKNOWLEDGEMENTS

I would like to thank my adviser, Dr. Mongi Abidi, whose continued encouragement and guidance have enabled me to bring this extensive effort to a successful conclusion. In addition, I would like to express gratitude to my committee members, Drs. Rajiv Dubey, Walter Green, Michael Roberts, and Ross Whitaker for their extremely helpful assistance and comments while completing my dissertation. Lastly, I would like to express special appreciation to my wife and children for their support and patience during this long process.

ABSTRACT

Determination of relative three-dimensional (3–D) position, orientation, and relative motion between two reference frames is an important problem in robotic guidance, manipulation, and assembly as well as in other fields such as photogrammetry. A solution to this problem that uses two-dimensional (2–D), intensity images from a single camera is desirable for real-time applications. Where the object geometry is unknown, the estimation of structure is also required. A single camera is advantageous because a standard video camera is low in cost, setup and calibration are simple, physical space requirements are small, reliability is high, and low-cost hardware is available for digitizing and processing the images. A difficulty in performing this measurement is the process of projecting 3–D object features to 2–D images, a nonlinear transformation. Noise is present in the form of perturbations to the assumed process dynamics, imperfections in system modeling, and errors in the feature locations extracted from the 2–D images. This dissertation presents solutions to the remote measurement problem for a dynamic system given a sequence of 2–D intensity images of an object where feature positions of the object are known relative to a base reference frame and where the feature positions are unknown relative to a base reference frame. The 3–D transformation is modeled as a nonlinear stochastic system with the state estimate providing six degree-of-freedom motion and position values. The stochastic model uses the iterated extended Kalman filter as an estimator and as a screw representation of the 3–D transformation based on dual quaternions. Dual quaternions provide a means to represent both rotation and translation in a unified notation. The method has been implemented and tested with both simulated and actual experimental data. Simulation results are provided along with comparisons to a point-based method using rotation and translation to show the relative advantages of the proposed method. Experimental test results using a camera mounted on the end effector of a robot arm to demonstrate relative motion and position control are also presented.

TABLE OF CONTENTS

Chapter 1. Introduction 1

1.1 Problem Statement 1

1.2 Motivation 5

1.3 Related Work 7

1.3.1 Pose Using Point-Based Methods 8

1.3.2 Methods for Real-Time Pose Determination 10

1.3.3 Pose Using Model-Based and Higher-Level Primitives Methods 17

1.3.4 Methods Incorporating Noise Estimation and Uncertainty . . . 26

1.3.5 Methods Using Kalman Filtering for Direct Pose and Motion Estimation 29

1.4 Synopsis of Dissertation 33

Chapter 2. Theoretical Foundation 35

2.1 Coordinate Frame Transformations 35

2.2 Quaternions 36

2.3 Dual Numbers 38

2.4 Dual Number Quaternions 40

2.5 3-D to 2-D Perspective Projection 41

2.6 Representation of 3-D Rotation by Quaternions 43

2.7 Representation of 3–D Rotation and Translation by Dual Number Quaternions 45

2.8 Iterated Extended Kalman Filter 47

Chapter 3. Pose and Motion Estimation Method 54
3.1 Pose and Motion Estimation Using IEKF and Dual Number Quaternions 55
3.2 State Assignment 55
3.3 System Model 57
3.4 Measurement Model 60
3.5 Representation of Lines in a Plane 62
3.6 Linearization of the State Transition Function 65
3.7 Linearization of the Measurement Function 67
3.8 Iterated Extended Kalman Filter Representation 72
3.9 Extension of Iterated Extended Kalman Filter to Estimate Structure . 73
3.9.1 Structure Representation 73
3.9.2 Extension to State Transition 74
3.9.3 Extension to Measurement Update 74

Chapter 4. Noise Analysis 76
4.1 Analysis of Line Noise 76
4.1.1 Probability Density Function of Line-Point Parameters 77
4.1.2 Covariance of x_{lp} and y_{lp} 87
4.2 Real-Time Implementation 93

Chapter 5. Experimental Results 96
5.1 Simulation Results 96
5.1.1 Dynamic Model 97
5.1.2 Model Trajectory 99
5.1.3 Camera Model 100
5.1.4 Object Model 101
5.1.5 Estimation Model 101

5.1.6 Comparison Model 102
5.1.7 Initial Conditions 102
5.1.8 Measurement Noise Model 102
5.1.9 Test Results 103
5.1.10 Simulation Using Adaptive Noise Estimation 130
5.2 Robot Arm Tests 132
5.2.1 Experimental Setup 136
5.2.2 Dynamic Model 136
5.2.3 Measurement Model 136
5.2.4 Initial Conditions 139
5.2.5 Measurement Noise Results 139
5.2.6 Target Motion Results 139
5.3 Test Results From Unknown Object Geometry 148
5.3.1 Structure Estimation 148
5.3.1.1 Calculation of Actual Structure for Reference 150
5.3.1.2 Initial State Estimate 151
5.3.1.3 Transformation of Structure Estimate Frame to Reference Frame 153
5.3.2 Response to Step Change in Motion 154

Chapter 6. Summary and Conclusions 159
6.1 Summary 159
6.2 Conclusions and Contributions 160
6.3 Future Work 164

References 166

Vita 179

LIST OF TABLES

5.1 Actual and assumed initial states of the extended Kalman filter for the four-point simulation tests. The initial states are the same for both the dual quaternion method and the point method. 104

5.2 Dual quaternion method and point method initial error covariance matrix and process noise matrix diagonal terms of the extended Kalman filter for the four-point simulation tests. 104

5.3 Actual and assumed initial states of the dual quaternion and point-based extended Kalman filter for simulation comparison with the adaptive measurement noise line method, the nonadaptive line method, and the comparison point method. 131

5.4 Dual quaternion method and point method initial error covariance matrix and process noise matrix diagonal terms of the extended Kalman filter for simulation comparison with the adaptive measurement noise line method, the nonadaptive line method, and the comparison point method. 131

5.5 Reference pipe line data as calculated by two image views separated by an x translation of 50 mm. The data show the $\vec{l}$ and $\vec{m}$ components of the 3–D lines. 151

LIST OF FIGURES

1.1 Transformation between two reference frames illustrating the fundamental problem to be solved. 3

1.2 Block diagram of a robot arm feedback control loop showing an application for pose and motion estimation. A camera mounted on the end effector of the robot arm is used as the measurement sensor. 6

2.1 3–D to 2–D perspective projection model assumed in pose estimation problem. The center of projection is the camera reference origin. 42

2.2 Diagram showing screw form of 3–D transformation between two reference frames. Eight parameters are needed to specify the complete transformation. 46

2.3 Extended Kalman filter block diagram showing functionality along with the inputs and outputs. The filter is seen to have a closed-loop feedback control where the measurement error is minimized. 52

3.1 Functional block diagram of estimation problem showing operations involved in pose estimation method from optical imaging to state estimate produced by the iterated extended Kalman filter. 58

3.2 Line point for a line in 2-D image plane is defined as the intersection point of the line and a perpendicular line passing through the origin. 64

4.1 Diagram showing a point pair with a noise radius of one standard deviation and the corresponding range of variation of the line connecting the points. 77

4.2 Probability density function of x_{lp} for 1.25 mm. length horizontal line centered about the y axis, y_s= 3.125 mm., point noise 0.02 mm. standard deviation. A best-fit Gaussian pdf is also plotted, but the differences are seen to be small. 83

4.3 Probability density function of y_{lp} for 1.25 mm. length horizontal line centered about the y axis, y_s= 3.125 mm., point noise 0.02 mm. standard deviation. A best-fit Gaussian pdf is also plotted, but the differences are seen to be small. 84

4.4 Probability density function of x_{lp} for 1.25 mm. length downward diagonal line at x_s=12.5 mm., y_s= 12.5 mm., point noise 0.02 mm. standard deviation. A best-fit Gaussian pdf is also plotted, but the differences are seen to be small. 85

4.5 Probability density function of x_{lp} for 1.25 mm. length upward diagonal line at x_s=12.5 mm., y_s= 12.5 mm., point noise 0.02 mm. standard deviation. A best-fit Gaussian pdf is also plotted, but the differences are seen to be small. 86

4.6 Variance of line-point parameters for 1.25 mm. horizontal line centered on the y axis as y_s varies, point noise 0.02 mm. standard deviation, compared with point noise variance. y_{lp} variance is always less than the point variance while the x_{lp} variance becomes greater after only a small displacement. 91

4.7 Variance of line-point parameters for 1.25 mm. diagonal line (45 degree angle) centered on the y axis as y_s varies, point noise 0.02 mm. standard deviation, compared with point noise variance. x_{lp} variance is always less than the point variance while the y_{lp} variance becomes greater after only a small displacement. 92

5.1 Functional block diagram of the estimation problem showing the operations involved in the pose estimation method from optical imaging to state estimate produced by the iterated extended Kalman filter. 98

5.2 Target shapes and dimensions used in the simulation experiments: (a) four-point coplanar target, and (b) six-point polyhedral 3-D target. 101

5.3 t_x translation pose estimate simulation results with a four-point target showing linear translation over time for the line method, the comparison point method, and the true x translation value from a sample run. Both methods are seen to give good estimation results for this axis. 106

5.4 t_y translation pose estimate simulation results with a four-point target showing linear translation over time for the line method, the comparison point method, and the true y translation value from a sample run. Both methods are seen to give good estimation results for this axis. 106

5.5 t_z translation pose estimate simulation results with a four-point target showing linear translation over time for the line method, the comparison point method, and the true z translation value from a sample run. More deviation from the true value is seen for this axis than in x and y although the errors are not large. 107

5.6 q_0 quaternion pose estimate simulation results with a four-point target showing constant rotation over time for the line method, the comparison point method, and the true q_0 quaternion value from a sample run. Both methods are seen to give good estimation results for this state variable. . 107

5.7 q_1 quaternion pose estimate simulation results with a four-point target showing constant rotation over time for the line method, the comparison point method, and the true q_1 quaternion value from a sample run. More deviation from the true value is seen for this axis than in the q_0 estimate with significant errors occurring at the beginning and at the end. 108

5.8 q_2 quaternion pose estimate simulation results with a four-point target showing constant rotation over time for the line method, the comparison point method, and the true q_2 quaternion value from a sample run. More deviation from the true value is seen for this axis than in the q_0 estimate with significant errors occurring at the beginning and at the end. 108

5.9 q_3 quaternion pose estimate simulation results with a four-point target showing constant rotation over time for the line method, the comparison point method, and the true q_3 quaternion value from a sample run. Both methods are seen to give good estimation results for this state variable. . 109

5.10 v_x translation velocity pose estimate simulation results with a four-point target showing constant linear velocity over time for the line method, the comparison point method, and the true v_x velocity value of -5 mm./sec. from a sample run. Both methods are seen to give good estimation results for this axis after the initial settling time. 109

5.11 v_y translation velocity pose estimate simulation results with a four-point target showing constant linear velocity over time for the line method, the comparison point method, and the true v_y velocity value of 2 mm./sec. from a sample run. Both methods are seen to give good estimation results for this axis after the initial settling time. 110

5.12 v_z translation velocity pose estimate simulation results with a four-point target showing constant linear velocity over time for the line method, the comparison point method, and the true v_z velocity value of -5 mm./sec. from a sample run. The line method estimates are seen to settle faster to the true value and with less error than the point method. 110

5.13 ω_x angular velocity pose estimate simulation results with a four-point target showing constant angular velocity over time for the line method, the comparison point method, and the true ω_x angular velocity value of -0.03 rad./sec. from a sample run. Both methods are seen to give good estimation results for this rotation angle after the initial settling time, although, higher divergence is present for the point method near the end of the simulation. 111

5.14 ω_y angular velocity pose estimate simulation results with a four-point target showing constant angular velocity over time for the line method, the comparison point method, and the true ω_y angular velocity value of 0.05 rad./sec. from a sample run. Both methods are seen to give good estimation results for this rotation angle after the initial settling time, although, higher divergence is present for the point method near the end of the simulation. 111

5.15 ω_z angular velocity pose estimate simulation results with a four-point target showing constant angular velocity over time for the line method, the comparison point method, and the true ω_z angular velocity value of -0.2 rad./sec. from a sample run. Both methods are seen to give good estimation results for this rotation angle after the initial settling time. 112

5.16 t_x translation pose estimate simulation results for a four-point target showing linear translation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time. . 112

5.17 t_y translation pose estimate simulation results for a four-point target showing linear translation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time. . 113

5.18 t_z translation pose estimate simulation results for a four-point target showing linear translation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is larger than for the x and y translations. Response time is faster for the line method with somewhat smaller errors over the simulation time. 113

5.19 q_0 quaternion pose estimate simulation results for a four-point target showing rotation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time. 114

5.20 q_1 quaternion pose estimate simulation results for a four-point target showing rotation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time except at the end where the point method has somewhat larger error. 114

5.21 q_2 quaternion pose estimate simulation results for a four-point target showing rotation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time except at the end where the point method has somewhat larger error. 115

5.22 q_3 quaternion pose estimate simulation results for a four-point target showing rotation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time. 115

5.23 v_x translation velocity pose estimate simulation results for a four-point target showing linear translation velocity estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small; the line method errors are somewhat less over the simulation time. 116

5.24 v_y translation velocity pose estimate simulation results for a four-point target showing linear translation velocity estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small; the line method errors are somewhat less over the simulation time. 116

5.25 v_z translation velocity pose estimate simulation results for a four-point target showing linear translation velocity estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be larger than for the x and y velocity estimates. The line method has faster settling time with somewhat lower errors than the point method over the simulation time. 117

5.26 ω_x angular velocity pose estimate simulation results for a four-point target showing constant angular velocity error over time for the line method and the comparison point method from a sample run. The deviation for both methods is relatively large compared with the true velocity, although, the mean error is near zero. 117

5.27 ω_y angular velocity pose estimate simulation results for a four-point target showing constant angular velocity error over time for the line method and the comparison point method from a sample run. The deviation for both methods is relatively large compared with the true velocity, although, the mean error is near zero. 118

5.28 ω_z angular velocity pose estimate simulation results for a four-point target showing constant angular velocity error over time for the line method and the comparison point method from a sample run. The deviation for both methods is small compared with the true velocity over the simulation time. 118

5.29 t_x translation pose estimate simulation results for a four-point target showing linear translation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over most of the time interval. The Kalman filter predicted error is close to the actual error for the line method. 120

5.30 t_y translation pose estimate simulation results for a four-point target showing linear translation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over most of the time period. The Kalman filter predicted error is close to the actual error for the line method. 120

5.31 t_z translation pose estimate simulation results for a four-point target showing linear translation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over the last half of the simulated time period. The Kalman filter predicted error is close to the actual error for the line method. 121

5.32 q_0 quaternion pose estimate simulation results for a four-point target showing rotation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be small for both methods over the simulated time period. The Kalman filter predicted error for the line method greatly overestimates the actual error. 121

5.33 q_1 quaternion pose estimate simulation results for a four-point target showing rotation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be considerably larger for the point method than the line method over the simulated time period. The Kalman filter predicted error for the point method greatly underestimates the actual error while the predicted error for the line method is close to the actual error except near the end of the time period. 122

5.34 q_2 quaternion pose estimate simulation results for a four-point target showing rotation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be considerably larger for the point method than the line method over the simulated time period. The Kalman filter predicted error for the point method greatly underestimates the actual error while the predicted error for the line method is close to the actual error except near the end of the time period. 123

5.35 q_3 quaternion pose estimate simulation results for a four-point target showing rotation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be small for both methods over the simulated time period. The Kalman filter predicted error for the line method greatly overestimates the actual error. 124

5.36 v_x translation velocity pose estimate simulation results for a four-point target showing linear translation root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over most of the time period. The Kalman filter predicted error is close to the actual error for the line method. 124

5.37 v_y translation velocity pose estimate simulation results for a four-point target showing linear translation root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over most of the time period. The Kalman filter predicted error is close to the actual error for the line method. 125

5.38 v_z translation velocity pose estimate simulation results for a four-point target showing linear translation root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method, particularly during the initial settling time. The Kalman filter predicted error is close to the actual error for the line method but considerably underestimates the actual error for the point method. 126

5.39 ω_x angular velocity pose estimate simulation results for a four-point target showing angular root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over the latter portion of the simulation time. The Kalman filter predicted error is close to the actual error for the line method but considerably underestimates the actual error for the point method. 127

5.40 ω_y angular velocity pose estimate simulation results for a four-point target showing angular root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over the latter portion of the simulation time. The Kalman filter predicted error is close to the actual error for the line method but considerably underestimates the actual error for the point method. 128

5.41 ω_z angular velocity pose estimate simulation results for a four-point target showing angular root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over the latter portion of the simulation time. The Kalman filter predicted error is close to the actual error for the line method and for the point method. 129

5.42 t_x translation pose estimate simulation results for a four-point target offset from the image center. The figure shows linear translation root mean square estimation error over time for the adaptive line method, the nonadaptive line method and the comparison point method calculated from 100 sample runs. x and y translations were both initially 400 mm. The RMS error is seen to be significantly less for the point method than for the adaptive and nonadaptive line methods over most of the time period. Initial error overshoot is also much less with the point method. The adaptive line method shows less error than the nonadaptive method over most of the simulation. 133

5.43 t_y translation pose estimate simulation results for a four-point target offset from the image center. The figure shows linear translation root mean square estimation error over time for the adaptive line method, the nonadaptive line method and the comparison point method calculated from 100 sample runs. x and y translations were both initially 400 mm. The root mean square error is seen to be significantly less for the point method than for the adaptive and nonadaptive line methods over most of the time period. Initial error overshoot is also much less with the point method. The adaptive line method shows less error than the nonadaptive method over most of the simulation. 134

5.44 t_z translation pose estimate simulation results for a four-point target offset from the image center. The figure shows linear translation root mean square estimation error over time for the adaptive line method, the nonadaptive line method and the comparison point method calculated from 100 sample runs. x and y translations were both initially 400 mm. The root mean square error is seen to be significantly less for the point method than for the adaptive and nonadaptive line methods over most of the time period. Initial error overshoot is also much less with the point method. The adaptive line method shows less error than the nonadaptive method over most of the simulation. 135

5.45 Mitsubishi RV-E2 six-axis robot arm and the Cidtec camera used in the experimental tests. 137

5.46 Block diagram showing the system hardware used in the experimental setup. The hardware includes a Cidtec camera, frame grabber, PC compatible computer, and Mitsubishi robot arm and controller. 138

5.47 Measurement functions applied to obtain line point measurements from the camera image in robot arm experiments. 138

5.48 Histogram of the distribution of x line point measurement values from 500 camera images for a stationary target. 140

5.49 Histogram of the distribution of y line point measurement values from 500 camera images for a stationary target. 140

5.50 t_x translation pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a sinusoidal variation corresponding to the true x variation of the target. Note the step response at the beginning of the target motion. 141

5.51 t_y translation pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a sinusoidal variation corresponding to the true y variation of the target. Note the step response at the beginning of the target motion. 142

5.52 t_z translation pose estimate from experimental testing for constant target motion about z axis. The estimate shows a noisy variation about a mean z value with a small sinusoidal component. The true target z value is nominally constant. 142

5.53 q_0 quaternion pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a full 360 degree sinusoidal variation corresponding to the true q_0 variation of the target as it moves through two complete revolutions. 143

5.54 q_1 quaternion pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a relatively small noisy variation about zero in comparison to a true q_1 constant value of zero. . . 143

5.55 q_2 quaternion pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a relatively small noisy variation about zero in comparison to a true q_2 constant value of zero. . . 144

5.56 q_3 quaternion pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a full 360 degree sinusoidal variation corresponding to the true q_3 variation of the target as it moves through two complete revolutions. 144

5.57 v_x translation velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy sinusoidal variation compared to the true x sinusoidal velocity variation of the target. Note the step response at the beginning of the target motion. 145

5.58 v_y translation velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy sinusoidal variation compared to the true y sinusoidal velocity variation of the target. Note the step response at the beginning of the target motion. . . . 145

5.59 v_z translation velocity pose estimate from experimental testing for constant target motion about z axis. The estimate shows a noisy variation about a mean z velocity value of zero. The true target v_z velocity value is nominally zero. 146

5.60 ω_x angular velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy variation about a mean ω_x velocity value of zero. The true target ω_x velocity value is nominally zero. 146

5.61 ω_y angular velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy variation about a mean ω_y velocity value of zero. The true target ω_y velocity value is nominally zero. 147

5.62 ω_z angular velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy variation about a mean ω_z velocity value of -0.06 rad./sec. The true target ω_z velocity value is nominally constant at -0.06 rad./sec. 147

5.63 Pipe assembly used in the structure estimation robot arm experiments. The picture shows the four segments of white PVC pipe joined together. . 149

5.64 Structure estimate of the three $\vec{l}$ line vector components for line zero with motion along the z axis. The true structure values are also shown for comparison. The x and y components are close to the true values while the z component has a constant offset error. 155

5.65 Structure estimate of the three $\vec{m}$ line vector components for line zero with motion along the z axis. The true structure values are also shown for comparison. The y and z components are accurately estimated while the x component has significant error. 155

5.66 Structure estimate of the three $\vec{l}$ line vector components for line one with motion along the z axis. The true structure values are also shown for comparison. The x and y components have small errors in the estimate while the z component is seen to converge slowly to the true value. 156

5.67 Structure estimate of the three $\vec{m}$ line vector components for line one with motion along the z axis. The true structure values are also shown for comparison. The z component is estimated accurately while the x component is seen to converge to the correct value. Error in y, though initially small, increases with time. 156

5.68 Structure estimate of the three $\vec{l}$ line vector components for line two with motion along the z axis. The true structure values are also shown for comparison. Accurate estimates are obtained for all three components. . . 157

5.69 Structure estimate of the three $\vec{m}$ line vector components for line two with motion along the z axis. The true structure values are also shown for comparison. Accurate estimates are obtained for all three components. . . 157

5.70 Structure estimate of the three $\vec{l}$ line vector components for line four with motion along the z axis. The true structure values are also shown for comparison. The x and y components are close to the true values while the z component has a constant offset error. 158

5.71 Structure estimate of the three $\vec{m}$ line vector components for line four with motion along the z axis. The true structure values are also shown for comparison. The x and z components have relatively small errors while a constant offset error is present for the y component. 158

Chapter 1

Introduction

1.1 Problem Statement

Estimation of relative three-dimensional (3–D) position and orientation (pose) and structure as well as relative motion between two reference frames is an important problem in robotic guidance, manipulation, and assembly as well as in other areas such as photogrammetry, tracking, object recognition, and camera calibration. Pose is defined for an object in 3–D cartesian space consisting of an object reference frame and a base reference frame. The pose with respect to the base frame comprises the three position coordinates of the object reference frame origin and the three orientation angles of the object frame. Remote estimation of the relative pose and motion of a 3–D object without physically touching the object or without human intervention is the fundamental problem examined here. For example, an autonomous robot that moves along a path may need to determine its position relative to an obstacle. Similarly, a mechanical hand grasping a moving object requires that the gripper motion be matched to the object and then placed at the correct position and orientation. Automated spacecraft docking requires the measurement of pose relative to the docking port. The pose can be expressed relative to the sensing element frame or with respect to the base frame.

This pose estimation or determination problem has been a subject of considerable research for many years in computer vision, photogrammetry, and robotics with many solutions having been proposed. The methods available are generally deterministic and use single-vision cameras, stereo-vision cameras, or more direct 3–D measuring techniques

such as range images from laser, ultrasonic, and structured lighting devices. Two general classifications of sensors can be made: active and passive. Active devices (e.g., laser range finders) are those which emit energy and sense its reflection to determine the sensor response, while passive devices such as video cameras sense the natural or background energy (e.g., visible light or infrared) reflected off an object. This distinction is important in an application such as a battlefield environment, where a sensor that emits energy could be detected by enemy surveillance. However, relying on background energy alone may be insufficient in providing a good signal-to-noise ratio for feature detection and object recognition. Poor visibility caused by fog, smoke, or rain can produce images that are highly variable in quality with difficult to recognize features. Noise considerations, therefore, must be taken into account to a much greater extent than would be the case in a laboratory environment where illumination can be easily controlled. Even in a well-controlled setting, sensor noise must be considered since intrinsic sources can introduce a substantial amount of measurement noise. Depending on the accuracy required for pose measurement, this noise can either be ignored or the effects can be minimized. Generally, the highest accuracy is needed for camera calibration while model matching or grasping may permit larger errors in the estimate. Accuracy requirements vary for docking and tracking applications.

The general problem developed here is to locate an object and measure its motion in three dimensions based on three position coordinate parameters and three rotation coordinate parameters either relative to an observer or with respect to a fixed-reference frame. Figure 1.1 shows the basic problem with each reference frame and transformation shown.

Known as the exterior orientation problem in photogrammetry [1], this question has been addressed for photographs using a number of manual methods dating back to 1879 [2]. More recently, beginning in the 1960's, methods using computer vision techniques

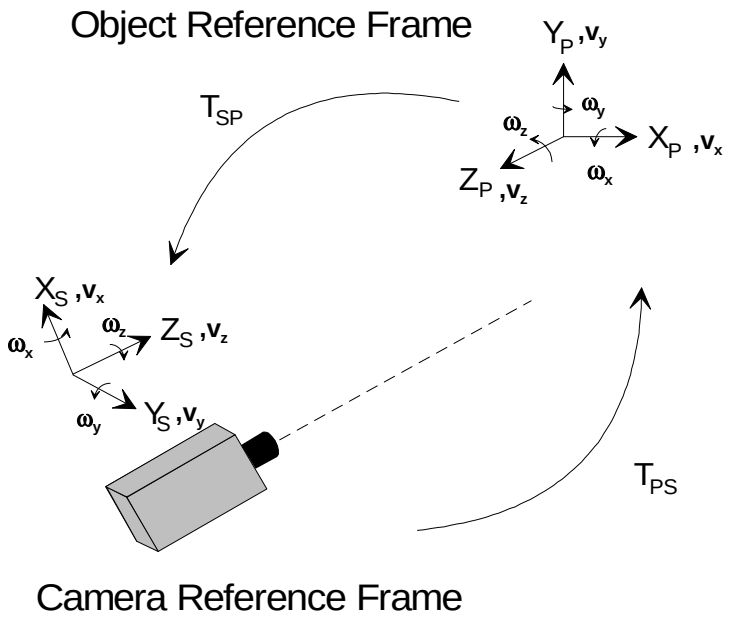

Fig. 1.1. Transformation between two reference frames illustrating the fundamental problem to be solved.

have been developed. Most measurement techniques for pose determination or estimation are image-based and can be classified into two major categories. These categories are point-based methods and model-based methods using higher-order geometric primitives. Each type involves acquiring an image, either two-dimensional (2–D) or 3–D, and then processing that image to arrive at a value for the pose. Noise effects may also be analyzed for either type. Specific techniques and applications that have been proposed are described in Section 1.3.

The particular sensor used can determine the ultimate accuracy achieved. The precise layout of the elements in the sensing charge coupled device (CCD) array along with the use of low distortion lenses gives low geometric distortion for CCD imaging cameras. Calibration for lens distortion can reduce the error still further. Feature detection and extraction software can then locate an object's feature in 2–D spatial coordinates to subpixel accuracy provided the signal-to-noise ratio is high. Noise and uncertainty are present due to the response nonuniformity of the individual CCD pixels along with photon and thermal noise. Laser range finders likewise have uncertainty in both range as well as geometric location correspondence. The imaged scene also influences the accuracy achieved. Extracting features from an object of low contrast with its surroundings is not as straightforward nor as accurate as one with high contrast. To overcome some of the limitations of natural objects, artificial markings may be placed on the object whose position relative to the object is known. These markings can consist of a high-contrast pattern that maximizes signal-to-noise ratio and optimizes a particular pose determination algorithm.

Other factors to be considered for pose and motion determination are the computational complexity and robustness of a method. The algorithm chosen may require a large computational overhead due to an optimization solution technique requiring many iterations. The choice of method usually involves a trade-off between accuracy and noise

rejection versus computation time. Robustness in this context is defined as the ability of an algorithm to provide a result within a known error bound or to indicate that the error condition cannot be met. Measurement data outside a known error bound relative to a working model may be ignored or given low weight. Obtaining accurate results for autonomous pose determination in everyday environments in which people operate with ease is a difficult problem and one that has not yet been fully solved. Variable conditions such as lighting and visibility, along with the presence of noise from various sources, are principle difficulties that need to be examined. This work is an attempt to model these uncertainties and imprecisions in such environments as applied to pose and motion estimation. The term estimation is used as opposed to determination since the modeling is done in a statistical sense where estimated results are computed. Other methods may determine the pose by directly calculating a set of equations or by calculating a least squared error solution that does not model the noise or process stochastically.

1.2 Motivation

A solution to this problem that uses two 2–D intensity images from a single camera is desirable in many instances where *a priori* geometric knowledge about an object in the unknown reference frame is available. One direct application for this problem is in robotic assembly and navigation. Figure 1.2 shows in block diagram form how pose and motion estimation could be applied in a robot arm feedback control loop. The diagram illustrates the measurement of the pose and motion of a camera mounted on the end of a robot arm with respect to an object of interest. This measurement is used as feedback for controlling the position of the arm.

Use of a single camera is advantageous for the following reasons: a CCD video camera is low in cost; setup and calibration is simple; physical space requirements are small; reliability is high; and standard low-cost hardware is available for digitizing and

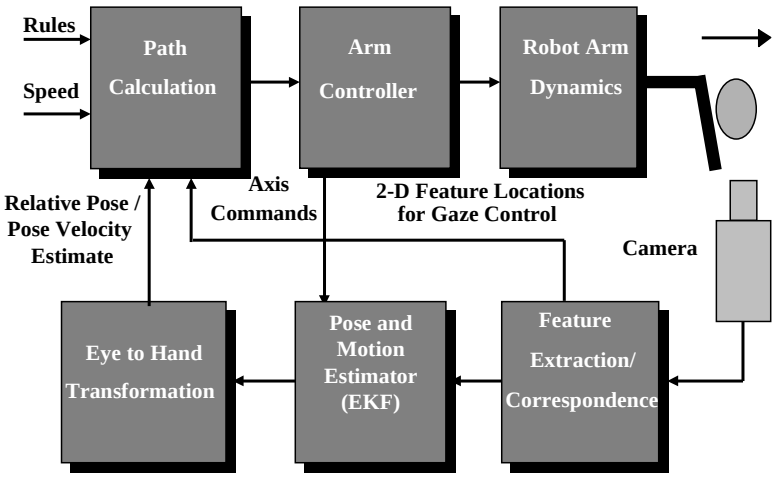

Fig. 1.2. Block diagram of a robot arm feedback control loop showing an application for pose and motion estimation. A camera mounted on the end effector of the robot arm is used as the measurement sensor.

processing the images. The primary difficulty in performing this measurement is that depth information is lost in the process of projecting 3–D object features to 2–D images, a nonlinear transformation. Noise is also present in the form of perturbations to the assumed process dynamics, imperfect system modeling, and feature locations extracted from the 2–D images.

This dissertation presents a solution to the remote measurement problem for a dynamic system given a sequence of 2–D intensity images of an object whose position and orientation are known relative to a base reference frame. The 3–D transformation is modeled as a nonlinear stochastic system with the state estimate providing the six degree-of-freedom motion and position values. The stochastic model uses the iterated extended Kalman filter (IEKF) as an estimator and a screw representation of the 3–D transformation based on dual quaternions. Dual quaternions, whose elements are dual numbers, provide a means to represent both rotation and translation in a unified notation. In the following, prior related work is reviewed pertaining to general methods of autonomous

pose estimation and determination of 3–D objects and to methods that model noise as part of the solution.

Previous solutions have used point-based image features in estimating the structure, pose, and motion. This work, instead, uses image line features as measurement inputs for the estimation. Line features are present in many scenes and objects to a greater extent than point features. They may be more visible, as well, under a wider range of lighting and environmental conditions than points. Also, straightforward techniques such as the Hough transform and line fitting to edges are available to extract the lines from the images. For these reasons, an approach was taken to develop the nonlinear estimation problem strictly in terms of 3–D lines used to describe an object or scene and 2–D image lines as measured quantities.

1.3 Related Work

This section provides a summary of previous work relating to noncontact position and orientation measurement methods as described in the published literature. Existing research shows a large number of position determination methods applicable to a variety of applications. While not as numerous, pose estimation methods have also been described, and these are summarized as well. Variations in the methods include type and location of the sensor, illumination requirements, the object or scene feature on which the pose is calculated, relative motion of the robot or object, iterative versus direct solution, and modeling of uncertainty or noise in an attempt to improve the robustness and accuracy of the results. Applications proposed are for research programs or for commercial prototype development in the general areas of robot location, manufacturing, camera calibration, and tracking of a moving object.

1.3.1 Pose Using Point-Based Methods

Point-based methods rely on the identification and location of feature points on a target object from a 2–D image of the scene. A rigid body is generally assumed but no explicit geometric model is given. Information concerning the geometric shape other than size is not used in calculating the pose. Coordinates of the points in a local or world reference frame may or may not be known.

Methods of this class, referred to as N-point perspective, were the first to be studied and, as a result, have been more extensively developed than model-based methods [3]. A perspective model that assumes the projection of a 3–D object onto a 2–D image plane through a pinhole camera model is generally used [1]. Both single-image and stereo methods have been reported; however, single-vision techniques have, by far, the greatest number of solutions. One reason for this is that point correspondence with an object from a single image is easier to determine than correspondences between two images and the object as required in stereo. The general framework is, given N corresponding points in the object and in the image, to solve for the relative pose between the camera and the object. The minimum N that produces a finite number of solutions is three, although, up to four solutions are possible. Four coplanar, noncollinear points give a unique solution. Four or five noncoplanar, noncollinear points may result in two solutions. For N greater than five noncollinear points, the result is unique and consists of an overdetermined set that can be solved using least squared error methods [4]. In general, as N increases, the accuracy of the results increases [5]. These overdetermined solutions are used for camera calibration in which a large number of points are needed through minimization of an error criteria to achieve the desired accuracy and to calculate both the external and internal camera parameters [5]. Three- and four-point coplanar targets have been directly used for pose determination. With stereo cameras, three corresponding points on an object are sufficient to uniquely identify the relative pose of the object although uncertainty

may be reduced through a larger number of points [6]. Range images, similarly, can determine pose with a minimum of three points. An advantage of 3–D range images over stereo is that the correspondence problem is not present since the three coordinates of an object point are determined directly. Algorithms for these techniques are primarily iterative. For the three- and four-point special cases, however, closed-form solutions have been demonstrated [7, 4, 8, 9].

Trabasso and Zielinski [10] describe an approximate calibration method for calculating exterior orientation parameters and the x and y scaling factors. A calibration block consisting of four coplanar points is assumed to be perpendicular to the optical axis of the camera. This method has been tested in an experimental work cell for robot grasping and placement of automobile bodyshells whose plane is approximately perpendicular to the camera axis. Placement accuracies of one millimeter were achieved.

Tsai has described a technique for high-accuracy, 3–D camera calibration using standard television cameras and lenses [5]. This paper gives a good survey of the calibration and pose determination techniques at the time of its publication. It also proposes a new two-stage method based on the radial alignment constraint that determines the six extrinsic and intrinsic parameters including focal length, two radial lens distortion coefficients, and the x coordinate scale factor. The first stage is a direct linear solution neglecting lens distortion. This result is used as the initial estimate in the nonlinear second stage that takes into account the lens distortion. An overdetermined set of points is used to achieve high accuracies using least squared error fitting. Using the radial alignment constraint, solutions are given for both coplanar calibration points and noncoplanar points. Test results are given where the number of calibration points used is 60. Total accuracy in 3–D is about one part in 2000. Fewer points gave higher errors while more points did not give a significant improvement in accuracy. The error obtained is approximately one-half the total theoretical predicted error.

A 3–D location method based on dual number quaternions is given by Walker et al. [11]. The method is formulated as a single-cost function to be minimized where the real part and dual part of the dual quaternion are used to represent rotation and translation. Conversions between the conventional homogeneous transformation matrix and the dual quaternion parts are given. An algorithm for the localization is detailed. Inputs are in terms of 3–D measured points and unit vectors corresponding to surface normals or to edge directions. Phong et al. [12] also describe a pose estimation method based on dual number quaternions. Their method is a 2–D to 3–D pose estimation rather than the 3–D to 3–D as given by Walker. Line features, or lines defined by pairs of corresponding points, are used as correspondences between the image and the 3–D object. The problem is set up as a sum of quadratic constraints. A trust region optimization algorithm is used to solve for the real part and dual part of the transformation. Experimental results are given for various numbers of line correspondences and varying amounts of added noise. Two sets of results are given. In the first set, the real quaternion is solved first and then used to solve for the dual part. The second set of results is from the simultaneous solution of both parts of the dual quaternion. In general, the simultaneous solution had less error but required more computation. This method is extended in [13] to point features based on the collinearity between the object point, the center of projection, and the image point.

1.3.2 Methods for Real-Time Pose Determination

Real-time methods are intended for those applications requiring fast response for control or needing low computational overhead for operation with low-cost hardware. These methods generally use a relatively small number of feature points, with four coplanar points being most widely used. Abidi and Chandra [8] propose a new closed-form algorithm for relative pose determination based on the volume measurement of tetrahedra. Using triples of the corner points of a quadrangular target, the volumes of the solids

defined by these points and the lens center are calculated. Object pose is recovered using these volumes without prior knowledge of the lens focal length as long as the coplanar target and the optical axis are not perpendicular. A solution is also given that uses the focal length as an input to remove this restriction. Lens distortion correction is applied to the image feature points before using the algorithm. To achieve higher accuracies, a nonlinear iterative optimization stage is given that is applied to the results of the direct algorithm. Experimental results and simulations show that average errors of 0.84% in distance and 0.66% for angles were obtained using the direct method alone, assuming pixel standard deviation noise of 0.5 pixels. Applying the optimization step gave errors of 0.19% and 0.05% respectively. Results are also given showing sensitivities of the pose to parameter changes and to the shape and relative size of the target. Target shape was not found to have a significant effect on accuracy. Abidi and Gonzalez [14] give an alternate method to compute the relative pose based on an algebraic solution. Three points are shown to be required to give a finite number of solutions. A fourth coplanar point is then shown to give a unique solution as long as no subset of three points are collinear.

An early paper on pose determination based on machine vision was written by Chen et al. [15] that extracted feature points on a workpiece. At least three points were extracted from an image of the workpiece that was then rotated through a known displacement with a second image being taken. Through stereo correspondence, the 3–D locations of the points relative to the camera were calculated. The transformation matrix is then calculated from the matched points. Haralick and Joo [2] present robust techniques for calculating the extrinsic pose parameters from a single image. Iterative procedures with both robust and nonrobust methods for calculating a least squared error solution are given. Experimental results and simulations show that the robust method has considerable advantage when outliers are present in the data. It is stated that outliers are

generally due to incorrect correspondence of point pairs, an occurrence which is commonly encountered.

Haralick et al. give closed-form iterative, least squared error solutions based on point correspondences for four different pose estimation problems [16]. These problems are 2–D to 2–D estimation, 3–D to 3–D estimation, 2–D perspective to 3–D, and 2–D perspective to 2–D perspective. A robust algorithm is also presented for each problem. The effects of varying amounts of noise on these solutions are also investigated. Both independent additive Gaussian noise and independent additive noise from a slash distribution are used in simulations to show the performance of each algorithm. The slash distribution is used to simulate the presence of outliers in the data due primarily to correspondence mismatches in the data. Experimental results show the advantage of the robust algorithm for this noise distribution. The results also show that hundreds of corresponding points must be used to achieve reasonable accuracies when the signal-to-noise ratio is less than 40 decibels.

An automatic spacecraft docking application described by Ho and McClamroch uses computer vision and a rhombus target to determine relative pose [17]. The algorithm used is point based where the four corners of the target are extracted. A nonlinear, iterative least squared error solution is given. Focal length is assumed known. These values are fed to observers that estimate the spacecraft rotational and translational velocities. Control loops maintain position and attitude while keeping the rhombus mark in view of the camera. Simulation results show that orientation accuracy is better than position accuracy with the largest error in the range estimate. However, this error is proportional to the range so that as range decreases during the docking maneuver the error also decreases to an acceptable level.

Holt and Netravali give a theoretical discussion of the pose determination problem given N corresponding points on an object whose coordinates are known in some 3–D

reference frame and the 2–D perspective projection onto an image plane [3]. If N is three, it is shown that, in general, there can be up to four solutions unless the points are collinear; in that case, there can be infinitely many solutions. The case of four coplanar points is examined for various combinations of object and image point collinearity conditions. Four noncoplanar corresponding points are shown to have, at most, four solutions.

An analytic solution to the perspective four-point pose determination problem is given by Horaud et al. [18]. The general case of noncoplanar is considered, with special cases of four coplanar points and four points forming a right vertex. While a detailed derivation is shown, no experimental results are provided. Howard and Book [19] describe a video-based sensor intended for automated docking systems. This system was developed for the National Aeronautics and Space Administration (NASA) with intended use on Space Station Freedom and other satellite servicing vehicles. A solid state video camera, digitizer, and special target form the basis of the system. The relative pose between the camera and the target is determined. A three-point correspondence that has three circles of retroreflective tape spaced in a line is determined from the target. The middle reflector containing the other two circles is mounted on a pole spaced away from the target plane. An algebraic method is used to calculate the pose from the image locations. No proof is given that the pose calculated is unique since as many as four solutions are possible with only three point correspondences.

A robot location determination method is given by Hung et al. [20] for use in mobile navigation. Multiple marks are proposed in which a trapezium is used as a target along with a mark identification. A modification of Fischler and Bolles' method [9] is used to calculate the pose based on obtaining the corner points of the trapezium to subpixel accuracy. Point correspondence and identification are addressed in detail.

Hung et al. [21] propose a method for pose determination based on a planar quadrangle target. From knowledge of the dimensions of the target, the relative pose of the

camera coordinate system is found. The technique directly solves for the positions of the vertices of the quadrangle relative to the camera perspective center. While the solution is exact, the algorithm is sensitive to noise. To reduce noise effects, a shape restoration stage is given that attempts to restore the noise-perturbed shape of the quadrangle to its original shape. This stage is a constrained optimization problem in which a suitable cost function is defined and then minimized iteratively. Results show relative errors of less than 1% for a noise standard deviation of 0.5 pixel.

Another pose determination method using four coplanar points is given by Kamata et al. [22]. Intermediate transformations are defined when relating world coordinates to camera coordinates. These intermediate transformations simplify the coordinate systems so that algebraic reasoning can be used to solve for the remaining transformation. Lens focal length is required. The full solution is a direct closed-form method.

Kite and Magee describe an algorithm for pose determination of a camera relative to a rectangular target of known size [23]. The 3–D problem is broken down into two 2–D problems. In addition, only three corners of the target must be visible. A major restriction in the use of this algorithm is that the optical axis of the camera must pass through the centroid of the target. Noise effects are not considered. An iterative algorithm for pose determination of a rigid body from landmark points is given by Krishnan et al. [24]. These points may be coplanar or noncoplanar. Any number of points greater than three may be used. This method iteratively calculates the coordinates of the N landmark points relative to the camera perspective center using least squared error for the overdetermined system. The pose transformation algorithm of Veldpaus et al. [25] is used once the 3–D coordinates of the individual points are known. Computer simulations were conducted using four to 12 points in coplanar arrangements with additive Gaussian noise having a standard deviation of one pixel. Two metrics that permit comparison between results from different methods are defined. These are the Range-Size Ratio (RSR) and the Focal-Length-to-Noise Ratio

(FNR). A significant result is that an increased number of landmarks does not always result in increased accuracy of pose parameters. Range errors were found to be less than 0.6% for planar targets in which the plane of the target was at least 20 degrees off normal to the camera. Results obtained were compared with Hung et al., Yuan, and Abidi and Chandra, all using four points [21, 1, 4]. While the range and angle errors given were lower than the other methods, the FNR of 800 was somewhat higher than that of Yuan (735) and Abidi and Chandra (520), which would be expected to produce lower errors.

Mandel and Duffie address errors in mobile robot docking [26]. A method is proposed to compensate for the docking errors. Two methods are described for measuring the error: (1) a touch trigger probe, or (2) a vision system. The reference target for the vision system is four points drawn on a cube with three points on one side and the fourth on an adjacent side. Four nonlinear equations are used to solve for the depth of the points. The transformation matrix is then derived through a least squared error approach by means of the singular value decomposition. This approach calculates a true rigid body transformation of a rotation and a translation. Experimental results using real hardware showed depth and angular errors of about 0.3%. Liu et al. describe methods of pose determination using point correspondences and line correspondences [27]. Both nonlinear and linear solutions are given in which the rotation and translation are calculated separately. Three line or point correspondences are needed for the nonlinear iterative method while eight line or six point correspondences are needed for the linear technique. Simulation results are given for each case. For line correspondence with the nonlinear method and using four lines with a 5% level of uniformly distributed noise, the relative error for rotation is 0.0086 and for translation is 0.0347. The relative errors for the linear method are 0.2832 and 0.0992, respectively, which are considerably worse. The algorithm for point correspondence forms lines from the points and uses the same algorithm as the line matching method. A video approach to autonomous docking for spacecraft is given

by Tietz and Germann [28]. The paper describes a complete guidance and control system with a TV camera on the chase vehicle. A target consisting of three lights arranged in a "T" pattern with the middle light in front of the object is used. Flashing lights are used to determine point correspondence. Equations are given for the pose calculation, but no derivation is given nor is there a justification for the use of only a three-point target so that a unique solution may be achieved.

Jatko et al. [29] describe a method for autonomous docking in a battlefield resupply application. A target consisting of six LEDs arranged as two quadrangular faces is used with a single camera for image acquisition. The LEDs flash at a preset frequency with temporal bandpass filtering being performed on the images. A least squared error iterative algorithm similar to Lowe was used for pose calculation [30]. The method was successfully implemented and demonstrated on a full-scale robot arm. Robustness to missing features as well as extraneous features is a key result. Goddard et al. [31], in a related application, use the four coplanar point method of [8] to measure the pose of each face of the six-point target. The point correspondence of the quadrilateral faces is used to determine which face is being viewed. Experimental results are given for relative error in all six parameters.

Wolfe et al. provides an analysis of the constraints provided by point correspondences for monocular pose determination [32]. The constraints imposed by different numbers of corresponding points are examined for a perspective camera model. Coordinate curves are used to trace the motion of a point from the camera reference as a position parameter is varied. For the case of four points, it is shown that knowledge that the points are in the shape of a rectangle is sufficient to recover the orientation without the known length and width. Position is recoverable when these parameters are also known. Pinkney et al. [33] and Pinkney and Perratt [34] describe a real-time photogrammetry system that is a part of the remote manipulator system used on the NASA Space Shuttle. Based on a TV image, the system calculates pose at frame rates through a point-based algorithm. A

least squared error fit to the pose value is performed on the individual frame values every n frames. The points are extracted from quadrangular or three-point targets.

Yuan presents a general method of pose determination or, as it is known in photogrammetry, the exterior orientation calibration problem. The method is based on a single image of known feature points [1]. Instead of the collinearity constraint, the algebraic structure of the transformation is used in deriving a nonlinear method for any number of points. A proof of the existence of a solution for n points is given as well as a proof of the uniqueness of a solution for the case of four noncollinear coplanar points. Specific solutions are given for the cases of three, four coplanar, four noncoplanar, and five noncoplanar corresponding points. Simulation results for coplanar and noncoplanar four-point solutions show sensitivity to variations in range, sensitivity to image-size ratio, and sensitivity to errors in image and feature point coordinates. Errors in position are relatively independent of range while coplanar points tend to give higher orientation errors than noncoplanar when the object plane is parallel to the image plane. A significant result is that noncoplanar targets give higher accuracies and have greater robustness than coplanar targets. No problems were encountered with multiple solutions for four noncoplanar points. The conditions under which multiple solutions may arise are not given, however.

1.3.3 Pose Using Model-Based and Higher-Level Primitives Methods

Model-based methods of pose determination use an explicit model for the geometry of the object in addition to its image in determining the pose. The object is modeled in terms of points, lines, curves, planar surfaces, or quadric surfaces. Some methods obtain these features from computer aided design models [35]. Other papers restrict or simplify the model so that the object can be described easily and the computation involved is reasonable [36]. As in point-based methods, a perspective vision model is generally assumed. Orthographic models with scaling that simplify the computation have

also been reported, but the accuracy is limited at closer ranges due to the perspective approximation [7]. For this class, methods have also been developed for single vision, stereo, and range images. From the 2–D image in single-vision techniques, which is the perspective projection of the known 3–D object, the problem is to determine the rotation and translation of the object that would give rise to the given 2–D projection. Edges and vertices must be recognized and matched with corresponding features in the object for the pose to be calculated. One important example is the calculation of the pose from the image of a circular feature (an ellipse) using a closed-form method [37]. An ellipse is fit to the feature edge through minimization of error criteria. Two solutions result from one image, however, so that an additional feature or an additional image is required to derive the correct pose.

Methods using range images are predominately model-based. The range image is 3–D rather than 2–D as with intensity images so that depth information is directly available for matching features to the object. Algorithms, however, are generally iterative and are based on nonlinear minimization of a cost function relating the image features with the object features [38]. Hypotheses and verification may also be performed to determine the correct pose [39]. A method has also been developed based on neural networks to solve the problem using Kohonen-Nets [40]. Stereo methods, by determining the 3–D coordinates of features through matching, are similar to range image methods [41]. Detailed summaries of methods in this class are given below. Chen describes a line-to-plane pose determination method in which the image data are lines that correspond to planar features on an object [42]. Point-to-point correspondence is not assumed. One example of its application is the projection of a light stripe across a cube generating three lines on three faces of the cube. The method shown is based on quaternions requiring the solution of an eighth-degree polynomial. No initial guess is required. A minimum of three lines is required, but the solution is not unique. Experience indicates that one

or two additional lines are required to determine a unique solution, but the uniqueness of solutions is not addressed. Closed-form solutions are given for commonly encountered special configurations. An analysis of the necessary and sufficient conditions for existence is given. Noise sensitivity simulations that show the effects on orientation were presented. Significant errors were seen through perturbations of the three line vectors. Choosing a triplet of lines is recommended as the initial step in an iterative solution using additional feature pairs and least squared error techniques. The author did not find a direct solution using additional feature pairs.

Chen and Tsai propose a solution for robot localization using an assumed shape for a target object [36]. The object must have a polygon-shaped top and a lateral surface perpendicular to the top with all dimensions of the polygon top and angles known. This class of objects includes cubes, most buildings, and many machine parts. Pencils are defined for a corner of the polygon from which a corresponding image pencil is extracted. Experimental results show errors of 5% or less with sensitivity to critical orientation angles. In comparison with the work by Hung et al. [21], the method has improved accuracy for general orientations but is worse at the critical angles.

Three-dimensional planar objects are considered by Consales et al. for determination of spatial orientation [43]. Assuming perspective projection, the method extracts the image junctions of object vertices where three edges come together. The paper shows that only two junction angles are required to calculate the orientation instead of three as described in previous work.

A triplet of image lines is used by Dhome et al. for pose determination of 3–D objects from a single image [44]. The lines are assumed to be the perspective projection of three ridges on the object model with correct matching applied. A search is performed for the model attitude consistent with image projections. The required transformations are calculated from an eighth-degree polynomial. Since eight solutions are possible, logical

rules are given that reduce the number of solutions. In actual use, matching of the model to the image lines must be performed in a general prediction-verification procedure. Lowe's method [30] was used for matching while the proposed method is used at the beginning of the procedure to reduce the searching required and to provide an initial value for Lowe's iterative procedure.

Ellis et al. gives a method for fitting ellipses to edge-based image data [45]. The ellipse parameters are used to determine the pose of the object. A particular emphasis is the handling of uncertainty in which Kalman filtering is used in extracting the ellipse data and used in the model matching. The uncertainty is propagated throughout the matching stages through a Mahalanobis distance measure and a Kalman filter. An orthographic projection is used to simplify the calculations. Results show the detection and grouping of ellipses in real images with many different types of structures. No data is given on the accuracy of fit or on the accuracy of pose determination using this method. Ellipse fitting is used by Foster and Sanderson to determine object orientation [46]. This method assumes objects with circular edges or surfaces and calculates the orientation based on the ellipse equation. An error measure based on the minimization of distance that is used to iteratively fit edge points to the ellipse equation is defined.

Safaee-Rad et al. describe a method to estimate the parameters of a quadratic curve from a single image. The emphasis is on accurate estimation of ellipse parameters. These ellipse parameters are then used for pose determination [47, 48, 49]. This work gives an exact closed-form solution for calculating relative pose from circular object features. Since the curve parameters are used, the method does not require individual point matching. No restrictions are placed on the optical axis direction, and perspective projection is assumed. The method is based on geometric reasoning and algebraic transformations to first calculate the relative orientation without knowledge of the radius. Lens focal length is required. Since the circular feature has rotational symmetry, only two orientation angles

can be determined. The third is arbitrary and can be set to zero. If the radius is known, the center coordinates can be determined directly. If the radius is unknown, two images are required with prescribed viewpoint differences. As with any method using circles alone as the determining feature, the mirror ambiguity gives two possible orientations for the viewed circle. Also given is an extended method for 3–D quadratic features in which pose is estimated for spherical features. Simulated and real experimental results are given. Combining the ellipse parameter finding method given in [48] and this method, the real object tests show average orientation angle errors of 0.80 degrees or less and average positional errors of 1.28 mm or less with the largest error in the depth axis.

Ferri et al. examine pose estimation through perspective inversion for four different problems [50]. These problems are: (1) four coplanar line segments; (2) three orthogonal line segments; (3) a circular arc; and (4) a quadric of revolution. For case two, the method gives a second-degree equation as the solution. Case three for elliptical images finds the orientation of planes that intersect with a cone whose vertex is the focal point. Two possible solutions are possible for each image ellipse. When the circle radius is known, the relative position of the center can be found. Experimental results are given for case three. The ellipse parameters are derived from image edge points. Errors are given as 3.7% for angles and 3.0% for center positions. The ellipse backprojection gave more robust results than the surface of revolution method especially in the presence of occluding boundaries.

Glachet et al. provide a method to determine the pose of an object of revolution from a single image and the model of the object [51]. The method first computes the perspective projection of the axis of the object based on the geometric properties of the object of revolution. This is performed through a prediction-verification scheme that computes a virtual image in which the contours are symmetrical. Experimental results show graphical images, but no error results are reported.

Goldberg and Lowe give a method for verification of 3–D model parameters given a single image [52]. This stage is performed after hypothesis generation in a model matching method for pose determination. A nonlinear least squared error solution is presented that minimizes the distance between the candidate model object with the projected pose and the corresponding image features. New work includes the definition of a coordinate tree data structure that consists of transformations between components of the model. This tree can also consist of null components that represent transformations without bodies. An advantage of this structure is the ease of calculating derivatives of points with respect to variable parameterization. No experimental results are given. A simplified method is given by Han and Rhee to calculate the six pose parameters given a special circular target [53]. This target consists of a circle with a center dot and a dot near the inside edge of the circle. A restrictive assumption is that the optical axis must past through the center dot. Each pose parameter is then calculated separately. Experimental results using this method as a camera calibration for two stereo cameras show average errors of less than one percent. No explicit noise evaluation is given.

Haralick describes a pose determination method from the projection of a conic or polygon of unknown size [54]. The technique decomposes the problem into two three-parameter problems if the shape and size of the curve are known. Exact point correspondence is not required. Conics considered are the ellipse, parabola, and hyperbola. It is assumed that the optical axis of the camera passes through the center of the image and that the focal length is known. The orientation is first determined as an optimization problem with three unknowns. The translation is then calculated. The same approach can be used for noncoplanar curves as long as the curve can be placed in parametric form. In addition, a disadvantage of the iterative optimization is that initial values must be supplied that may not converge. No experimental results are given. Haralick in [55] further shows that the relative orientation may be calculated from the perspective projection of a

rectangular target whose size and position are unknown. Given the size, then the relative position may be determined. The equations are relatively simple and are derived geometrically to give a closed-form solution. An ambiguity is present in determining which side of the rectangle the camera is on.

A standard pattern is proposed by Kabuka and Arenas that is used for mobile robot position measurement using a single image [56]. This pattern, which is composed of a circle and bar codes, when viewed from any position, will permit the extraction of the relative pose from a projected image. The circle used in the target gives rise to an ellipse in the image, and other marks determine the rotational ambiguity. Parameters of this image ellipse are used to derive the pose parameters. A method is given for calculating the size, position, and orientation of the ellipse based on position-invariant moments. The derivation of the pose from these parameters is also shown. Analysis of error sources show that assumptions made in the derivation give rise to errors. One example is that the optical axis passes through the center of the circle; another is that the swing angle is zero. Experimental results are given with errors for the pan angle and the distance values. As an extension of the above work, Hussain and Kabuka [57] further analyze the errors associated with this method. The error sources considered are discretization of the image, segmentation of pixels, and perspective distortion in which it is stated that the projection of the circular target onto the image plane is not a true ellipse. Architectures for real-time moment generation up to seventh order are also proposed.

Lowe gives a method for object pose determination through model matching with hypothesis and verification [30, 58]. Model features are initially matched to image features, and an error function is defined that computes the error for each image axis between the projected image features and the observed image features. This function is the sum of the products of the partial derivatives of the pose parameters times the pose correction values. Rotations are represented by quaternions to simplify the computations. The

Newton-Raphson method is applied to the error function to iteratively solve for the pose correction values. If convergence is not obtained, an incorrect match is indicated. This method is extended to parametric models so that both pose and model parameters are solved for during the iterative algorithm. Line-to-line correspondences may also be used in addition to point-to-point. An advantage of this approach is that depth information is not explicitly required. No quantitative results are given on the accuracies obtained with this method.

Neira et al. outline a method for validation of geometric relations as part of an overall object recognition scheme [59]. Early estimation of object location is required during the model matching. Uncertainty is incorporated in a symmetries and perturbation model. Geometric features are classified into one of six transformation subgroups. Data fusion of observations is performed through an IEKF. No details are given on this integration method. Solutions to the validation of geometric relations are given for all combinations of symmetries. A means for verification is also given. The approach, "recognizing while locating," enables unary and binary constraints to validate the minimum number of observations that can give an object location hypothesis. Additional features are then used to verify this hypothesis. The methodology for selecting the initial set of observations is given. Kriegman et al. give constraints for recognizing and locating curved 3–D objects from single image features [60]. The features considered are those dependent on the viewpoint of the curved object and are not intrinsic to the object model. The object points are related to the image points by a series of n equations and n unknowns that are the coordinates of the model points. An example is shown, but the general method is compute intensive.

A method of determining pose based on real or virtual 3–D lines in a target scene is described by Kumar [61, 62]. Correspondence is achieved between the target lines and the image lines without endpoint correspondence. The application is for a mobile

robot in an outdoor environment where the features are distant from the camera. An uncertainty measure is developed relating the variance in the solution to the noise in the input parameters. This measure assumes a perfect 3–D model with noise occurring only in the image data. The method used is based on the constraints given by Liu et al. [27] with a nonlinear optimization technique proposed by Horn [63] to solve the relative orientation problem. While algorithms for line correspondences are given, point algorithms have also been developed. Two basic methods are presented in which one first calculates rotation and then translation while the second calculates rotation and translation at the same time. The simultaneous solution generally gives the best results. Closed-form solutions are given for the variance of the rotation and translation as a function of the noise variance of the image data. Simulated and real experiments were performed using the developed method. Results were good, but possible improvements are noted along with a needed increase in robustness due to mismatching of lines.

Linnainmaa et al. describe an approach for pose determination using triples of points on an object matched to triples of points in an image [64]. These image triples form a triangle that are junctions connected by edges or contours. A generalized Hough transform is used to evaluate which object triangles correspond to the image triangles and to evaluate relative pose. Further geometric constraints are applied to the clusters to improve the matching. Multiple solutions from the three-point pose extraction are not a problem since the clustering and subsequent constraints eliminate all but the correct values. Experimental results are given that show good results based on large numbers of triangle pairs.

A method that estimates pose through model feature matching is given by Ray [65]. One or more monocular images are used to extract line features from an object that are matched to model features through an interpretation tree. A cost function is defined for the interpretation tree branches that assumes that the approximate pose is known for

the object. After correspondence is established, an optimization function produces the maximum likelihood pose estimate based on all feature matches. Experimental results show higher accuracy using multiple views where position errors were about 0.1 inch and 3 degrees in rotation.

Roth et al. gives another model-based approach for pose correction of a mobile robot [66]. In combination with dead reckoning, this method matches model feature points with image data points. A reverse projection of points to 3–D space is used to find the pose correction vector for rotation and translation. As a preprocessing step, the orientation is first estimated by means of a vanishing point technique. This measurement is used to restrict the search space for feature matching. No experimental results are given.

Sheu and Bond describe a method to solve for the object pose from a single image based on a hierarchy of features [67]. This method is an extension of previous work in model matching. These features are primitives, generalized, and compound. Primitive features include points, lines, and ellipses. A generalized feature is a grouping of primitive features; a compound feature combines generalized features. Ellipse-to-ellipse correspondence is a significant contribution that permits a reduction in the complexity of the inverse projection problem. The systematic handling of the feature matching is also significant and provides a robust method.

1.3.4 Methods Incorporating Noise Estimation and Uncertainty

The methods described in this section incorporate noise modeling or estimation as an integral part of the location determination solution. Uncertainty is explicitly estimated or predicted based on input noise statistics and on the given algorithm.

In presenting a new data fusion technique, Abidi [68] gives a one-dimensional example of the fusion of stereo and ultrasonic range data. Stereo triangulation for position measurement, while good for the transverse coordinates, x and y, is not as accurate for

depth measurement, z. An ultrasonic sensor, however, has good depth accuracy but poor x, y resolution. A quadrangular target is used in which the four corner point coordinates are found. For both the stereo images and the range data, probability distribution functions are computed for the corner point locations in x, y, and z. Data from one coordinate at a time is fused to provide the output results.

Ayache and Faugeras provide a method for representing uncertainty in measurements of points, lines, and planes that are used in building visual maps of the environment for a mobile robot [69]. Two images are taken at each robot location as a stereo pair. The extended Kalman filter (EKF) is used as a model for estimating the position of these primitives and the uncertainty through the covariance matrix. The feature location noise from the image is modeled as zero mean Gaussian noise with a known variance. As the robot moves, additional images are taken and are used to update the filter and to improve the estimate of the 3–D locations. The solution for rotation and translation of relative frames for images taken during movement is similarly modeled. Multiple primitives are matched for both the first and second positions of the robot. Each matched pair is used as an input to the filter that provides an estimate of the relative displacement and the error. Questions are raised as to whether the modeling is sufficient for lines and planes and whether the assumption of Gaussian distributions is valid for actual situations. Posterior marginal pose estimation is described by Wells in which a probabilistic model of image features along with prior estimates of feature matches and pose is used to derive a pose objective function [70]. The pose is then found from the optimization of this objective function. Parameters of the probabilistic model are obtained from the image. Noise from image features is assumed to be independent with a Gaussian distribution. A linear projection model is assumed. The objective function is shown to have a relatively simple form under these assumptions. Experimental results are given for synthetic-range

imagery. The objective function has a sharp peak near the correct pose while local maxima are also present.

Scene depth and image motion are computed by a method given by Singh that measures image flow from time-varying images [71]. Kalman filtering is used to calculate the depth using image flow. An advantage is that the covariance matrices are available that provide an estimate of the error as subsequent images are acquired. The filtering technique is based on one proposed by Matthies et al. [72] that estimates depth and depth uncertainty at each pixel as opposed to other techniques based on image features. Correlation techniques are used to measure the flow as well as to estimate the uncertainty. A discussion and comparison of both the feature-based method and the proposed method are given with quantitative measurements. The convergence rate and accuracies obtained are similar. Experimental results are given for lateral motion of the camera.

Fischler and Bolles propose a method called random sample consensus (RANSAC) for fitting a model to experimental data in which the data may contain a significant percentage of gross errors [9]. This method is applied to the perspective n point pose determination problem which is also solved as a part of this paper. The minimum number of points for a finite number of solutions is shown to be three, and a closed-form algorithm is given based on geometric reasoning. Examples of multiple solutions are given for three, four noncoplanar, and five noncoplanar point sets. Four coplanar points, and six points in a general position, are shown to give a unique solution. The RANSAC procedure is applied to a set of overconstrained points and initially starts with as small a data set as possible. The set is expanded iteratively including additional points that do not vary more than a threshold from an initial model. Points greater than the threshold are considered outliers and are not used in the pose determination. Least squared error techniques may be applied to the remaining points. Rationale for selecting the threshold and the maximum number of iterations is given. Implementation details and experimental results are given

for simulated problems and for determining camera pose from an aerial image. The method removed all gross errors from each trial.

1.3.5 Methods Using Kalman Filtering for Direct Pose and Motion Estimation

Estimation methods and applications using Kalman filtering are described in this section. The problems being solved include not only pose estimation but also motion and 3–D structure of a rigid object. Some of these are "Structure from Motion" problems where *a priori* knowledge of the object is not available. These methods use point features, generally, and require a large number of features per image to solve for the many state variables. Batch nonlinear optimization methods are also discussed [73] where estimates are made by analyzing an entire sequence of images at once. For real-time applications such as robotic control, visual servoing, etc., batch analysis is not a feasible option. Extended Kalman filtering is the most widely used method of recursive estimation.

A recursive method of estimating motion, structure, and focal length is given by Azarbayejani and Pentland [74]. A sequence of 2–D images from which feature points are extracted is used as input. An EKF is used for estimation. The camera model uses the image plane as the origin to aid in parameterizing the imaging geometry. A single parameter, the depth, provides the estimate of the state for each structure point in contrast to other approaches that use three parameters per point [75, 73]. This parameterization is claimed to be more stable for recursive estimation than previous methods. However, recovery of the 3–D x and y point coordinates is subject to the noise present in the image features and could result in significant errors. Quaternions are used to represent rotation indirectly. The three rotational velocities are used in the state vector while a global rotation state is maintained between iterations but is not used in the EKF. Simulated, as well as experiments on actual image sequences, are given along with the performance.

Broida et al. in [75] also describe a motion and structure estimation problem and a solution based on the IEKF. Point-based image features are inputs with the 3–D coordinates estimated using three state variables per point. The camera-referenced object center x and y, translation velocity, and the 3–D point coordinates are state variables normalized to the object center depth. A rotational quaternion and angular velocity complete the state estimate. A normalization step on the quaternion estimate is performed immediately after the measurement update to constrain the extra degree of freedom. Initial estimates for the filter are obtained from a batch method applied to a set of images. Experimental results are given for both simulated as well as real image sequences. For the simulated case, uniformly distributed noise (to mimic quantization error) is added to the image points. Only four points from the vertices of a cube are processed. Convergence is obtained with 10% noise and initial estimates within about 20% of the true values.

A Kalman filter approach is used by Lee and Kay for 3–D pose and motion estimation from stereo images [76, 77]. Consecutive object frames are used as input to estimate the orientation from which the position is calculated with object rotations and translation velocities constant. The Kalman filter state is expressed as a rotation using quaternions for linearization. Measurement noise is derived from 2–D images rather than the 3–D feature points that give a more accurate noise estimate when the object distance from the camera varies over a wide range. For each stereo pair, the orientation is first estimated through a least squared error method at each measurement sample. The Kalman filter is then applied over the time sequence of estimates. Simulation results considerably improved estimates over the individual measurements.

Westmore and Wilson describe the use of an EKF to provide an estimate of the position of a camera mounted on a robot end point [78]. This filter is implemented as part of the measurement portion of a closed-loop position control for a six-degree-of-freedom robot arm. Only three of the six pose parameters, however, are estimated due to the

constraint that the object motion be limited to a 2–D surface. Two feature points are extracted from the image and used as inputs to the filter. An implementation and results are given in which the update rate is 61 Hz. Wang and Wilson describe the estimation of both relative position and orientation of a moving object using an EKF [79]. Feature points from the object are extracted from the image and are provided to the Kalman filter. Using a minimum of three object points, an estimate for the six-element pose vector is obtained. However, five noncoplanar points were found to give better estimates. Motion is assumed smooth enough so that the first derivatives of the pose parameters are constant. Any deviation from this assumption appears as a disturbance to the state model. Both simulation and experimental results are given. An implementation is also described that is similar to that of Westmore and Wilson [78]. Tracking errors under simulation were found to be within 0.6 mm. in depth and within 0.4 degree for pitch and yaw. The measurement error covariance matrix was initialized based on image point location variance while the disturbance covariance matrix was initialized based on known motion trajectories. The Kalman filter output estimate variances were a factor of three to four less than the measurement noise variances, indicating the increased level of confidence in the estimates. For the real results, the tracking errors were not as good as the simulated ones. However, error was attributed to the forward kinematic calculations for determining the reference positions. The filter output error variance was much less than the forward kinematics error variance.

An algorithm for estimating depth from known camera motion is given by Matthies and Kanade [72]. The algorithm, suitable for parallel processing, uses Horn's optical flow method [80] and a Kalman filter to estimate depth at each image pixel location. Lateral translation is shown to give the highest sensitivity to the depth measurement. A comparison is made to a Kalman filter approach using edge locations. Performance is shown to be almost equal to the feature-based approach. Experimental results with real

images show that the algorithm performs well. Burl uses a parallel version of the EKF to estimate depth from sequential high noise images containing a moving object [81]. The filter is used to estimate Fourier coefficients as well as velocity of the object. A reduced order filter is obtained by truncating the Fourier coefficients.

In the area of robot arm control using vision, Allen et al. [82] describe a robotic system for tracking and grasping a moving object. The imaging system consists of two cameras and computes optical flow using triangulation to determine the 3–D position. A standard Kalman filter is used to predict the position from the 3–D measurements. Experimental results are given showing the robot arm successfully picking up a moving toy train. The train motion is approximately 2–D. Papanikolopoulos et al. also describe a system for tracking a moving object [83] with a single camera mounted on a robot arm. This system, similarly, uses optical flow and a sum of squared differences for matching between image frames. An object moving in a plane is tracked where depth to the object is given. Various control techniques are also given including proportional-integral (PI), pole assignment, and linear quadratic Gaussian. Both steady state, time invariant, and time-varying Kalman filters are used to estimate the transformation state.

Wilson et al. [84] provide a method for visual servoing from a robot end effector. Point-based visual servoing is used along with an EKF to estimate the 3–D pose of a workpiece object using a single camera. Knowledge of the object is required for the estimate. The EKF estimates the six-element pose vector (three translations and three angles) and the six first derivatives of the pose vector. Manipulator control is achieved through specification of joint angle coordinates. The transformation from the end effector frame to the joint coordinate frame is given. A proportional-derivative control is used in the actual implementation. Experimental results are given for pose estimation and for visual servoing. Accuracy is estimated by relative displacement calculated from the kinematics solution. Robustness of the pose estimate is also shown by randomly inserting

missing features into the measurement. The missing features are handled in the measurement equations by increasing the variance of the corresponding term of the measurement error covariance matrix R. Only small perturbations were observed in the state estimate when up to two features (out of a total of five) were occluded.

1.4 Synopsis of Dissertation

Chapter 2 of this dissertation provides background theory on elements needed for the development of the proposed pose estimation theory. These elements include the basic theory of 3–D transformations, perspective projection, quaternions, dual numbers, estimation theory, and extended Kalman filtering. In Chapter 3, the theoretical approach to the method is developed along with the new theory and analysis needed to fully explain the operation. A rationale is given for the use of the IEKF as an estimator. This area includes the projection of 3–D lines to the 2–D image plane, the formulation of the IEKF with the dual quaternion transformation, and analysis of error estimates. Chapter 4 provides an analysis of the noise statistics of the line-point parameters in comparison with point noise. This includes derivation of the probability density function in terms of the line end point density functions and the derivation of mean, variance, and covariance of the line points. Chapter 5 gives the system implementation in a robotic arm application in which a single camera is mounted on the end effector of the robot. Both relative pose and motion are estimated between the camera and a known object using the method described in Chapter 3. Results are also given for structure estimation when the object geometry is unknown. Experimental results are given for simulated testing as well as testing on an actual robot arm. Simulation results are given for the errors obtained under different types of initial estimates. The method is compared to an IEKF estimation method based on point features. Analysis of the advantages of the proposed method is given. Chapter 6

summarizes the results and presents the conclusions obtained from this work. Suggestions for future directions are also given.

Chapter 2

Theoretical Foundation

This chapter develops the theoretical foundation for the pose estimation method presented in Chapter 3. The areas covered include coordinate transformations, quaternions, dual numbers, estimation theory, and Kalman filtering theory.

2.1 Coordinate Frame Transformations

Coordinate frame transformations between orthogonal 3-D references can be represented in several ways. The most common representation is the two-step process consisting of a displacement of the origin of one reference frame with respect to the other followed by a rotation around three angles. The rotation is given by a 3-by-3 orthonormal matrix R while the displacement is a three-element vector $\vec{d}$. The coordinate transformation is then given by

$$\vec{x}_c = R\vec{x}_w + \vec{d}. \tag{2.1}$$

$\vec{x}_c$ and $\vec{x}_w$ are three-element vectors consisting of the 3–D point coordinates in the form $[x\ y\ z]^T$.

A homogeneous 4-by-4 matrix T, defined in terms of R and $\vec{d}$,

$$T = \begin{bmatrix} R & \vec{d} \\ \vec{0}^T & 1 \end{bmatrix}, \tag{2.2}$$

can be applied to the transformation as follows:

$$\vec{y}_c = T\vec{y}_w. \tag{2.3}$$

In this case, $\vec{y}_c$ and $\vec{y}_w$ are four-element vectors consisting of the 3-D point coordinates in the form $[x\ y\ z\ 1]^T$.

2.2 Quaternions

A quaternion is a four-component number consisting of a scalar part and three orthogonal parts. Quaternions are an extension of complex numbers to R^4 with three imaginary parts. Formally, a quaternion q can be defined as

$$q = q_0 + q_1\vec{i} + q_2\vec{j} + q_3\vec{k} \tag{2.4}$$

where each of the q_i is a real number, and $\vec{i}$, $\vec{j}$, and $\vec{k}$ are orthogonal imaginary unit vectors. From the above definition, the class of quaternions can be seen to include scalars, ordinary complex numbers, and three-element spatial vectors. The unit vectors have the following properties:

$$\vec{i}^2 = -1, \vec{j}^2 = -1, \vec{k}^2 = -1;\ and \tag{2.5}$$

$$\vec{i}\vec{j} = \vec{k}, \vec{j}\vec{k} = \vec{i}, \vec{k}\vec{i} = \vec{j}. \tag{2.6}$$

A geometric interpretation of a quaternion is given below in Section 2.6. Addition and subtraction of quaternions are straightforward and can be performed element by element to form a result that is also a quaternion. These operations are associative and commutative. Multiplication by a scalar real number is likewise closed. The quaternion of all zeros is a zero element over addition. An additive inverse for each quaternion also exists.

Quaternions, therefore, are a group that forms a vector space over the real numbers. An equivalent representation for quaternions is a column vector $[q_0\ q_1\ q_2\ q_3]^T$. Multiplication of the two quaternions s and t is defined as

$$\begin{aligned} st &= (s_0t_0 - s_1t_1 - s_2t_2 - s_3t_3) + \vec{i}(s_0t_1 + s_1t_0 + s_2t_3 - s_3t_2) \\ &\quad + \vec{j}(s_0t_2 - s_1t_3 + s_2t_0 + s_3t_1) + \vec{k}(s_0t_3 + s_1t_2 - s_2t_1 + s_3t_0). \end{aligned} \tag{2.7}$$

Quaternion multiplication is associative and distributive although it is not commutative. The unit element quaternion is $1 + \vec{i}0 + \vec{j}0 + \vec{k}0$. A multiplicative inverse element exists for all quaternions except for the zero element so that the quaternion group forms a division algebra. Matrix forms of quaternion multiplication are given below where quaternions are expressed as four element vectors:

$$pq = \begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix} q = \overset{+}{M}_p\, q \tag{2.8}$$

$$qp = \begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & p_3 & -p_2 \\ p_2 & -p_3 & p_0 & p_1 \\ p_3 & p_2 & -p_1 & p_0 \end{bmatrix} q = \overset{-}{M}_p\, q. \tag{2.9}$$

The 4-by-4 matrices $\overset{+}{M}_p$ and $\overset{-}{M}_p$ are identical except that the lower right 3-by-3 submatrix is transposed. The conjugate of a quaternion is

$$q^* = q_0 - \vec{i}q_1 - \vec{j}q_2 - \vec{k}q_3. \tag{2.10}$$

The square of the magnitude of a quaternion is then defined as

$$\|q\|^2 = qq^* = q_0^2 + q_1^2 + q_2^2 + q_3^2 = q \cdot q. \tag{2.11}$$

The multiplicative inverse element of quaternion q is then

$$q^{-1} = (1/||q||^2)q^*. \tag{2.12}$$

Consider the set of quaternions with unit magnitude. Unit quaternions form a subgroup over multiplication where the inverse is simply the conjugate. The quaternion matrices $\overset{+}{M}_p$ and $\overset{-}{M}_p$ are orthogonal when the quaternion is a unit quaternion. Quaternions are useful in describing 3-D rotations as discussed below in Section 2.4. For additional properties, see [63, 85].

2.3 Dual Numbers

A dual number is defined as [11]

$$d = a + \epsilon b \tag{2.13}$$

where a, b are real numbers and ϵ is defined as $\epsilon^2 = 0$. In the following, a will be referred to as the real part and b as the dual part. Addition is defined as

$$d1 + d2 = (a1 + \epsilon b1) + (a2 + \epsilon b2) = a1 + a2 + \epsilon(b1 + b2), \tag{2.14}$$

and multiplication is defined as

$$d1d2 = (a1 + \epsilon b1)(a2 + \epsilon b2) = a1a2 + \epsilon(a1b2 + b1a2). \tag{2.15}$$

These operations are closed as well as associative and commutative. Multiplication, as shown above, distributes over addition. Addition has an identity element, the number zero. From these properties, it follows that dual numbers form an abelian group under addition. A multiplicative identity exists, but not every dual number has a multiplicative inverse; i.e., dual numbers with zero real parts. Thus, dual numbers form an abelian ring under multiplication and division. The conjugate of a dual number is

$$d^* = a - \epsilon b. \tag{2.16}$$

The product of a dual number and its conjugate is

$$dd^* = a^2 \tag{2.17}$$

while the modulus of a dual number is simply the real part,

$$|d| = a. \tag{2.18}$$

Note that the modulus can be negative. The Taylor series expansion of a dual function about its real part has a particularly simple form:

$$f(a + \epsilon b) = f(a) + \epsilon b f'. \tag{2.19}$$

This property is useful in the expansion of commonly used functions. For example,

$$\sin(\hat{\theta}) = \sin(\theta) + \epsilon d \cos(\theta) \tag{2.20}$$

where $\hat{\theta} = \theta + \epsilon d$.

Dual numbers were first proposed by Clifford [86] and developed by Study [87]. Study represented the dual angle of two skew lines in 3-D space as a dual number. The dual angle is

$$\hat{\theta} = \theta + \epsilon d \tag{2.21}$$

where θ is the angle between the line vectors, and d is the shortest distance between the lines.

2.4 Dual Number Quaternions

The dual number concept may be applied to quaternions as well as vectors and matrices. As described below, a dual quaternion can be used to represent a general 3-D transformation. The dual-number quaternion is defined as

$$\hat{q} = r + \epsilon s \tag{2.22}$$

where r and s are each quaternions [11]. Addition and multiplication are defined in a similar manner to real quaternions. Analogous to the real quaternion, dual-number quaternions form a vector space over the dual numbers and likewise form an associative algebra. The conjugate of a dual quaternion is defined as

$$\hat{q}^* = r^* + \epsilon s^*. \tag{2.23}$$

The squared magnitude of a dual quaternion is

$$\|\hat{q}\|^2 = \hat{q}\hat{q}^* = rr^* + \epsilon(rs^* + sr^*). \tag{2.24}$$

The squared magnitude is a dual number with non-negative real part. If the real part is not zero, then an inverse exists and is

$$\hat{q}^{-1} = \hat{q}^* / \|\hat{q}\|^2. \tag{2.25}$$

A unit dual quaternion is defined as one whose magnitude is $1 + \epsilon 0$. Unit dual quaternions form a subgroup over multiplication with the inverse equal to the conjugate. The conditions for a unit dual quaternion are $rr^* = 1$ and $rs^* + sr^* = 0$. These conditions imply that

$$\vec{r} \cdot \vec{r} = 1, \vec{r} \cdot \vec{s} = 0. \tag{2.26}$$

2.5 3-D to 2-D Perspective Projection

The problem of pose determination is shown in Figure 2.1 along with the separate 3-D reference frames for the target object and camera. A perspective projection model, commonly referred to as the pinhole lens model, is used where the lens center is the camera reference origin. The process of 3-D to 2-D projection is considered. The first step is to transform the object coordinates to the camera reference. Next, the x and y coordinates of the projected object onto the image plane are found. These relations are, given image coordinates $(x_i \ y_i)$ and camera coordinates $(x_c \ y_c \ z_c)$,

$$x_i = x_c \frac{\lambda}{z_c}, \ and \tag{2.27}$$

$$y_i = y_c \frac{\lambda}{z_c} \tag{2.28}$$

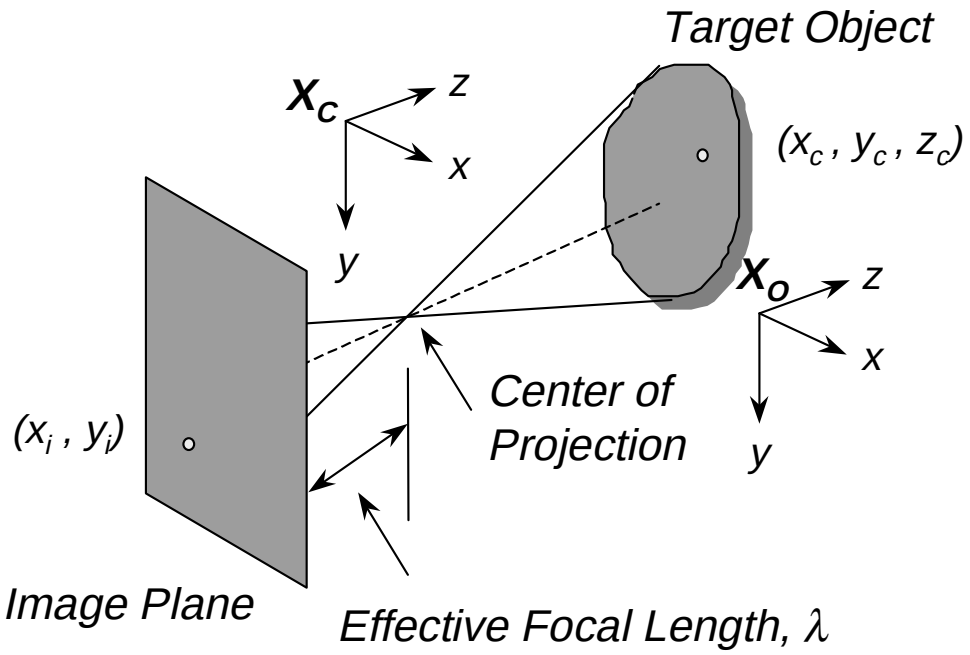

Fig. 2.1. 3–D to 2–D perspective projection model assumed in pose estimation problem. The center of projection is the camera reference origin.

where λ is the effective focal length [88]. As can be seen, the transformation is nonlinear and noninvertible. That is, the relations are unique for 3-D to 2-D, but the 3-D coordinates cannot be determined uniquely from the 2-D image coordinates.

From one intensity image of the target, the 2-D feature coordinates in the image plane can be extracted. These coordinates, along with the geometric model of the target and the intrinsic camera parameters, provide the necessary information for pose calculation. Camera calibration performed off-line prior to the pose measurement is used to determine the intrinsic camera parameters such as pixel size, focal length, optical center, and lens distortion. The relation between the 3-D camera coordinates and the 2-D image is based on the assumed perspective projection vision model.

While the calculation of the image coordinates is straightforward given the 3-D coordinates and the camera parameters, the inverse operation of determining the 3-D coordinates from the 2-D image plane coordinates is more difficult since the correspondence of 2-D to 3-D coordinates is not unique.

2.6 Representation of 3-D Rotation by Quaternions

Quaternions can be used to represent 3-D rotations. The unit quaternion $q = [q_0 \ \vec{q}]$ may be placed in the form

$$q = \cos\frac{\theta}{2} + (\sin\frac{\theta}{2})\vec{u} \tag{2.29}$$

where $\cos\frac{\theta}{2} = q_0$, $\sin\frac{\theta}{2} = \sqrt{\vec{q}\cdot\vec{q}}$, and $\vec{u} = \frac{\vec{q}}{\sqrt{\vec{q}\cdot\vec{q}}}$ when $\vec{q}\cdot\vec{q}$ is not zero [85]. This expression describes the relationship between the quaternion elements and a rotation in 3-D. The angle-axis representation of rotation is applicable here since an axis of rotation is described along with the magnitude of rotation. In this case, θ represents the magnitude of rotation about an axis $\vec{u}$. The composite product of quaternions given by

$$r' = qrq^* \tag{2.30}$$

describes the rotational transformation from r to r'. r and r' are quaternions whose scalar element is zero and whose imaginary part is the 3-D coordinate of a point in the two reference frames. q is a unit quaternion. The 3-by-3 rotation matrix R defined in terms of a unit quaternion is

$$R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}. \tag{2.31}$$

Composition of rotations is easily realized through multiplication of quaternions as shown by

$$r' = (pq)r(pq)^*. \tag{2.32}$$

Horn [63] notes that fewer arithmetic operations are required to multiply two quaternions than are required to multiply two 3-by-3 rotation matrices. Matrix multiplication of orthonormal matrices does not necessarily result in an orthonormal matrix due to finite precision arithmetic. The quaternion resulting from a product of quaternions, however, can easily be normalized to obtain a unit magnitude. It is not straightforward to find the nearest orthonormal matrix from the product of rotation matrices.

2.7 Representation of 3–D Rotation and Translation by Dual Number Quaternions

Quaternions are limited in the sense that only rotation is represented in a full 3–D transformation. Translation must be dealt with separately. Dual-number quaternions, however, provide a framework that may be used to represent both rotation and translation. To describe the relationship of dual-number quaternions to the 3–D transformation, an explanation of the screw representation of a transformation is required. The screw transformation is shown pictorially in Figure 2.2.

The required parameters include the screw axis, the screw angle, and the screw pitch. The screw axis is described by a line in 3–D space that has direction $\vec{l}$ passing through point $\vec{p}$. A 3–D line is defined by the two three-element vectors $\vec{l}$ and $\vec{m}$ where $\vec{m} = \vec{p} \times \vec{l}$. While six parameters are used in this definition, only four degrees of freedom are present due to the constraints $\vec{l} \cdot \vec{m} = 0$ and $\|\vec{l}\| = 1$ [89]. This particular formulation leads to the dual quaternion representation of the line,

$$\hat{\vec{l}} = \vec{l} + \epsilon \vec{m}. \tag{2.33}$$

The screw transformation then requires eight parameters for its representation. Walker [11] and Daniilidis [89] have shown that a unit dual quaternion $\hat{q} = r + \epsilon s$ representing the 3-D transformation can be expressed as

$$\hat{q} = \begin{bmatrix} \cos(\hat{\theta}/2) \\ \sin(\hat{\theta}/2)\hat{\vec{l}} \end{bmatrix} \tag{2.34}$$

where $\hat{\vec{l}}$ is a dual vector that represents the screw axis about which the coordinate system has rotated and translated as given by $\hat{\theta}$. An alternate form of a unit dual quaternion is

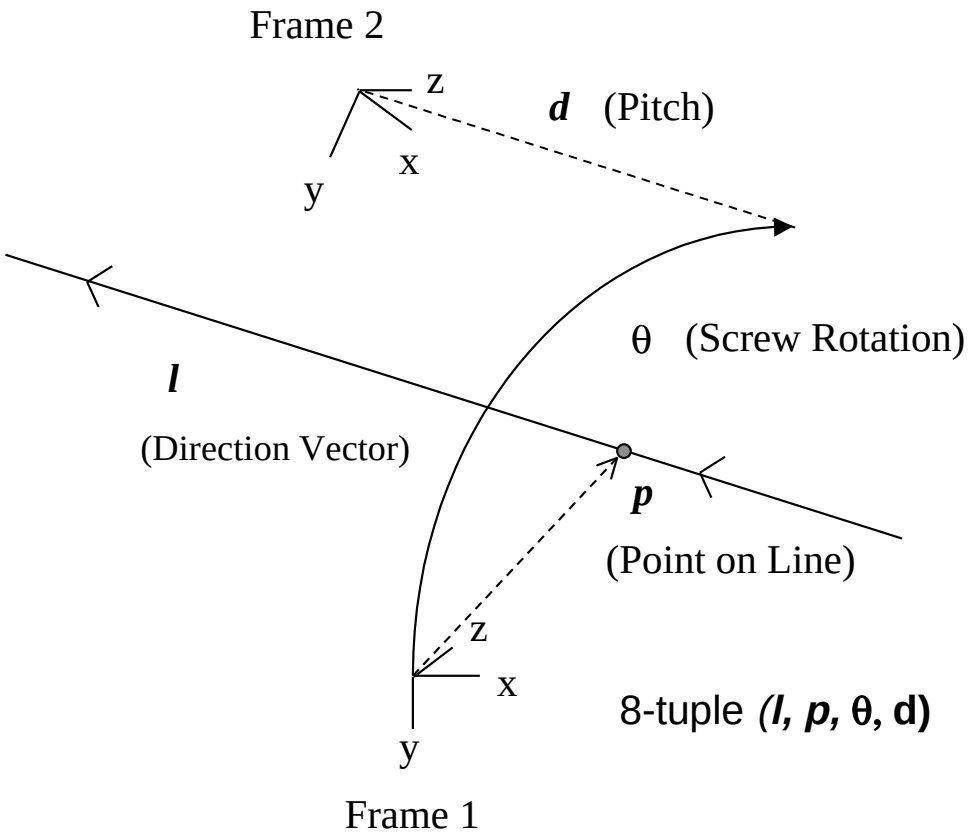

Fig. 2.2. Diagram showing screw form of 3–D transformation between two reference frames. Eight parameters are needed to specify the complete transformation.

$$\hat{q} = q + \epsilon\frac{t}{2}q. \tag{2.35}$$

q, in this form, is a real unit quaternion representing only rotation while t is a vector quaternion representing translation.

The resulting 3-D transformation of a line can be expressed simply as

$$\hat{l}' = \hat{q}\hat{l}\hat{q}^*. \tag{2.36}$$

The form of this equation is identical to that of Equation 2.30 for real quaternions. Compositions of tranformations are likewise similar,

$$\hat{l}' = \hat{p}\hat{q}\hat{l}\hat{q}^*\hat{p}^*. \tag{2.37}$$

An expression for the dual quaternion $\hat{q}$ can also be represented in terms of a dual angle that is similar to the angular form of the standard quaternion

$$\hat{q} = \begin{bmatrix} \cos(\frac{\hat{\theta}}{2}) \\ \sin(\frac{\hat{\theta}}{2})\hat{\vec{n}} \end{bmatrix} \tag{2.38}$$

where $\hat{\vec{n}}$ is a dual vector representation of a 3-D line and $\hat{\theta}$ is a dual angle.

The unit dual quaternion transforms lines from one coordinate frame to another. In comparison, the real quaternion transforms points from one frame to another that differ by only a finite rotation.

2.8 Iterated Extended Kalman Filter

A dynamic process or system may be defined by a set of state variables, some or all of which cannot be measured directly. Available measurements are a function of the state

variables and may be noisy. The process itself may not be deterministic. The Kalman filter, first proposed by R.E. Kalman in 1960 [90], is the minimum variance Bayes' estimate of the state variables of a linear system model. It is the optimum in a least squares sense of a linear system with additive Gaussian noise in both the process and the measurement. In addition, it is the optimal linear filter for all other noise distributions [91, 92]. The standard Kalman filter assumes a linear system model as well as a linear measurement model. The linear system model is described by

$$\vec{x}_k = \Phi_{k-1}\vec{x}_{k-1} + \vec{w}_{k-1} \tag{2.39}$$

where $\vec{x}_k$ is the state at sample k; $\vec{x}_{k-1}$ and $\vec{w}_{k-1}$, respectively, are the states and process noise at sample $k-1$. The process noise, $\vec{w}_{k-1}$, is assumed to be white, zero-mean Gaussian distribution with covariance matrix Q_k at sequence number k. Φ_{k-1} is the state transition matrix at sample k. The linear measurement model is given by

$$\vec{z}_k = H_k\vec{x}_k + \vec{v}_k \tag{2.40}$$

where $\vec{z}_k$ is the measurement vector at sample k. $\vec{v}_k$ is the measurement noise, also assumed white, zero-mean Gaussian with covariance matrix R_k. The filter provides estimates of the state and the error covariance matrix associated with the estimate. The state estimate extrapolation from one sample to the next is

$$\tilde{x}_k(-) = \Phi_{k-1}\tilde{x}_{k-1}(+). \tag{2.41}$$

$\tilde{x}_k$ is used to denote the estimate of x_k. The $(-)$ and $(+)$ notations represent the estimate before and after the measurement update, respectively. The Kalman filter also estimates

the state error through the error covariance matrix P_k. The error covariance extrapolation is given by

$$P_k(-) = \Phi_{k-1} P_{k-1}(+) \Phi_{k-1}^T + Q_{k-1}. \tag{2.42}$$

In addition to the extrapolation equations that incorporate the dynamic model, the filter incorporates the measurement information through the use of the measurement update equations. The state estimate update equation is

$$\tilde{x}_k(+) = \tilde{x}_k(-) + K_k(\vec{z}_k - H_k \tilde{x}_k(-)). \tag{2.43}$$

K_k is called the filter gain and is given by

$$K_k = P_k(-) H_k^T (H_k P_k(-) H_k^T + R_k)^{-1}. \tag{2.44}$$

To complete the Kalman filter definition, the error covariance update equation is

$$P_k(+) = (I - K_k H_k) P_k(-). \tag{2.45}$$

Initial estimates for the state and error covariances are $\tilde{x}_0 = E[\vec{x}(0)]$ and $P_0 = E[(\vec{x}(0) - \tilde{x}_0)(\vec{x}(0) - \tilde{x}_0)^T]$. The measurement and process noises are assumed uncorrelated, i.e., $E[\vec{w}_i^T \vec{v}_j] = 0$ for all i, j.

To apply the Kalman filtering technique to nonlinear dynamic processes and to nonlinear measurement models, extensions to the standard method have been developed. The EKF and IEKF linearize the nonlinear function about the state estimate using the linear terms of a Taylor series expansion about the current estimate. Considering the EKF first, the nonlinear system model has the form

$$\vec{x}_k = \phi_{k-1}(\vec{x}_{k-1}) + \vec{w}_{k-1}. \tag{2.46}$$

The next state is assumed to be an explicit nonlinear function of the present state with additive process noise. The nonlinear measurement model is

$$\vec{z}_k = h_k(\vec{x}_k) + \vec{v}_k \tag{2.47}$$

with the measurements being an explicit nonlinear function of the present state and with additive measurement noise. The linear term of the Taylor series expansion of the state transition function is found from

$$\Phi_k(\tilde{x}_k(-)) = \left.\frac{\partial \phi_k(\vec{x})}{\partial \vec{x}}\right|_{\vec{x}=\tilde{x}_k(-)} \tag{2.48}$$

while the linear measurement expression is given by

$$H_k(\tilde{x}_k(-)) = \left.\frac{\partial h(\vec{x})}{\partial \vec{x}}\right|_{\vec{x}=\tilde{x}_k(-).} \tag{2.49}$$

The state estimate extrapolation uses the nonlinear function in predicting the next state,

$$\tilde{x}_k(-) = \phi_{k-1}(\tilde{x}_{k-1}(+)), \tag{2.50}$$

and the nonlinear measurement function is used in calculating the state update,

$$\tilde{x}_k(+) = \tilde{x}_k(-) + K_k(\vec{z}_h - h_k(\tilde{x}_k(-))). \tag{2.51}$$

However, the computation of K_k requires the linearized term H_k in an equation that is identical to the standard Kalman filter,

$$K_k = P_k(-)H_k^T(H_kP_k(-)H_k^T + R_k)^{-1}. \tag{2.52}$$

The term is also used in the error covariance update equation

$$P_k(+) = (I - K_kH_k)P_k(-) \tag{2.53}$$

while the covariance extrapolation requires the linearized Φ_k that is also the same as the regular Kalman filter,

$$P_k(-) = \Phi_{k-1}P_{k-1}(+)\Phi_{k-1}^T + Q_{k-1}. \tag{2.54}$$

The initial conditions are the same as the standard filter. An implicit assumption is that the functions ϕ and h are differentiable over all $\vec{x}$. The transition matrix Φ and the measurement matrix H are linearized about the latest state estimate. As a result, K_k and P_k are random variables that depend on the sequence of state estimates, unlike the standard filter for which these variables may be computed prior to any measurements. P_k is, thus, not the true covariance matrix but is only an approximation. Figure 2.3 shows, in block diagram form, the functionality of the EKF. The inputs consist of the measurements, the measurement noise covariance, and the process noise covariance; the outputs are the next state estimate and the error covariance matrix. The diagram shows the filter operation as a feedback control loop where the summing junction minimizes the error between the measurements and the estimated measurements.

In the IEKF, the updated state estimate $\tilde{x}_k(+)$ is used repeatedly as a new linearization point to obtain new estimates of $\tilde{x}_k(+)$, K_k, and $P_k(+)$. The iterative expressions for these updated variables are

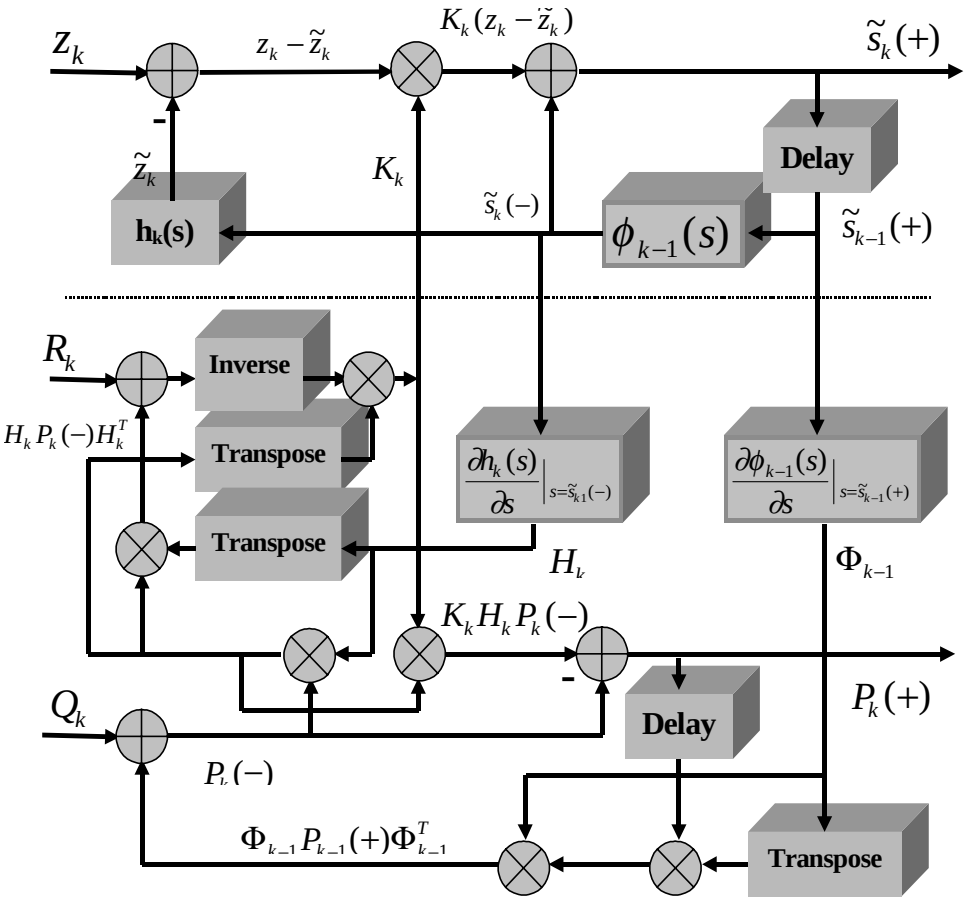

Fig. 2.3. Extended Kalman filter block diagram showing functionality along with the inputs and outputs. The filter is seen to have a closed-loop feedback control where the measurement error is minimized.

$$\begin{aligned}
\tilde{x}_{k,i+1}(+) &= \tilde{x}_k(-) + K_{k,i}(\vec{z}_k - h_k(\tilde{x}_{k,i}(+)) - H_k(\tilde{x}_{k,i}(+))(\tilde{x}_k(-) - \\
&\quad -\tilde{x}_{k,i}(+))) \qquad (2.55)\\
K_{k,i} &= P_k(-)H_k^T(\tilde{x}_{k,i}(+))(H_k(\tilde{x}_{k,i}(+))P_k(-)H_k^T(\tilde{x}_{k,i}(+)) + \\
&\quad +R_k)^{-1} \qquad (2.56)\\
P_{k,i+1}(+) &= (I - K_{k,i}H_k(\tilde{x}_{k,i}(+)))P_k(-) \qquad (2.57)\\
\tilde{x}_{k,0}(+) &= \tilde{x}_k(-) \qquad (2.58)
\end{aligned}$$

where i denotes the iteration number. The state estimate will converge after a finite number of iterations. The result, in general, is a more accurate estimate of the state at the expense of additional computational cost.

Chapter 3

Pose and Motion Estimation Method

This chapter develops the method for pose and motion estimation from monocular camera images. The perspective projection equations for points from 3–D to 2–D are shown in Chapter 2 to be nonlinear. Calculation of relative position and orientation from the 3–D coordinates of corresponding points in two reference frames as shown by [9], [63], and [4] is also nonlinear. For the minimum three points needed to calculate a pose, multiple solutions are possible. Solutions with four or more points are overconstrained with a minimum error criterion used to determine the result. Motion estimation requires a dynamic system model that, as will be shown, is nonlinear in the rotational dynamics. Direct linear estimation techniques, such as the standard Kalman filter, are not applicable.

The problem examined in this chapter is: estimation of the 3–D pose and motion of an object with respect to a camera. Two-dimensional images of the object are available from the camera at regular time intervals. The object and the camera, or both, may be moving with respect to a fixed reference frame. The object is well defined, of known geometry and size, and has a sufficient number of point and line features visible from a particular view angle from which the pose estimation can be uniquely determined. The approach to solving this estimation problem is given below. An extension to this theory is also provided to estimate object structure where the object geometry is initially unknown.

3.1 Pose and Motion Estimation Using IEKF and Dual Number Quaternions

The pose and motion estimation method is based on the IEKF and uses a dual quaternion representation of a general 3-D transformation. The IEKF is applicable to this problem since the dynamic state is highly nonlinear as is the measurement function. Use of the dual quaternion representation provides a consistent, uniform treatment of both rotational and translational states.

3.2 State Assignment

The state assignment for this problem provides an estimate of the pose or transformation between the camera and the object reference frames and the first derivatives of this transformation. The assignment is based on the dual quaternion representation of the 3-D transformation as described in Chapter 2. Since the dual quaternion has a real part and a dual part, both of which are quaternions, a direct implementation of this structure would require eight variables with an additional eight for the first derivatives. However, since there are only six degrees of freedom in the 3-D transformation and since the filter computation time is proportional to the number of state variables, fewer variables are desirable. Based on the approach given by Broida [75], the state variable assignment is

$$\vec{s} = \begin{bmatrix} t_x & t_y & t_z & q_0 & q_1 & q_2 & q_3 & v_x & v_y & v_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^T . \tag{3.1}$$

Thirteen variables are used: t_i and v_i terms are the linear translation and linear velocity, respectively; q_i is the rotational quaternion in each axis; and ω_i is the rotational velocity in each axis. Translation, rather than the dual part of the dual quaternion, is part

of the state estimate since the dual part can readily be calculated from the translation and the rotational real quaternion. This relation is given by

$$\hat{q} = q + \epsilon \frac{t}{2} q \tag{3.2}$$

where t is a vector quaternion formed from the translation vector and by setting the scalar part to zero. The first derivative is

$$\dot{\hat{q}} = \dot{q} + \epsilon(\frac{\dot{t}}{2} q + \frac{t}{2} \dot{q}). \tag{3.3}$$

Thus, the first derivative of the dual quaternion $\dot{q}$ can easily be computed from the linear velocity and the other known state variables. Also, with the linear velocity as state variables, the state update for the linear translation becomes very simple. The first derivative of the translation is also continuous for all real dynamic translations.

Chou [85] gives the relation between quaternion angular velocity and the spatial angular velocity,

$$\Omega = \begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = 2\dot{q}q^*. \tag{3.4}$$

Ω is a vector quaternion where the vector portion is the angular velocity about the axes. Solving for $\dot{q}$ gives

$$\dot{q} = \frac{1}{2} \Omega q. \tag{3.5}$$

This equation shows that from an estimate Ω of angular velocity, the derivative of the rotational quaternion may be easily determined. In a real physical system, the angular velocity is continuous; from this equation, the quaternion derivative is also continuous.

While a minimal state variable representation of rotation using three rotation angles instead of the four needed by a quaternion is possible, several difficulties arise. A finite rotation range is used for each angle, generally $[-\pi, \pi)$ or $[0, 2\pi)$. A discontinuity exists at the point where the wraparound occurs, i.e., the value becomes 0 at a 2π rotation angle. The quaternion would also have to be calculated from the rotation angles. Additional computation, including the evaluation of several trigonometric functions, negates the advantage of reducing the number of state variables by one. However, since the quaternion has four parameters, an additional degree of freedom is present. As a result, normalization of the quaternion to unit magnitude after each iteration is performed.

3.3 System Model

A functional block diagram of the overall system is shown in Figure 3.1. This diagram shows the basic operations involved in the pose estimation problem. A system model will be defined to fit the discrete Kalman filter formulation. That is, a function ϕ and additive noise $\vec{w}$ are required that can be used to calculate the next state in terms of the present state as

$$\vec{s}_k = \phi_{k-1}(\vec{s}_{k-1}) + \vec{w}_{k-1}. \tag{3.6}$$

$\vec{w}$ is assumed to be additive Gaussian noise with zero mean and covariance matrix Q_{k-1}. The state transition function ϕ extrapolates from the state at time interval $k-1$ to the next state at time interval k. The linear and angular velocities are assumed constant so that

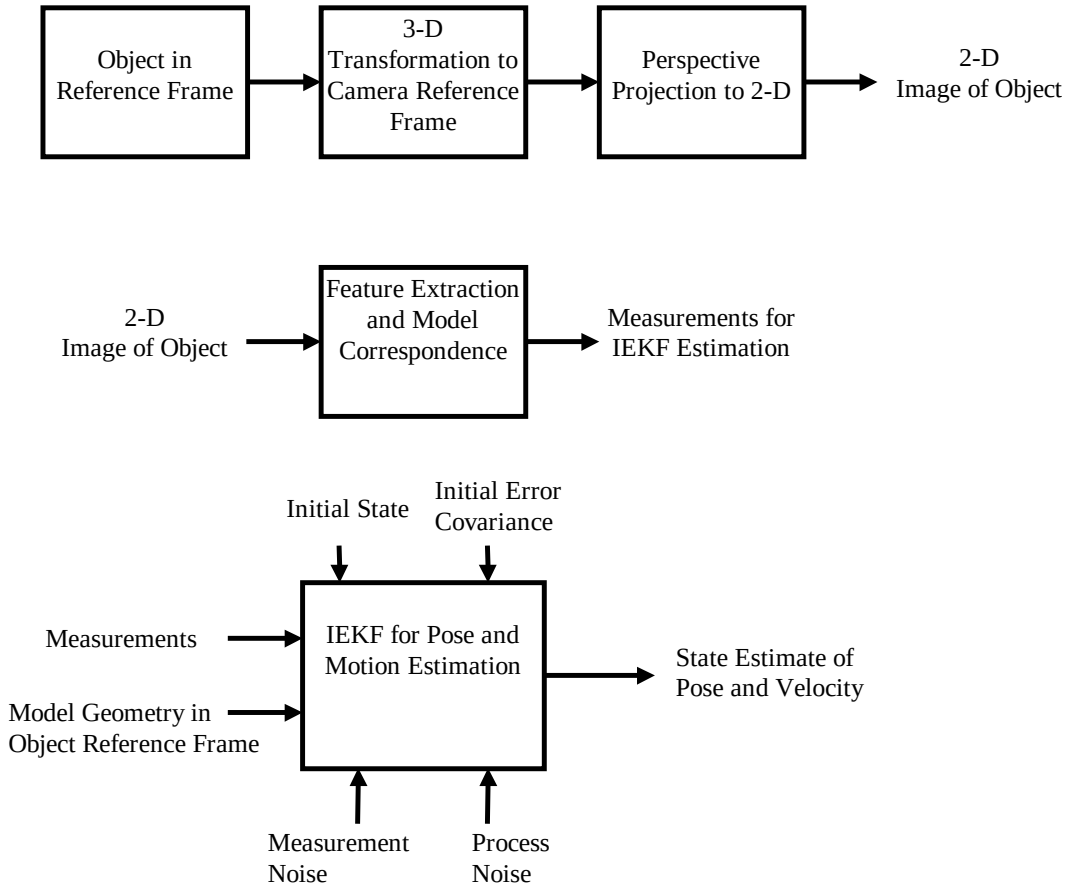

Fig. 3.1. Functional block diagram of estimation problem showing operations involved in pose estimation method from optical imaging to state estimate produced by the iterated extended Kalman filter.

$\omega_i(t_k) = \omega_i(t_{k-1})$ and $v_i(t_k) = v_i(t_{k-1})$. Translation updates are also straightforward if the velocity between time intervals is assumed constant.

The quaternion propagation in time must be derived. The differential equation is given in Equation 3.5 where quaternion multiplication is applied. An alternate form where Ω is expanded to its equivalent 4-by-4 matrix is

$$\dot{q} = \frac{1}{2}\begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & -\omega_z & \omega_y \\ \omega_y & \omega_z & 0 & -\omega_x \\ \omega_z & -\omega_y & \omega_x & 0 \end{bmatrix} q = \bar{\Omega} q. \tag{3.7}$$

The solution when all ω_i are constant is

$$q(t) = e^{\bar{\Omega}(t-t_k)} q(t_k), \tag{3.8}$$

which is similar to that given in [75]. Since $\frac{2\bar{\Omega}}{\|\Omega\|}$ is an orthogonal matrix and a skew-symmetric matrix, the eigenvalues are $\pm i, \pm i$ [93]. After solving for the closed form of the matrix exponential [94], the solution, when simplified, is

$$q(t) = \left[\cos(\frac{\|\Omega\|(t-t_k)}{2})I + \frac{2}{\|\Omega\|}\sin(\frac{\|\Omega\|(t-t_k)}{2})\bar{\Omega}\right] q(t_k). \tag{3.9}$$

The state transitions for each variable have now been defined. The state transition function is

$$\phi_k(\vec{s}) = \begin{bmatrix} t_x + \tau v_x \\ t_y + \tau v_y \\ t_z + \tau v_z \\ Q_{tran} q(t_k) \\ v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{3.10}$$

where

$$Q_{tran} = \left[\cos(\frac{\|\Omega\|(\tau)}{2}) I + \frac{2}{\|\Omega\|} \sin(\frac{\|\Omega\|\tau)}{2}) \bar{\Omega} \right]. \tag{3.11}$$

Q_{tran} is the 4-by-4 quaternion transition matrix.

3.4 Measurement Model

The measurement update equation is

$$\vec{z}_k = h_k(\vec{s}_k) + \vec{v}_k. \tag{3.12}$$

The measurement noise $\vec{v}_k$ is a zero mean, Gaussian sequence, with covariance matrix R_k. h_k comprises the 3-D transformation and perspective projection functions shown in Figure 3.1. Since the dual quaternion operation transforms lines to lines, the given model features from the object are lines represented as dual vector quaternions

$$\hat{l}_i = l_i + \epsilon m_i \tag{3.13}$$

where $\vec{l_i}$ is the unit direction vector of the line and $\vec{m}_i$ is defined as $\vec{p_i} \times \vec{l_i}$, with $\vec{p_i}$ being the coordinates of a point $\vec{p}$ on the line i.

The minimum number of model lines is four for coplanar lines and six for noncoplanar lines to calculate a consistent unique pose. The following equation is used to calculate each transformed line:

$$\hat{l}'_i = \hat{q}\hat{l}_i\hat{q}^*. \tag{3.14}$$

Perspective projection is then applied to the transformed lines. The projected line lies in a plane defined by the 3-D line and the center of projection. This plane is described by the equation

$$m_x x + m_y y + m_z z = 0 \tag{3.15}$$

that intersects the image plane at $z = -\lambda$. The result is an equation of the projected line in the $z = -\lambda$ plane,

$$m_x x_i + m_y y_i = m_z \lambda, \tag{3.16}$$

where x_i and y_i are the image plane coordinates. Placing this equation in a normalized form results in

$$\frac{m_x x_i}{\sqrt{m_x^2 + m_y^2}} + \frac{m_y y_i}{\sqrt{m_x^2 + m_y^2}} = \frac{m_z \lambda}{\sqrt{m_x^2 + m_y^2}}. \tag{3.17}$$

From this equation, the direction vector of the image line is

$$\vec{l}_i = \begin{bmatrix} -\frac{m_y}{\sqrt{m_x^2+m_y^2}} \\ \frac{m_x}{\sqrt{m_x^2+m_y^2}} \\ 0 \end{bmatrix}. \tag{3.18}$$

$\left|\frac{m_z\lambda}{\sqrt{m_x^2+m_y^2}}\right|$ is the minimum distance from the origin to the line. Since the image plane is located at $z = -\lambda$, the distance from the perspective center is

$$d_{pc} = \sqrt{\left(\frac{m_z\lambda}{\sqrt{m_x^2+m_y^2}}\right)^2 + \lambda^2} = \frac{\lambda|\vec{m}|}{\sqrt{m_x^2+m_y^2}}. \tag{3.19}$$

Since $\vec{m}_i = \vec{p}_i \times \vec{l}_i$ where $\vec{p}_i$ is a point on the image line and $\vec{m}_i$ defines the same plane as $\vec{m}$,

$$\vec{m}_i = d_{pc}\frac{\vec{m}}{|\vec{m}|} = \frac{\lambda}{\sqrt{m_x^2+m_y^2}}\vec{m}. \tag{3.20}$$

The projected image line is now defined by a dual vector as

$$\hat{\vec{l}}_i = \vec{l}_i + \epsilon\vec{m}_i. \tag{3.21}$$

3.5 Representation of Lines in a Plane

A rationale for investigating lines as features as opposed to points or higher-order curves is that planar surfaces are commonly found on manmade objects, particularly in indoor environments. Intersection of these surfaces then results in 3-D lines. These issues are discussed by Talluri in [95]. The image of an indoor environment containing

these surfaces results in linear features in the image plane that can be extracted in a straightforward procedure.

The result of the perspective projection from the 3-D lines is a set of dual vector quaternion coplanar lines located in the image plane. Six parameters are required to represent each of these lines. Since the lines are restricted to a known plane, however, only two degrees of freedom are present. A format is also needed to compare these transformed lines with line features measured from the acquired images. The format used for representing these lines is an x, y point called the line point. A line point is defined as the intersection of the line feature with a line passing through the image origin that is perpendicular to the line feature. Figure 3.2 illustrates the definition of the line point on the image plane. The line point is unique for all lines except for those lines that pass through the origin. For these cases, it is assumed the lines approach the origin with a distance δ but do not actually pass through the origin. The line point has the advantage of minimum state representation and a simple distance measure as well as being continuous for all lines. Other line representations such as slope, intercept or ρ, θ have discontinuities that cause considerable difficulties when partial derivatives are taken in calculating the IEKF operating point.

The line point is calculated from the dual vector image line as

$$x_{lp} = l_{iy} m_{iz} \tag{3.22}$$

$$y_{lp} = -l_{ix} m_{iz}. \tag{3.23}$$

In terms of the 3-D line dual vector components,

$$x_{lp} = \lambda \frac{m_x m_z}{m_x^2 + m_y^2} \tag{3.24}$$

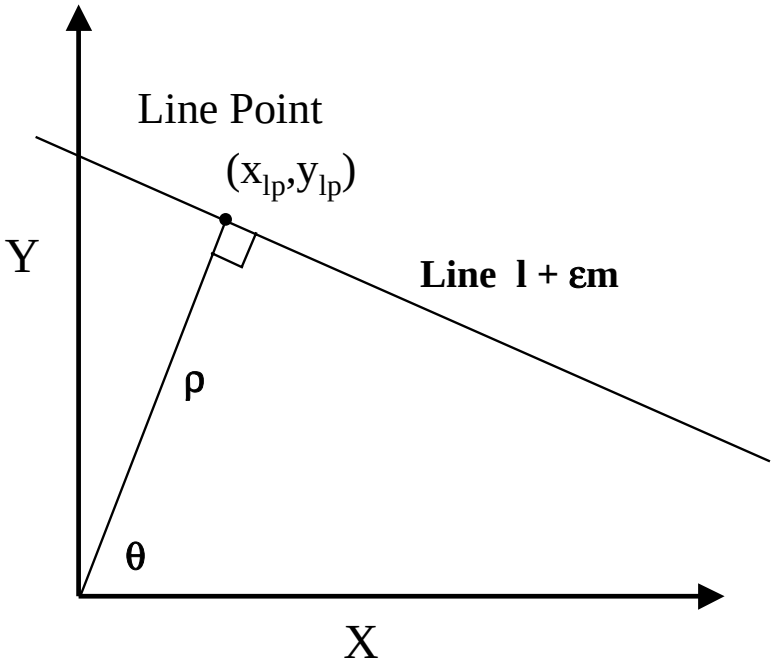

Fig. 3.2. Line point for a line in 2-D image plane is defined as the intersection point of the line and a perpendicular line passing through the origin.

$$y_{lp} = \lambda \frac{m_y m_z}{m_x^2 + m_y^2}. \tag{3.25}$$

Line features from the acquired image are likewise placed in this representation as measured variables.

3.6 Linearization of the State Transition Function

The linearized state transition matrix is computed as the partial derivative of the state transition function with respect to each state variable and evaluated at time $t = t_k$. The time dependency requires that it be computed at each state update. Formally, Φ_k is determined as

$$\Phi_k(\tilde{x}_k(-)) = \left. \frac{\partial \phi_k(\vec{x})}{\partial \vec{x}} \right|_{\vec{x}=\tilde{x}_k(-).} \tag{3.26}$$

Φ_k is a matrix of dimension $n \times n$ where n is the number of state variables. In the problem being addressed here, $n = 13$. This matrix is

$$\Phi_k = \begin{bmatrix} 1 & 0 & 0 & \cdots & \tau & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & \tau & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & \tau & 0 & 0 & 0 \\ 0 & 0 & 0 & & 0 & 0 & 0 & & & \\ \vdots & & & Q_{tran} & & & \vdots & & Q_\omega & \\ 0 & \cdots & & & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \cdots & & & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & \cdots & & & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & \cdots & & & & & \cdots & 1 & 0 & 0 \\ 0 & \cdots & & & & & \cdots & 0 & 1 & 0 \\ 0 & \cdots & & & & & \cdots & 0 & 0 & 1 \end{bmatrix}. \tag{3.27}$$

Q_{tran} is defined above in Equation 3.11. Q_ω is defined as

$$Q_\omega = \frac{\partial Q_{tran} q(t_k)}{\partial \vec{\omega}}\Big|_{\vec{\omega}=\vec{\omega}_k} \tag{3.28}$$

where $\omega = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T$. Q_ω is a 3-by-4 matrix. Each row may be computed separately as given below. The partial derivative of $Q_{tran} q(t_k)$ with respect to each ω_i is

$$\begin{aligned} \frac{\partial Q_{tran} q(t_k)}{\partial \omega_i} &= \left(\frac{-\omega_i \tau}{2|\omega|} \sin(\frac{|\omega|\tau}{2}) + \left(\frac{-\omega_i \tau}{|\omega|^2} \cos(\frac{|\omega|\tau}{2}) - \frac{\omega_i}{|\omega|^3} \sin(\frac{|\omega|\tau}{2}) \right) \bar{\Omega} + \right. \\ &\quad \left. + \frac{1}{|\omega|} \sin(\frac{|\omega|\tau}{2}) \frac{\partial \bar{\Omega}}{\partial \omega_i} \right) q(t_k). \end{aligned} \tag{3.29}$$

The partial derivatives of $\bar{\Omega}$ with respect to ω_x, ω_y, and ω_z are

$$\frac{\partial \bar{\Omega}}{\partial \omega_x} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.30}$$

$$\frac{\partial \bar{\Omega}}{\partial \omega_y} = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} \tag{3.31}$$

$$\frac{\partial \bar{\Omega}}{\partial \omega_z} = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}. \tag{3.32}$$

3.7 Linearization of the Measurement Function

The linearized measurement matrix is the partial derivative of the measurement function with respect to each state variable and evaluated at time $t = t_k$. Thus, it is time dependent and must be calculated at each measurement update step. Formally, H_k is determined as

$$H_k(\tilde{s}_k(-)) = \left.\frac{\partial h(\vec{s})}{\partial \vec{s}}\right|_{\vec{s}=\tilde{S}_k(-).} \tag{3.33}$$

H_k is a $2i$-by-n matrix where i is the number of measured lines and n is the number of states. For each measured line, the partial derivative of each line-point coordinate needs to be computed

$$\frac{\partial x_{lp}}{\partial s_i} = \frac{\partial x_{lp}}{\partial m_x}\frac{\partial m_x}{\partial s_i} + \frac{\partial x_{lp}}{\partial m_y}\frac{\partial m_y}{\partial s_i} + \frac{\partial x_{lp}}{\partial m_z}\frac{\partial m_z}{\partial s_i} \tag{3.34}$$

and

$$\frac{\partial y_{lp}}{\partial s_i} = \frac{\partial y_{lp}}{\partial m_x}\frac{\partial m_x}{\partial s_i} + \frac{\partial y_{lp}}{\partial m_y}\frac{\partial m_y}{\partial s_i} + \frac{\partial y_{lp}}{\partial m_z}\frac{\partial m_z}{\partial s_i}. \tag{3.35}$$

s_i above is a state variable.

The dual quaternion transformation repeated here is

$$\hat{l} = \hat{q}\hat{l}_m\hat{q}^*. \tag{3.36}$$

Expanding each dual quaternion gives

$$l + \epsilon m = (r + \epsilon s)(l_m + \epsilon m_m)(r^* + \epsilon s^*) \tag{3.37}$$

$$= r l_m r^* + \epsilon(r l_m s^* + r m_m r^* + s l_m r^*). \tag{3.38}$$

Let $t = [0\ \bar{t}]^T$ be a quaternion where $\bar{t}$ is the three-element translation vector. Then, $s = \frac{1}{2}tr$ [11]. Substituting in the above equation and solving for m gives

$$m = r m_m r^* + \frac{1}{2} r l_m r^* t^* + \frac{1}{2} t r l_m r^* \tag{3.39}$$

using the relation $s^* = \frac{1}{2} r^* t^*$. m is in terms of the state variables. The quaternion multiplications may be replaced by the corresponding matrix forms to give

$$m = \overset{+}{M}_r \overset{-}{M}^*_r\ m_m + \frac{1}{2}\ \overset{-}{M}_{t^*} \overset{+}{M}_r \overset{-}{M}_{r^*}\ l_m + \frac{1}{2}\ \overset{+}{M}_t \overset{+}{M}_r \overset{-}{M}_{r^*}\ l_m. \tag{3.40}$$

.

Factoring results in

$$m = \overset{+}{M}_r \overset{-}{M}_r^* \; m_m + \frac{1}{2}(\overset{+}{M}_t + \overset{-}{M}_{t^*}) \; \overset{+}{M}_r \overset{-}{M}_{r^*} \; l_m. \tag{3.41}$$

Let $R = \overset{+}{M}_r \overset{-}{M}_r^*$ and $M_{t+t^*} = \overset{+}{M}_t + \overset{-}{M}_{t^*}$. R has the standard 3-by-3 rotation matrix in the lower right submatrix. The partial derivatives of m with respect to each state variable can now be calculated

$$\frac{\partial m}{\partial t_i} = \frac{1}{2} \frac{\partial M_{t+t^*}}{\partial t_i} R l_m \tag{3.42}$$

where t_i is one of the three translation variables;

$$\frac{\partial m}{\partial q_i} = \frac{\partial R}{\partial q_i} m_m + \frac{1}{2} M_{t+t^*} \frac{\partial R}{\partial q_i} l_m \tag{3.43}$$

where q_i is one of the four rotational quaternion variables;

$$\frac{\partial m}{\partial v_i} = 0 \tag{3.44}$$

where v_i is one of the three linear velocity variables; and

$$\frac{\partial m}{\partial \omega_i} = 0 \tag{3.45}$$

where ω_i is one of the three angular velocity variables. M_{t+t^*} also has a simplified form,

$$M_{t+t^*} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -2t_3 & 2t_2 \\ 0 & 2t_3 & 0 & -2t_1 \\ 0 & -2t_2 & 2t_1 & 0 \end{bmatrix}. \tag{3.46}$$

l_m and m_m are quaternions that compose the dual quaternion representation of the object model and are known from the geometric description of the model. The matrix R is

$$R = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 0 & 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 0 & 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}. \tag{3.47}$$

The partial derivatives of R with respect to each rotational quaternion variable are

$$\frac{\partial R}{\partial q_0} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2q_0 & -2q_3 & 2q_2 \\ 0 & 2q_3 & 2q_0 & -2q_1 \\ 0 & -2q_2 & 2q_1 & 2q_0 \end{bmatrix} \tag{3.48}$$

$$\frac{\partial R}{\partial q_1} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2q_1 & 2q_2 & 2q_3 \\ 0 & 2q_2 & -2q_1 & -2q_0 \\ 0 & 2q_3 & 2q_0 & -2q_1 \end{bmatrix} \tag{3.49}$$

$$\frac{\partial R}{\partial q_2} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -2q_2 & 2q_1 & 2q_0 \\ 0 & 2q_1 & 2q_2 & 2q_3 \\ 0 & -2q_0 & 2q_3 & -2q_2 \end{bmatrix} \tag{3.50}$$

$$\frac{\partial R}{\partial q_3} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -2q_3 & -2q_0 & 2q_1 \\ 0 & 2q_0 & -2q_3 & 2q_2 \\ 0 & 2q_1 & 2q_2 & 2q_3 \end{bmatrix}. \tag{3.51}$$

The partial derivatives of M_{t+t^*} with respect to each translational variable are

$$\frac{\partial M_{t+t^*}}{\partial t_0} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 \\ 0 & 0 & 2 & 0 \end{bmatrix} \tag{3.52}$$

$$\frac{\partial M_{t+t^*}}{\partial t_1} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \end{bmatrix} \tag{3.53}$$

$$\frac{\partial M_{t+t^*}}{\partial t_2} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.54}$$

The partial derivatives of the image points x_{lp} and y_{lp} with respect to each element of $\vec{m}$ are given below:

$$\frac{\partial x_{lp}}{\partial m_x} = \frac{\lambda m_z}{(m_x^2 + m_y^2)} - 2\frac{\lambda m_x^2 m_z}{(m_x^2 + m_y^2)^2} \tag{3.55}$$

$$\frac{\partial x_{lp}}{\partial m_y} = -2\frac{\lambda m_x m_y m_z}{(m_x^2 + m_y^2)^2} \tag{3.56}$$

$$\frac{\partial x_{lp}}{\partial m_z} = \frac{\lambda m_x}{(m_x^2 + m_y^2)} \tag{3.57}$$

$$\frac{\partial y_{lp}}{\partial m_x} = -2\frac{\lambda m_x m_y m_z}{(m_x^2 + m_y^2)^2} \tag{3.58}$$

$$\frac{\partial y_{lp}}{\partial m_y} = \frac{\lambda m_z}{(m_x^2 + m_y^2)} - 2\frac{\lambda m_y^2 m_z}{(m_x^2 + m_y^2)^2} \tag{3.59}$$

$$\frac{\partial y_{lp}}{\partial m_z} = \frac{\lambda m_y}{(m_x^2 + m_y^2).} \tag{3.60}$$

3.8 Iterated Extended Kalman Filter Representation

The required elements of the IEKF are given above. An initial estimate is required based on prior knowledge. An initial error covariance matrix, P_0, that is dependent on the prior knowledge must be specified. Process and measurement noise covariance matrices, Q_k and R_k, respectively, must also be specified. After each iteration, the derived partial derivatives with respect to each state variable are calculated to determine the linearized equations for the Kalman filter. With the updated state estimate, a new linearization is then performed about this state. Several iterations about the present state are then performed converging to the most accurate estimate.

3.9 Extension of Iterated Extended Kalman Filter to Estimate Structure

When the object or scene geometry is unknown, the method described above is not sufficient since the 3–D model parameters are assumed known. In this section, the estimation method is extended to estimate structure in addition to the position and orientation. The structure is estimated in terms of line features for consistency with the previously developed theory. Both pose and structure are estimated simultaneously. However, without prior information about an absolute dimension of the object, the structure and the translation can only be estimated to within a scale factor due to perspective projection [75]. A small object in close proximity to the camera will project the same image as a large object far away. Rotation, on the other hand, can be determined absolutely. This section gives the representation of the structure by lines and the extensions to the IEKF.

3.9.1 Structure Representation

Lines have been represented above by the dual vector $\vec{l}+\epsilon\vec{m}$ where $\vec{l}$ is the direction vector normalized to unit magnitude and $\vec{m} = \vec{p} \times \vec{l}$ with $\vec{p}$ being a point on the line. The magnitude of $\vec{m}$ is equal to the perpendicular or shortest distance from the origin to the line. Also, since $\vec{m}$ is orthogonal to $\vec{l}$, $\vec{m} \cdot \vec{l} = 0$. Using this latter relation to estimate the sixth, five state variables are estimated per line. These variables are:

$$\begin{bmatrix} m_x & m_y & m_z & l_x & l_y \end{bmatrix}. \tag{3.61}$$

l_z can be derived from the dot product relation. The normalization relation for $\vec{l}$ is applied after l_z is calculated. This step is analogous to the quaternion normalization performed after the measurement update. The state vector given by Equation 3.1 is extended by $5n$ variables where n is the number of structure lines being estimated.

3.9.2 Extension to State Transition

Since the structure is assumed constant with time, the extension to the state transition function $\phi_k(s_k)$ and the state transition matrix $\Phi_k(s_k)$ is straightforward. For the state transition function, additional elements corresponding to the number of structure variables are added to the vector function to give an identity update. Additional columns are appended to the state transition matrix with an identity update for the corresponding state variable and zero in all other terms. Each structure line extends the matrix by five columns.

3.9.3 Extension to Measurement Update

The measurement update requires the partial derivatives of the image plane parameters with respect to each state variable. As given previously, the relation between the camera referenced line parameter vector $\vec{m}$ and the object referenced line is

$$\vec{m} = R\vec{m}_m + M_{t+t^*}Rl_m. \tag{3.62}$$

The partial derivatives of $\vec{m}$ with respect to each additional state variable have the form for the vector $\vec{m}_m$,

$$\frac{\partial \vec{m}}{\partial \vec{m}_m} = \begin{bmatrix} \partial m_x/\partial m_{m_x} & \partial m_y/\partial m_{m_x} & \partial m_z/\partial m_{m_x} \\ \partial m_x/\partial m_{m_y} & \partial m_y/\partial m_{m_y} & \partial m_z/\partial m_{m_y} \\ \partial m_x/\partial m_{m_z} & \partial m_y/\partial m_{m_z} & \partial m_z/\partial m_{m_z} \end{bmatrix} = R^T, \tag{3.63}$$

which is just the transpose of the rotation matrix. For the vector l_m, the result is

$$\frac{\partial \vec{m}}{\partial \vec{l_m}} = (M_{t+t^*}R)^T. \tag{3.64}$$

This expression is in terms of the translation and rotation and, as above, does not depend on the particular state variable. The first two rows correspond to the state variables l_{m_x} and l_{m_y}. The measurement update matrix is augmented with these values during estimation with the IEKF.

Chapter 4

Noise Analysis

4.1 Analysis of Line Noise

In the simulation tests, noise is applied to the projected image points of the model. The noise is assumed to be additive, independent, identically distributed Gaussian noise. These noisy points are then used directly as measurements for the reference point method against which the line method is compared. From these noisy points, lines are formed by connecting pairs of points. These lines, as represented by line-point parameters given in Chapter 3, are used as measurements for the line method. Since the noise for the points is independent with respect to x and y as well as each other, the measurement error covariance matrix for the point method is a straightforward diagonal matrix whose diagonal elements are the noise variances while the off-diagonal terms are zero. Figure 4.1 shows the noise added to the points as well as the variation of the line connecting the points. The noise characterization of the line, however, is not as simple as that for the points. The line-point measurement error covariance matrix has significant cross-covariance terms and is dependent on the location of the points within the image as well as the noise statistics. In this chapter, the derivation of the probability density function (pdf) for the x and y line-point parameters is given along with the determination of the covariance matrix. An approximation to the covariance matrix that permits real-time implementation is also shown.

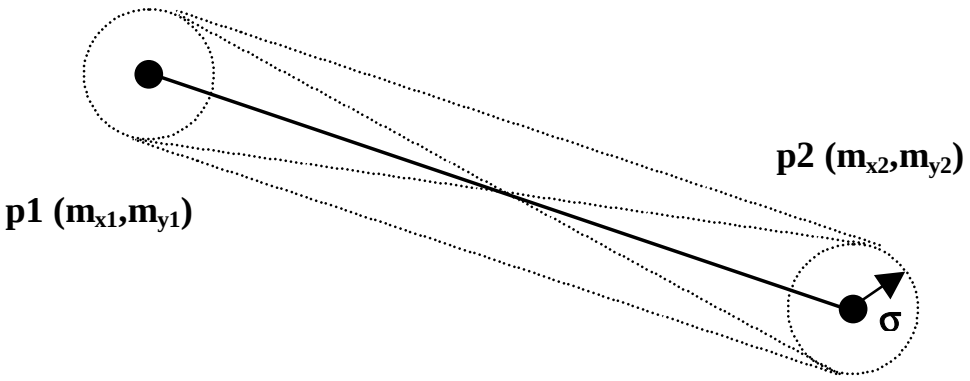

Fig. 4.1. Diagram showing a point pair with a noise radius of one standard deviation and the corresponding range of variation of the line connecting the points.

4.1.1 Probability Density Function of Line-Point Parameters

Consider two image points, $p1$ and $p2$, in an image where the x and y positions of both points are independent, identically distributed Gaussian random variables. Connecting these points as shown in Figure 4.1 produces a line whose line-point values can be expressed as a function of the end point x and y locations. The problem addressed here is to calculate the pdf of the x and y line-point values given the pdf of the end points. The end points are assumed to have a Gaussian density function. These pdf's are given below:

$$\begin{aligned}
p_{x_1}(x_1) &= \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{(x_1 - m_{x_1})^2}{2\sigma^2}\right) \\
p_{x_2}(x_2) &= \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{(x_2 - m_{x_2})^2}{2\sigma^2}\right) \\
p_{y_1}(y_1) &= \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{(y_1 - m_{y_1})^2}{2\sigma^2}\right) \\
p_{y_2}(y_2) &= \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{(y_2 - m_{y_2})^2}{2\sigma^2}\right)
\end{aligned} \tag{4.1}$$

where m_i is the mean value of point i.

The standard deviation σ of the point noise is assumed to be the same for all points. Mean values are the mean projections of the 3–D point on an object to the image plane. To simplify the derivation, the following intermediate transformation is initially defined:

$$\begin{aligned} x_d &= \frac{1}{2}(x_2 - x_1) \\ y_d &= \frac{1}{2}(y_2 - y_1) \\ x_s &= \frac{1}{2}(x_2 + x_1) \\ y_s &= \frac{1}{2}(y_2 + y_1). \end{aligned} \tag{4.2}$$

The pdf of a variable y given $y = f(x)$, and the pdf of x is [96]:

$$p_y(y) = \frac{p_x(x)}{|dy/dx|}\Big|_{x=f^{-1}(y)} \tag{4.3}$$

provided there is a one-one correspondence between x and y. Extending the pdf transformation to two variables results in

$$\begin{aligned} u &= f_1(x, y) \\ v &= f_2(x, y) \\ p_{uv}(u, v) &= \frac{p_{xy}(x, y)}{\left|J\left(\frac{u}{x}\frac{v}{y}\right)\right|} \end{aligned} \tag{4.4}$$

where J is the Jacobean as defined by the determinant

$$J = \begin{vmatrix} \partial u/\partial x & \partial u/\partial y \\ \partial v/\partial x & \partial v/\partial y \end{vmatrix}. \tag{4.5}$$

It is assumed that J exists, is continuous over the domain, and is nonzero. This ensures that the Jacobean has the same sign over the variable domain. In cases where the Jacobean is continuous but passes through zero, it is possible to consider the regions of same sign separately in evaluating the pdf [97].

Applying the bivarate pdf transformation to the specific case being examined here results in the pdf of x_s, y_s, x_d, and y_d in terms of the pdf of the point variables x_1, x_2, y_1, and y_2. First, the Jacobeans, $J_{x_d x_s}$ and $J_{y_d y_s}$, are determined:

$$J_{x_d x_s} = \begin{vmatrix} \partial x_d/\partial x_1 & \partial x_d/\partial x_2 \\ \partial x_s/\partial x_1 & \partial x_s/\partial x_2 \end{vmatrix} = \begin{vmatrix} -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{vmatrix} = -\frac{1}{2} \tag{4.6}$$

$$J_{y_d y_s} = \begin{vmatrix} \partial y_d/\partial y_1 & \partial y_d/\partial y_2 \\ \partial y_s/\partial y_1 & \partial y_s/\partial y_2 \end{vmatrix} = \begin{vmatrix} -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{vmatrix} = -\frac{1}{2}. \tag{4.7}$$

Applying the transformations in Equation 4.2 and simplifying gives

$$\begin{aligned} p_{x_d x_s}(x_d, x_s) &= \frac{p_{x_1 x_2}(x_1, x_2)}{\left|J_{x_d x_s}\right|} = 2 p_{x_1}(x_1) p_{x_2}(x_2) \\ p_{y_d y_s}(y_d, y_s) &= \frac{p_{y_1 y_2}(y_1, y_2)}{\left|J_{y_d y_s}\right|} = 2 p_{y_1}(y_1) p_{y_2}(y_2). \end{aligned} \tag{4.8}$$

Since the point random variables are statistically independent, the joint pdf's are equal to the product of the individual pdf's. Substituting the Gaussian expressions for the pdf of x_1 and x_2 in this equation and combining terms gives

$$
\begin{aligned}
p_{x_d x_s}(x_d, x_s) &= \frac{1}{2\pi {s^2}} exp\left(-\frac{(x_d - m_{x_d})^2}{2{s^2}}\right) exp\left(-\frac{(x_s - m_{x_s})^2}{2{s^2}}\right) \\
p_{y_d y_s}(y_d, y_s) &= \frac{1}{2\pi {s^2}} exp\left(-\frac{(y_d - m_{y_d})^2}{2{s^2}}\right) exp\left(-\frac{(y_s - m_{y_s})^2}{2{s^2}}\right)
\end{aligned} \tag{4.9}
$$

where $s = \frac{\sigma}{\sqrt{2}}$ is the standard deviation of the transformed variables x_d, x_s, y_d, and y_s.

Since $p_{x_d x_s}(x_d, x_s)$ and $p_{y_d y_s}(y_d, y_s)$ may be separated to give separate Gaussian pdf's, $p_{x_d} p_{x_s}$ and $p_{y_d} p_{y_s}$, x_s, x_d, y_s, and y_d are statistically independent. As shown below, this transformation simplifies the calculation of the pdf of the line-point parameters, x_{lp} and y_{lp}, given below. The line point, as given in Chapter 3, can be found from the line direction and a point on the line. The direction vector l of the line formed by the two points is

$$
\begin{bmatrix} l_x \\ l_y \end{bmatrix} = \begin{bmatrix} \frac{x_d}{\sqrt{x_d^2 + y_d^2}} \\ \frac{y_d}{\sqrt{x_d^2 + y_d^2}} \end{bmatrix}. \tag{4.10}
$$

A point p on the line is (x_s, y_s), the line midpoint, and, since $m = p \times l$, m is

$$
m = \begin{bmatrix} 0 \\ 0 \\ x_s l_y - y_s l_x \end{bmatrix}. \tag{4.11}
$$

To recall the previously derived relation for x_{lp} and y_{lp} from Chapter 3,

$$
x_{lp} = l_y m_z \text{ and } y_{lp} = -l_x m_z. \tag{4.12}
$$

Substituting for l_x, l_y, and m_z gives

$$\begin{aligned} x_{lp} &= \frac{x_s y_d^2 - y_s x_d y_d}{x_d^2 + y_d^2} \\ y_{lp} &= \frac{y_s x_d^2 - x_s x_d y_d}{x_d^2 + y_d^2}. \end{aligned} \tag{4.13}$$

Note that the x_{lp} and y_{lp} are duals in that the corresponding equations are the same except that x is replaced by y to convert from one equation to the other. Any result for one term may be easily found for the other.

Since x_{lp} is a function of the derived variables, the pdf of x_{lp} may be found from the transformation given by the form of Equation 4.4 except that four variables are involved in the transformation,

$$p_{x_{lp}x_d y_d y_s}(x_{lp}, x_d, y_d, y_s) = \frac{p_{x_d y_d x_s y_s}(x_d, y_d, x_s, y_s)}{\left| J_{x_{lp}} \left(\frac{x_{lp} x_d y_d y_s}{x_d y_d x_s y_s} \right) \right|}. \tag{4.14}$$

x_s is eliminated in the transformation. Equation 4.13 is used to solve for x_s in terms of the remaining variables giving

$$x_s = \frac{(x_d^2 + y_d^2) x_{lp} + y_s x_d y_d}{y_d^2}. \tag{4.15}$$

The Jacobean $J_{x_{lp}}$ is

$$J_{x_{lp}} = \begin{vmatrix} \frac{\partial x_{lp}}{\partial x_d} & \frac{\partial x_{lp}}{\partial y_d} & \frac{\partial x_{lp}}{\partial x_s} & \frac{\partial x_{lp}}{\partial y_s} \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{4.16}$$

$$= \frac{y_d^2}{x_d^2 + y_d^2}. \tag{4.17}$$

$J_{x_{lp}}$ here is positive and continuous and exists for all pairs of points that are physically separate. Substituting this result in Equation 4.14 gives

$$p_{x_{lp}x_d y_d y_s}(x_{lp}, x_d, y_d, y_s) = \frac{x_d^2 + y_d^2}{y_d^2} p_{x_d}(x_d) p_{y_d}(y_d) p_{x_s}(x_s) p_{y_s}(y_s). \tag{4.18}$$

The products of the individual pdf's are shown instead of the joint density function since all variables are independent. Integration of this expression provides the sought after pdf of x_{lp},

$$p_{x_{lp}}(x_{lp}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left(\frac{x_d^2 + y_d^2}{y_d^2}\right) p_{x_d}(x_d) p_{y_d}(y_d) p_{x_s}(x_s) p_{y_s}(y_s) dx_d dy_d dy_s, \tag{4.19}$$

with Equation 4.15 substituted for x_s. This result has no direct analytical solution but may be solved numerically. Figures 4.2, 4.3, 4.4, and 4.5 show several examples of $p_{x_{lp}}$ and $p_{y_{lp}}$ calculated numerically. The examples show two horizontal lines, a -1 slope line and a +1 slope line. In addition, the best-fit Gaussian is shown overlaid on the plot. For each

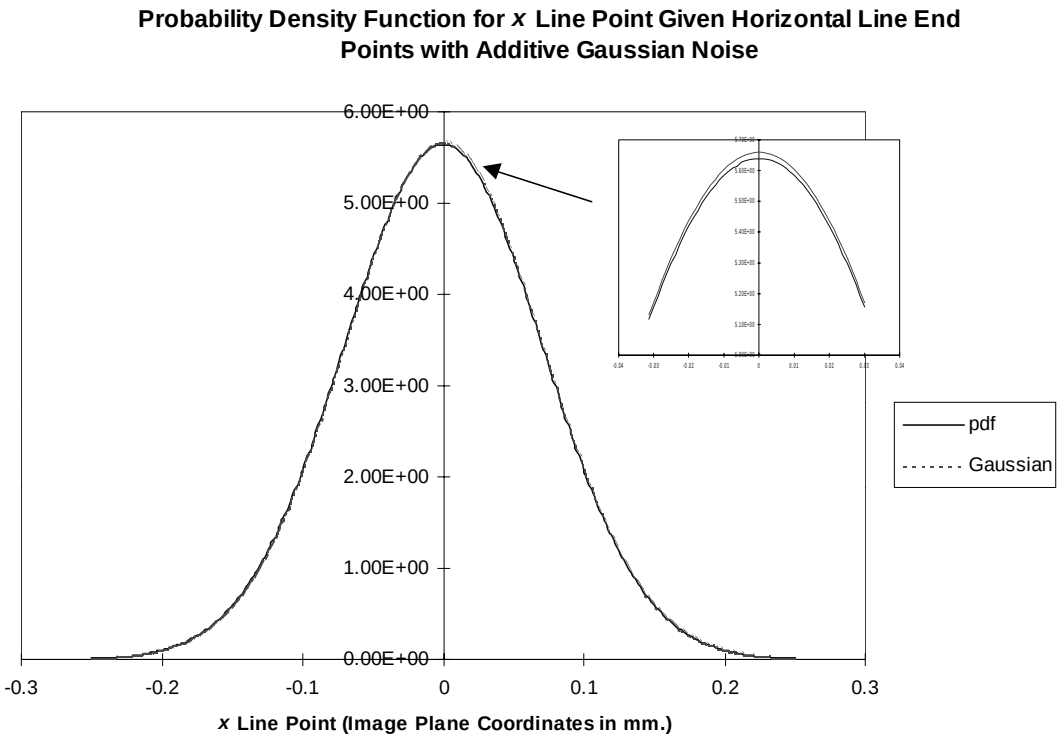

Fig. 4.2. Probability density function of x_{lp} for 1.25 mm. length horizontal line centered about the y axis, y_s= 3.125 mm., point noise 0.02 mm. standard deviation. A best-fit Gaussian pdf is also plotted, but the differences are seen to be small.

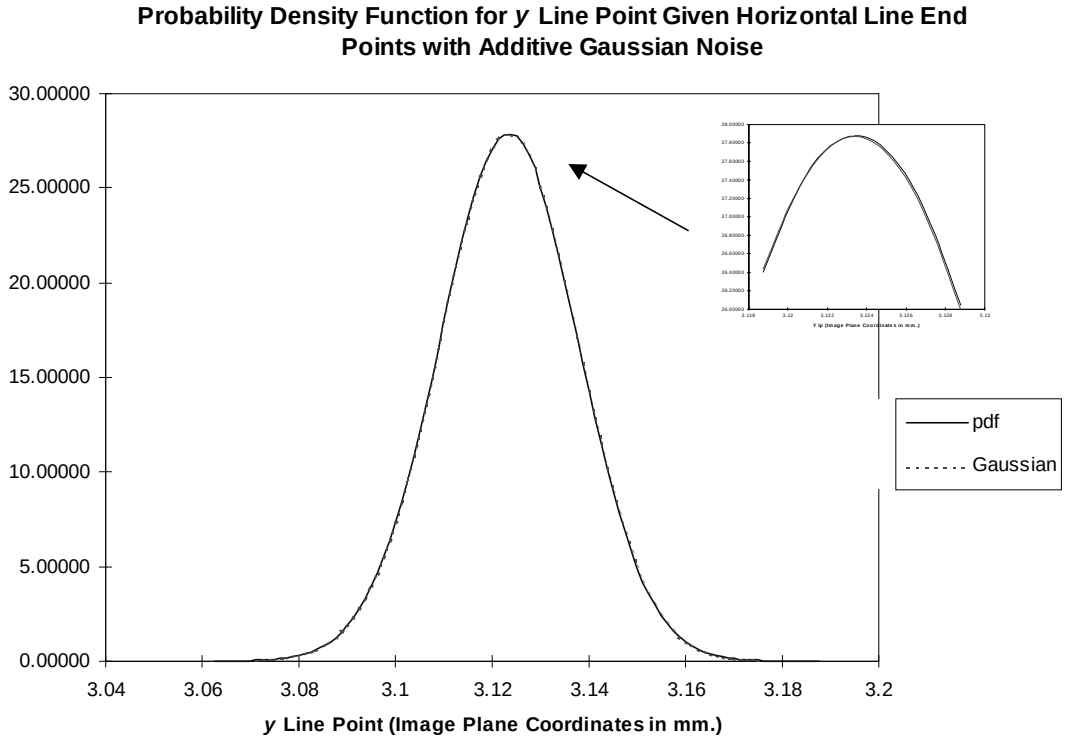

Fig. 4.3. Probability density function of y_{lp} for 1.25 mm. length horizontal line centered about the y axis, y_s= 3.125 mm., point noise 0.02 mm. standard deviation. A best-fit Gaussian pdf is also plotted, but the differences are seen to be small.

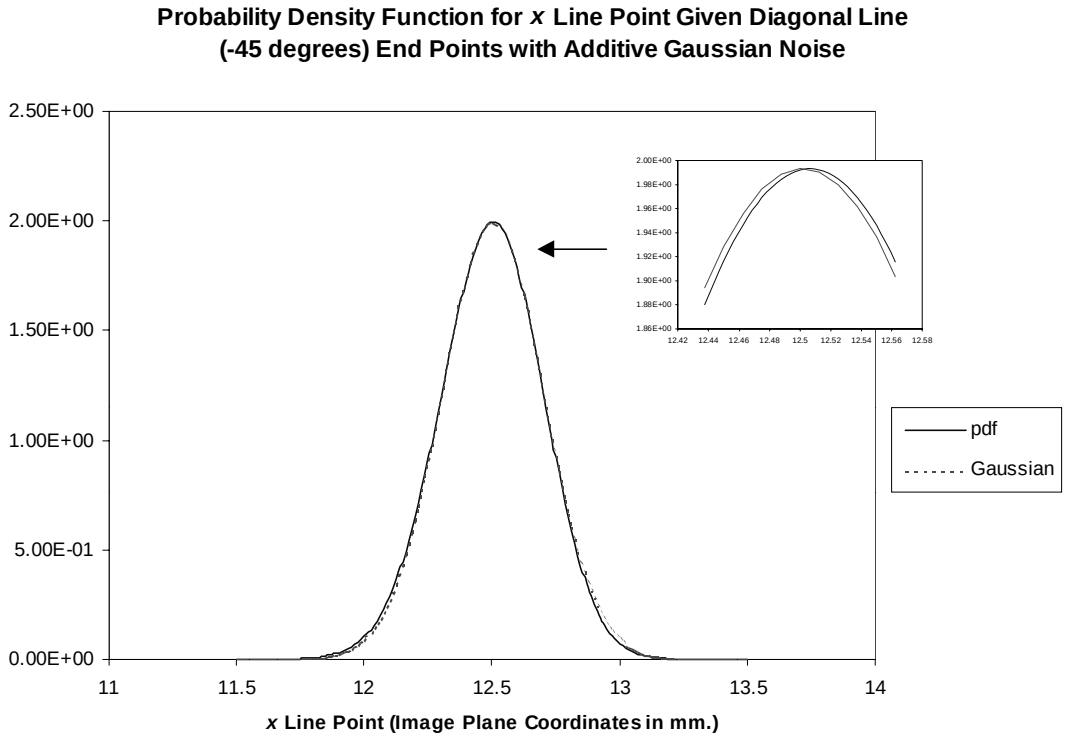

Fig. 4.4. Probability density function of x_{lp} for 1.25 mm. length downward diagonal line at x_s=12.5 mm., y_s= 12.5 mm., point noise 0.02 mm. standard deviation. A best-fit Gaussian pdf is also plotted, but the differences are seen to be small.

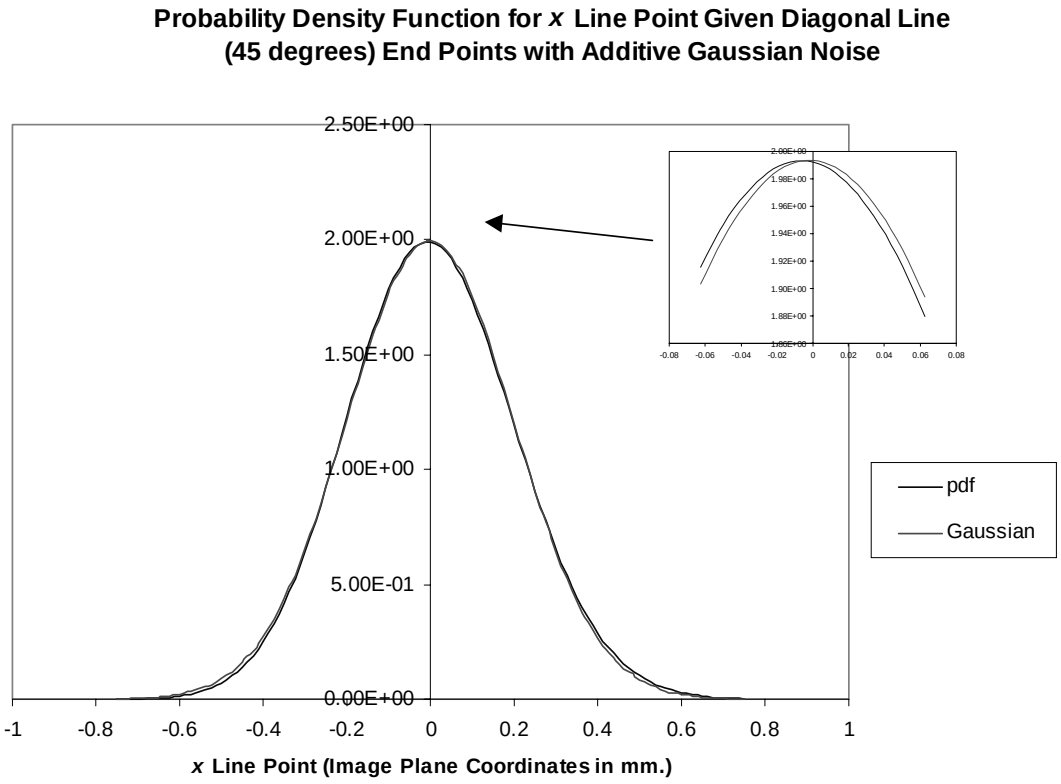

Fig. 4.5. Probability density function of x_{lp} for 1.25 mm. length upward diagonal line at x_s=12.5 mm., y_s= 12.5 mm., point noise 0.02 mm. standard deviation. A best-fit Gaussian pdf is also plotted, but the differences are seen to be small.

case, the Gaussian is seen to provide a good fit to the data. These examples show that for a wide range of point-pair combinations, the pdf of the line point may be approximated by a Gaussian. The motivation here is to show that the line-point measurement input noise to the Kalman filter is approximately Gaussian, a condition necessary to achieve optimum performance. Note also that the variance of the density functions is not equal to the point noise variance and varies depending on the line configuration.

4.1.2 Covariance of x_{lp} and y_{lp}

In this section, the covariance expressions for the line-point parameters x_{lp} and y_{lp} are derived. The expectation of the line-point parameter, x_{lp} in terms of the derived point parameters, is

$$\begin{aligned}\overline{x_{lp}} &= \overline{f_x(x_d, y_d, x_s, y_s)} \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_x(x_d, y_d, x_s, y_s) p_{x_d}(x_d) p_{y_d}(y_d) p_{x_s}(x_s) p_{y_s}(y_s) \\ &\quad dx_d dy_d dx_s dy_s; \end{aligned} \tag{4.20}$$

and, for y_{lp}, it is

$$\begin{aligned}\overline{y_{lp}} &= \overline{f_y(x_d, y_d, x_s, y_s)} \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_y(x_d, y_d, x_s, y_s) p_{x_d}(x_d) p_{y_d}(y_d) p_{x_s}(x_s) p_{y_s}(y_s) \\ &\quad dx_d dy_d dx_s dy_s. \end{aligned} \tag{4.21}$$

The variance of x_{lp} is

$$var_{x_{lp}} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(x_{lp} - \overline{x}_{lp})^2 p_{x_d}(x_d)p_{y_d}(y_d)p_{x_s}(x_s)p_{y_s}(y_s)dx_d dy_d dx_s dy_s; \tag{4.22}$$

and, for y_{lp}, it is

$$var_{y_{lp}} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(y_{lp} - \overline{y}_{lp})^2 p_{x_d}(x_d)p_{y_d}(y_d)p_{x_s}(x_s)p_{y_s}(y_s)dx_d dy_d dx_s dy_s \tag{4.23}$$

while the covariance of x_{lp} and y_{lp} is

$$\begin{aligned}\overline{(x_{lp}y_{lp})^2} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(x_{lp} - \overline{x}_{lp})(y_{lp} - \overline{y}_{lp})p_{x_d}(x_d)p_{y_d}(y_d)p_{x_s}(x_s)p_{y_s}(y_s) \\ &\quad dx_d dy_d dx_s dy_s.\end{aligned} \tag{4.24}$$

These expressions are exact and may be solved numerically since no analytical solution is known. However, a quadruple integration is computationally intensive. Simplifications are possible to reduce the required computation with one approach given here. Equation 4.13 may be expressed in the form

$$\begin{aligned}x_{lp} &= \sin^2\theta x_s - \sin\theta\cos\theta y_s \\ y_{lp} &= \cos^2\theta y_s - \sin\theta\cos\theta x_s\end{aligned} \tag{4.25}$$

where

$$\sin\theta = \frac{y_d}{\sqrt{x_d^2 + y_d^2}} \text{ and } \cos\theta = \frac{x_d}{\sqrt{x_d^2 + y_d^2}}. \tag{4.26}$$

θ is the angle of the line measured with respect to the x axis. The range is restricted to between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$ since the signs of x_d and y_d are arbitrary depending on which point is chosen as 1 and which is chosen as 2. The expected value or mean for the line-point parameters can then be expressed as

$$\begin{aligned} \overline{x}_{lp} &= \overline{\sin^2\theta} m_{x_s} - \overline{\sin\theta\cos\theta} m_{y_s} \\ \overline{y}_{lp} &= \overline{\cos^2\theta} m_{y_s} - \overline{\sin\theta\cos\theta} m_{x_s} \end{aligned} \tag{4.27}$$

since θ and x_s, y_s are independent. This reduces the problem to calculating the means of the trigonometric quantities

$$\begin{aligned} \overline{\sin^2\theta} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{y_d^2}{x_d^2 + y_d^2} p_{x_d}(x_d) p_{y_d}(y_d) dx_d dy_d \\ \overline{\cos^2\theta} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{x_d^2}{x_d^2 + y_d^2} p_{x_d}(x_d) p_{y_d}(y_d) dx_d dy_d \\ \overline{\sin\theta\cos\theta} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{x_d y_d}{x_d^2 + y_d^2} p_{x_d}(x_d) p_{y_d}(y_d) dx_d dy_d. \end{aligned} \tag{4.28}$$

The integral here is only a double integral requiring considerably less computation than the triple integral of Equation 4.20. The expressions for the variances and covariance can also be simplified. These quantities are given below in terms of expectation operators:

$$
\begin{aligned}
s^2_{x_{lp}} &= \overline{x^2_{lp}} - \overline{x}_{lp}{}^2 \\
s^2_{y_{lp}} &= \overline{y^2_{lp}} - \overline{y}_{lp}{}^2 \\
s_{x_{lp}y_{lp}} &= \overline{x_{lp}y_{lp}} - \overline{x}_{lp}\overline{y}_{lp}
\end{aligned} \tag{4.29}
$$

where $s^2_{x_{lp}}$ and $s^2_{y_{lp}}$ are the variances of x_{lp} and y_{lp}, respectively, and $s_{x_{lp}y_{lp}}$ is the covariance of x_{lp}, y_{lp}. Substituting Equation 4.25 for x_{lp} and y_{lp} and simplifying gives the following:

$$
\begin{aligned}
s^2_{x_{lp}} &= \overline{\sin^4\theta}(m^2_{x_s} + s^2) - 2\overline{\cos\theta\sin^3\theta}m_{x_s}m_{y_s} + \overline{\cos^2\theta\sin^2\theta}(m^2_{y_s} + s^2) - \overline{x}^2_{lp} \\
s^2_{y_{lp}} &= \overline{\cos^4\theta}(m^2_{y_s} + s^2) - 2\overline{\cos^3\theta\sin\theta}m_{x_s}m_{y_s} + \overline{\cos^2\theta\sin^2\theta}(m^2_{x_s} + s^2) - \overline{y}^2_{lp} \\
s_{x_{lp}y_{lp}} &= 2\overline{\cos^2\theta\sin^2\theta}m_{x_s}m_{y_s} - \overline{\cos^3\theta\sin\theta}(m^2_{y_s} + s^2) \\
&\quad -\overline{\cos\theta\sin^3\theta}(m^2_{x_s} + s^2) - \overline{x}_{lp}\overline{y}_{lp}
\end{aligned} \tag{4.30}
$$

where s^2 is the variance of x_s and y_s.

Each of the trigonometric expectations may be solved by double integration in the form of Equation 4.28. These values are then used to calculate the variances and covariances. Numerical integration has been used to calculate these covariances for several cases of line orientation. Figure 4.6 shows the change in $s^2_{x_{lp}}$ for a 1.25 mm. long horizontal line centered about the y axis as y_s increases from 0 to 12.5 mm. Figure 4.7 gives similar results for a diagonal line at a 45 degree angle except that the large variability is in $s^2_{y_{lp}}$. While the covariance is negligible for horizontal or vertical lines, the covariance is maximum for a 45 degree slope line. Another factor that influences the variance is the

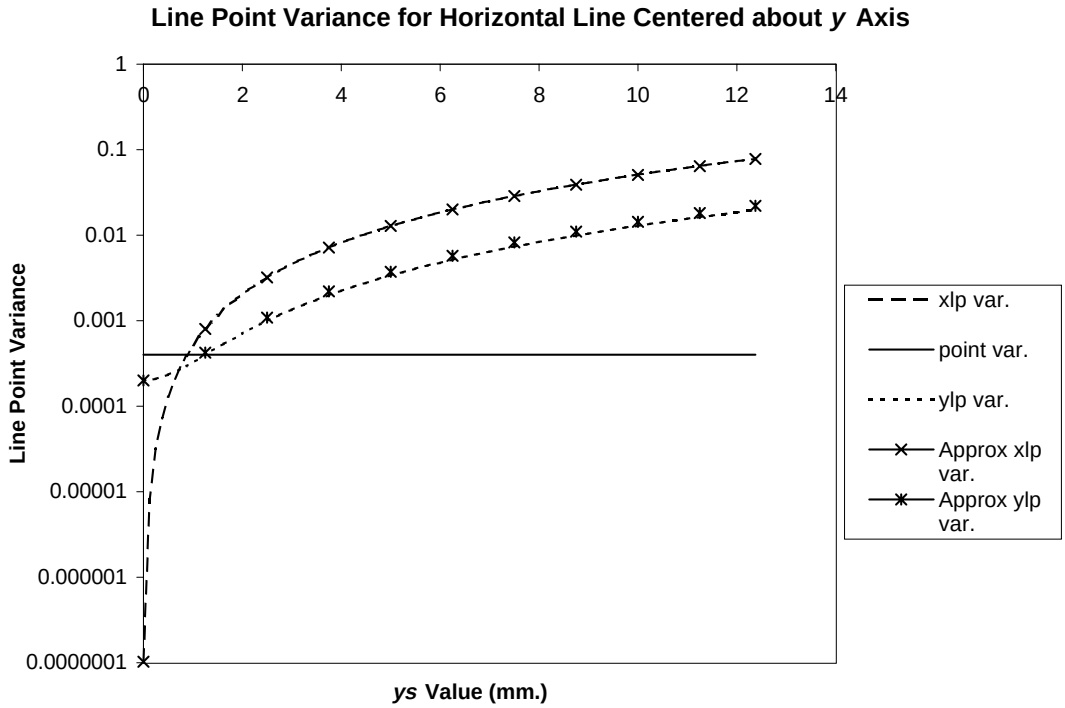

Fig. 4.6. Variance of line-point parameters for 1.25 mm. horizontal line centered on the y axis as y_s varies, point noise 0.02 mm. standard deviation, compared with point noise variance. y_{lp} variance is always less than the point variance while the x_{lp} variance becomes greater after only a small displacement.

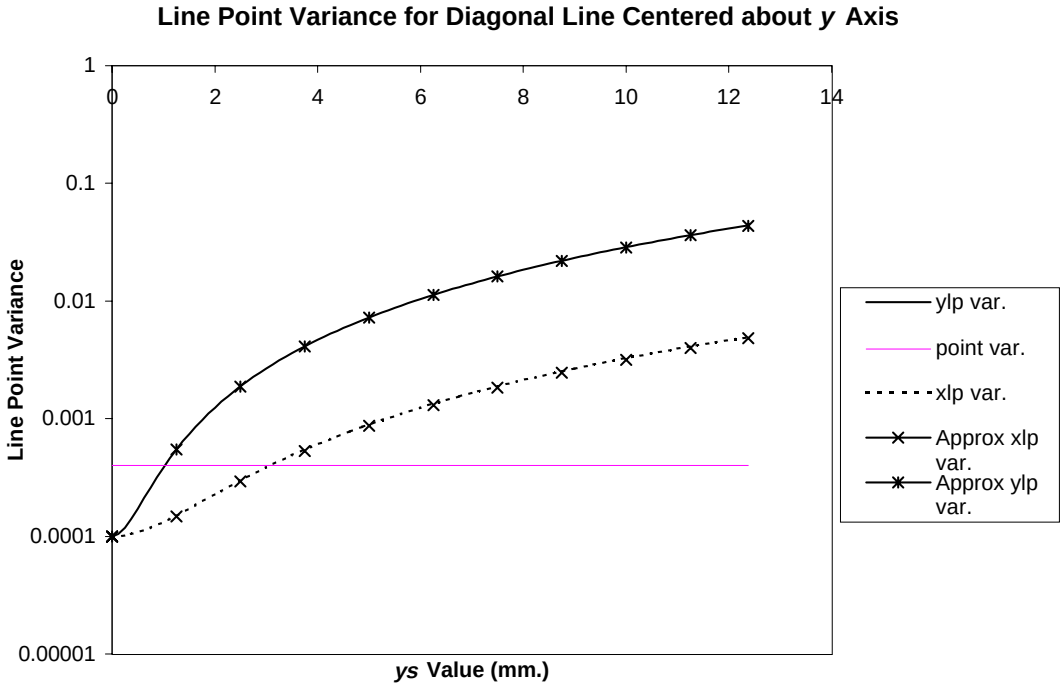

Fig. 4.7. Variance of line-point parameters for 1.25 mm. diagonal line (45 degree angle) centered on the y axis as y_s varies, point noise 0.02 mm. standard deviation, compared with point noise variance. x_{lp} variance is always less than the point variance while the y_{lp} variance becomes greater after only a small displacement.

length of the line. As the line length increases, the variation in angle decreases resulting in smaller variances.

This result is an important factor in the Kalman filter performance. The covariances of the line points determine the values of the measurement covariance matrix in the filter formulation. The magnitude of the covariances reflects the uncertainty in the measurement. Where the covariance is low, the measured quantity has low uncertainty. When the Kalman filter performs the measurement update, the input measurements are weighted according to the covariance matrix values. In comparison with the point method, the variances and covariances may be lower where the lines are longer and near the center of the image. Short lines and lines farther away from the origin will have larger variances than the points from which the lines are formed. The line method would be expected to provide better estimates for objects in the center of the image than would the point method. As the object moves significantly away from the center or moves away from the camera, the performance advantage of the line method would be expected to decrease and, at some position, to drop below that of the point method. The implication here for a moving object or scene is that the measurement covariance matrix must be updated to provide the optimal weighing for the measurements at each sample instance. A fixed covariance matrix would be expected to give worse performance than an adaptive one for the same measurements.

4.2 Real-Time Implementation

The evaluation of the covariance values depends on the numerical double integration of several terms. The direct numerical calculation of these quantities is not feasible for a real-time implementation. An approximation, however, has been developed that enables its use in real-time applications. In the expressions for the variance calculations above, the expected values of the trigonometric quantities depend on m_{x_d} and m_{y_d}. For a scene

with moving objects that are being tracked, the line features that are projected to the image plane thus change with time. At each iteration, the measurement covariance matrix R needs to be updated based on the variance of the line-point parameters. The following procedure gives the steps needed for the real-time estimation broken down into preliminary calculations and then the calculations done during each iteration.

The preprocessing calculations are as follows:

Let $m_{ij} = \overline{\cos^i \theta \sin^j \theta}$. Calculate m_{10}, m_{01}, m_{11}, m_{20}, m_{02}, m_{40}, m_{31}, m_{22}, m_{13}, and m_{04} for a table of angles from $-\frac{\pi}{2}$ to $\frac{\pi}{2}$. The number of steps stored in this table determines the resulting accuracy of the estimate. For example, 100 steps would resolve the angle to 1.8 degrees. The variance terms are given by

$$v_{ij} = m_{ij} - \prod_i m_{10} \prod_j m_{01}. \tag{4.31}$$

v_{ij} is the variance of the corresponding $\sin^i \theta \cos^j \theta$ term with respect to x_d and y_d. An initial variance and an initial line length are assumed. These form the variance and line length squared ratio. During each iteration, an adjustment is made to the variance v_{ij} based on the actual line length.

The run time processing calculations performed at each iteration are:

Calculate the angle and length squared of each feature line. The variance of the x_d and y_d variables is assumed known. Calculate the ratio of the variance and line length squared. Then, calculate the ratio of this quantity to the

precalculated ratio. Index the moment and variance table according to the nearest angle value. Then, calculate

$$s_{20} = m_{10}m_{10} + ratio\, v_{20} \tag{4.32}$$

$$s_{11} = m_{10}m_{01} + ratio\, v_{11} \tag{4.33}$$

and the remaining s_{ij} terms in a similar manner. These s_{ij} terms are the approximate expected value terms to be substituted in Equation 4.30. The ratio is used to correct the variances for differences in line length and point noise standard deviation.

The variances calculated using this simplified method are shown in Figures 4.6 and 4.7. The real-time method is seen to give good results in comparison with the true values. Simulation results in applying this adaptive method are given in Chapter 5.

Chapter 5

Experimental Results

The theoretical model for the image-based pose and motion estimation method has been developed in Chapter 3. A suitable application is a dynamic system application with a recursive solution that can be used in a real-time environment. This chapter provides an evaluation for this model through both simulation and actual physical system testing in this environment. The purpose is to demonstrate that the method works, that it performs accurately and reliably, and that the relative performance in comparison with other prior methods is good.

A dynamic vision application for the approach is defined where a camera provides the visual feedback for a robot performing a task. Simulation testing is presented first. This testing measures the accuracy of the estimation under an assumed noise distribution and magnitude. A target object with point features is used for the simulation. The results are compared with corresponding results from a point-based extended Kalman filtering method using an identical target to illustrate the performance differences. Speed of convergence, mean square error, and stability are presented and analyzed. In the second part of the chapter, actual test results measuring relative motion and position from a robot arm are presented. Relative accuracy and noise characterization are given.

5.1 Simulation Results

Simulation testing has been performed to evaluate the performance of the pose estimation method under a variety of conditions. The results are obtained by varying noise levels, initial conditions, and object complexity to characterize errors, convergence,

and stability in the state estimate. The EKF calculates the state estimate that includes the translation, the rotation quaternion, the linear velocity, and the angular velocity. Mean square errors associated with each of these state variables are presented. Figure 5.1 shows the functional blocks associated with the estimation. The following dynamic model assumptions given below were used for the simulation.

5.1.1 Dynamic Model

The object to be tracked is simulated, along with a camera located some distance away viewing the object. The reference frame for the object is defined in terms of the object and therefore moves as the object moves relative to a base coordinate reference. The camera and its corresponding reference frame are fixed relative to this base frame. A path for the object is defined in terms of its position and orientation over time with respect to the camera reference frame

$$\vec{x}(t) = \vec{f}(t) \tag{5.1}$$

where $\vec{x}(t)$ is the position and orientation state vector and $\vec{f}(t)$ is, in general, a nonlinear function of time. Smooth continuous motion is assumed so that all time derivatives of the state exist. In addition, $\vec{f}(t)$ is assumed to be the solution of a linear differential vector equation. With these assumptions, a state vector composed of $\vec{x}$ and a finite number of derivatives with respect to time of $\vec{x}$ permits the expression of the path motion as a linear system model,

$$\vec{s}_k = \Phi_{\text{true}} \vec{s}_{k-1}, \tag{5.2}$$

where Φ_{true} is a constant coefficient matrix of dimension n-by-n. n is the total number of 3-D parameters and derivatives required to represent the state s_k. The simulated target

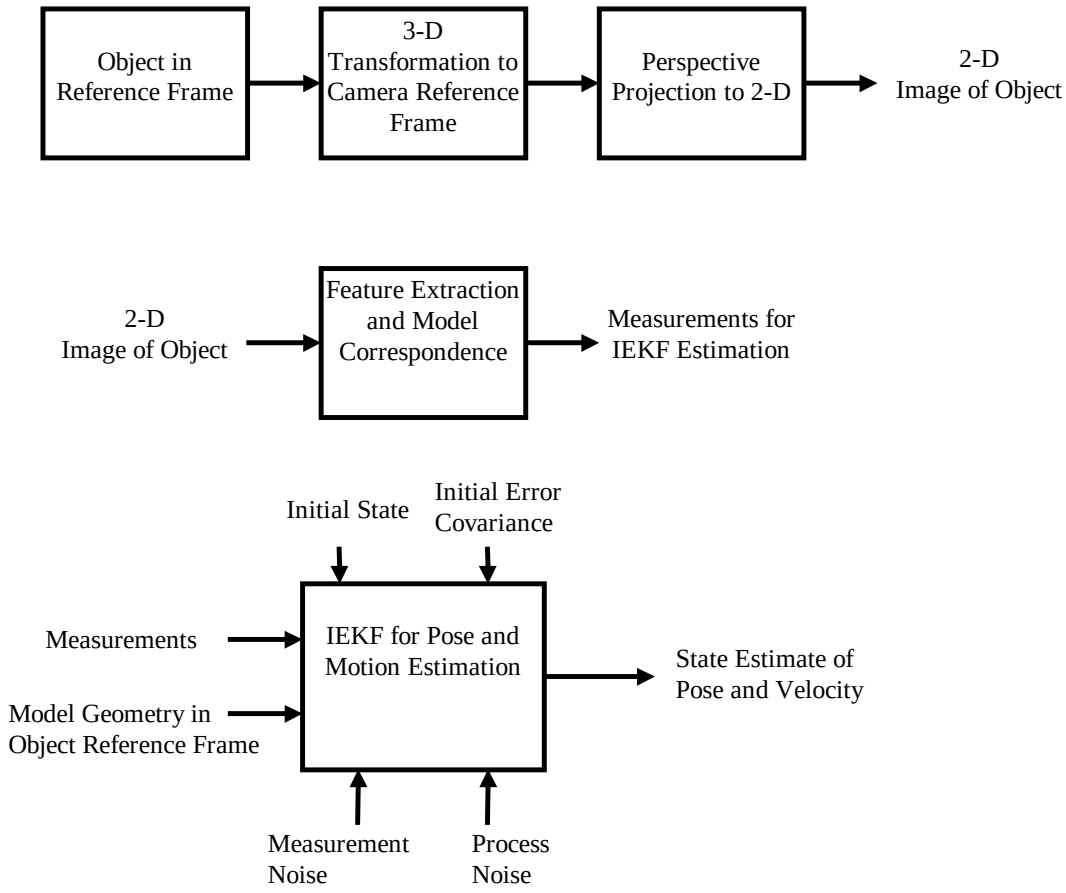

Fig. 5.1. Functional block diagram of the estimation problem showing the operations involved in the pose estimation method from optical imaging to state estimate produced by the iterated extended Kalman filter.

object starts at an initial position and orientation aligned with the camera reference and separated by a distance of 1000 cm. in the z-axis direction. The position and orientation are modeled by the set of variables

$$\text{pose} = \{x \ y \ z \ \theta_x \ \theta_y \ \theta_z\} \tag{5.3}$$

where θ_x, θ_y, and θ_z are rotation angles about the x, y, and z axes, respectively. These orthogonal rotation angles are commonly referred to as pitch, yaw, and roll. A trajectory is defined for the simulation that assumes constant velocity in all state variables.

5.1.2 Model Trajectory

For the given trajectory with constant velocities, each pose variable has the form

$$s^i = c_i t + d_i \tag{5.4}$$

with velocity c_i and initial value d_i. In this formulation, the state vector is defined as

$$\vec{s} = [x \quad y \quad z \quad \theta_x \quad \theta_y \quad \theta_z \quad v_x \quad v_y \quad v_z \quad \omega_x \quad \omega_y \quad \omega_z]^T. \tag{5.5}$$

The value of the second derivative of the ith state variable is

$$\ddot{s}^i = w_i \tag{5.6}$$

where w_i is a white noise sequence from $N(0, \sigma)$. Forming the transition matrix for the state sequence gives

$$\vec{s}_k = \Phi_{k-1}\vec{s}_{k-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & c_1\tau & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & c_2\tau & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & c_3\tau & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & c_4\tau & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & c_5\tau & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & c_6\tau \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.7}$$

with time interval τ and initial state vector

$$\vec{s}_0 = [d_1 \;\; d_2 \;\; d_3 \;\; d_4 \;\; d_5 \;\; d_6 \;\; c_1 \;\; c_2 \;\; c_3 \;\; c_4 \;\; c_5 \;\; c_6]^T . \tag{5.8}$$

5.1.3 Camera Model

An ideal camera model is used in the simulation. Perspective projection is assumed for the camera with a known effective focal length. No lens distortion correction is applied. Image acquisition and feature extraction are assumed performed so that the simulated measurement data corresponds to the location of the feature in the image plane in actual units. Noise of an assumed magnitude and distribution is added to the image feature locations before processing.

5.1.4 Object Model

A target object is simulated with individual feature points. Pairs of these points, when extracted from the image plane, are connected together to form lines. For these tests, two objects were defined. The first has four coplanar points in a rectangular pattern. The second target has six points with four being coplanar, as with the first target, and two located outside the plane. Figure 5.2(a) shows the dimensions and the shape of the four-point target while Figure 5.2(b) is for the six-point target.

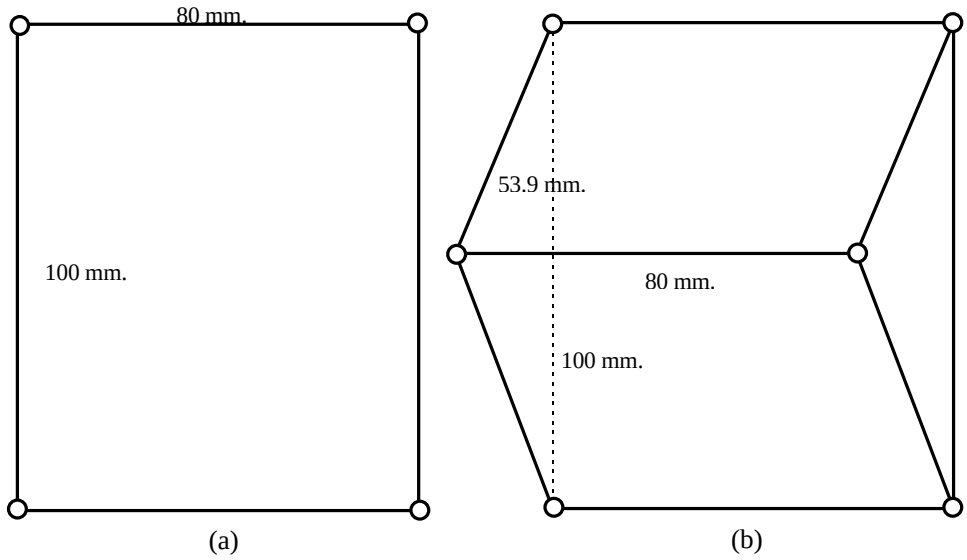

Fig. 5.2. Target shapes and dimensions used in the simulation experiments: (a) four-point coplanar target, and (b) six-point polyhedral 3-D target.

5.1.5 Estimation Model

The IEKF proposed here is used to estimate the position, orientation, and corresponding velocities of the object with respect to the camera. The noisy image plane

feature locations are used as inputs along with the *a priori* knowledge. As previously given, the estimated state vector is

$$\vec{s} = \begin{bmatrix} x & y & z & q_0 & q_1 & q_2 & q_3 & v_x & v_y & v_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^T . \tag{5.9}$$

5.1.6 Comparison Model

To evaluate the proposed method, a comparison model for estimation using an EKF has been simulated based on previous work [75]. The state vector of estimated variables is the same as Equation (5.9) above. Point features instead of line features are used for input measurements. The frame transformation for a point $\vec{x}$ is modeled as

$$\vec{x}' = R\vec{x} + \vec{t}. \tag{5.10}$$

The same series of simulated tests were performed on this model. A comparison of results that shows the performance differences obtained between the two methods is given.

5.1.7 Initial Conditions

Initial conditions requiring specification include the initial state, $\vec{s}_0$, and the error covariance matrix, P_0. The state vector may be considered a collection of Gaussian random variables with covariance P_0. The initial state is a sample taken from each random variable. Process noise given by the covariance matrix Q is also specified as an initial condition for the simulations, remaining constant throughout. Similarly, measurement noise given by the covariance matrix R is specified initially as a constant.

5.1.8 Measurement Noise Model

Optimal linear Kalman filtering assumes that both measurement noise and process noise are zero mean, Gaussian distributed. A Gaussian distribution is generally a good

approximation for phenomena that are the result of a large number of random fluctuations. However, in the case considered here, line or point features are extracted from a 2–D digital image containing pixels quantized to discrete locations. The measurement noise corresponds to the variation in location of the feature from successive image frames. As given by [98], the actual feature locations are not always closely approximated by a true Gaussian distribution. Since the pixels are quantized and since the features are very localized within the image, a more exact noise model is used. This model is a zero mean, truncated Gaussian distribution in which no noise samples are present exceeding an integral number of standard deviations. In this section, the truncation factor is set at two standard deviations of the normal Gaussian distribution.

5.1.9 Test Results

Simulation tests were performed using the four-point object model. These tests give both a sample run and the experimentally determined RMS error over time for the model trajectory. In addition, the point-based IEKF method was implemented and tested under simulation as a basis for comparison. The results show data from both methods.

Table 5.1 gives the initial state conditions for the simulation using the trajectory described above with a four-point geometric model. The initial states are the same for both methods. Table 5.2 gives the assumed initial error covariance diagonal terms and the process noise covariance diagonal terms for both the dual quaternion method and the point method used for comparison. Note that the process noise is different for each method. In all cases, the off-diagonal terms of the covariance matrices are zero. For these tests, the simulated sample interval is 0.1 second. In these tests, the nonadaptive dual quaternion measurement error method was used. The error values were held constant over the entire measurement interval.

Table 5.1.
Actual and assumed initial states of the extended Kalman filter for the four-point simulation tests. The initial states are the same for both the dual quaternion method and the point method.

State	Translation $x\ y\ z$ (mm.)	Quaternion $q_0\ q_1\ q_2\ q_3$	Linear Velocity $x\ y\ z$ (mm./sec.)	Rotational Velocity $x\ y\ z$ (rad./sec.)
True Initial State	10 10 1000	1 0 0 0	-5 2 -5	-0.03 0.05 -0.2
Initial State Estimate	0 0 990	0.9998 0.01 0.01 0.01	0 0 0	0 0 0

Table 5.2.
Dual quaternion method and point method initial error covariance matrix and process noise matrix diagonal terms of the extended Kalman filter for the four-point simulation tests.

Noise Covariance	Translation $x\ y\ z$ (mm.^2)	Quaternion $q_0\ q_1\ q_2\ q_3$	Linear Velocity $x\ y\ z$ $(\text{mm./sec.})^2$	Rotational Velocity $x\ y\ z$ $(\text{rad./sec.})^2$
Dual Quaternion Error Covariance	100	0.01	100	0.1
Dual Quaternion Process Noise	10^{-5}	10^{-5}	10^{-5}	10^{-6}
Point Method Error Covariance	100	0.01	10	0.1
Point Method Process Noise	10^{-4}	10^{-6}	10^{-4}	10^{-5}

The input noise is a truncated Gaussian with a standard deviation of 0.02 mm. as described above for the measurement noise model. This noise level corresponds to approximately four percent of the image size in the image. Based on this noise level, the measurement error covariance matrix has diagonal elements of 0.0004 mm.^{2} for each measured variable. The off-diagonal terms are all zero. Although the dynamic model is the same for both the dual quaternion method tests and the comparison method tests, tuning and stabilization adjustments were required for both methods. Nonzero process noise quantities were needed even though the dynamic model assumed no process uncertainty. The resulting process noise and initial error covariance diagonal are given in the table. These values represent experimentally determined optimum values for the assumed dynamic model. Stability was a significant problem for the reference method with this level of noise. Estimate errors occasionally became unbounded when no process noise was used. Stability was not as great a concern for the dual quaternion method since the errors did not become unbounded, reaching approximately ten percent error in the z-translation parameter. Significant improvements in errors were achieved, however, by adjustment of these values. Error results are given later in this section.

Figures 5.3 through 5.15 show the results of a sample run with the four-point model where the true path is shown along with the dual quaternion estimated values and the comparison method estimation. The simulation was run for a time of 30 seconds with a measurement interval of 0.1 second. As can be seen, the estimates from both methods track the state variables reasonably well. However, the depth estimates, z and v_z, and angular velocities show larger differences, particularly as shown in the z axis, Figure 5.5; linear z-axis velocity, Figure 5.12; and the angular y-axis velocity, Figure 5.14. The dual quaternion method settles to the correct value faster than the point-based method for these cases. For linear y-axis velocity the point method has a larger overshoot before settling. Further details are shown in Figures 5.16 through 5.28.

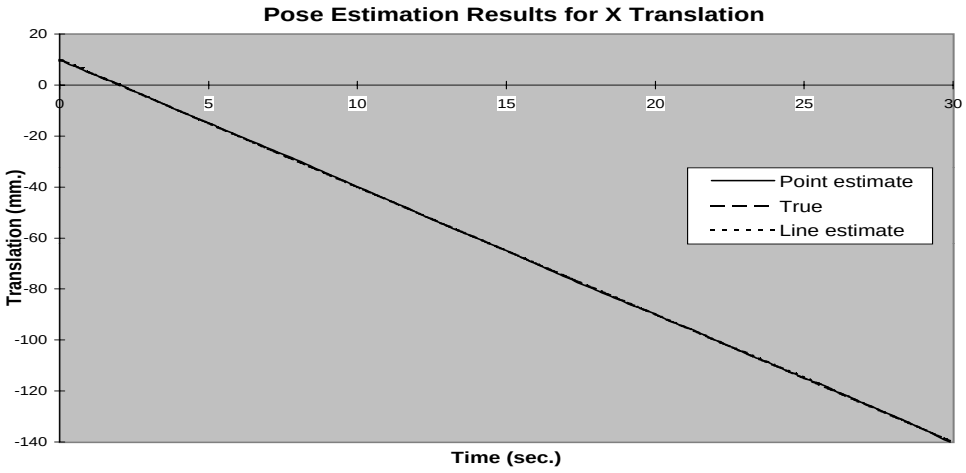

Fig. 5.3. t_x translation pose estimate simulation results with a four-point target showing linear translation over time for the line method, the comparison point method, and the true x translation value from a sample run. Both methods are seen to give good estimation results for this axis.

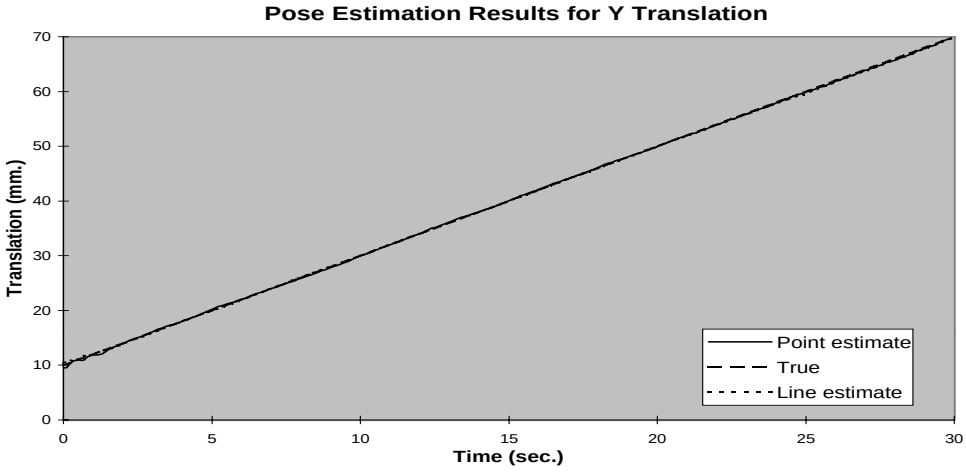

Fig. 5.4. t_y translation pose estimate simulation results with a four-point target showing linear translation over time for the line method, the comparison point method, and the true y translation value from a sample run. Both methods are seen to give good estimation results for this axis.

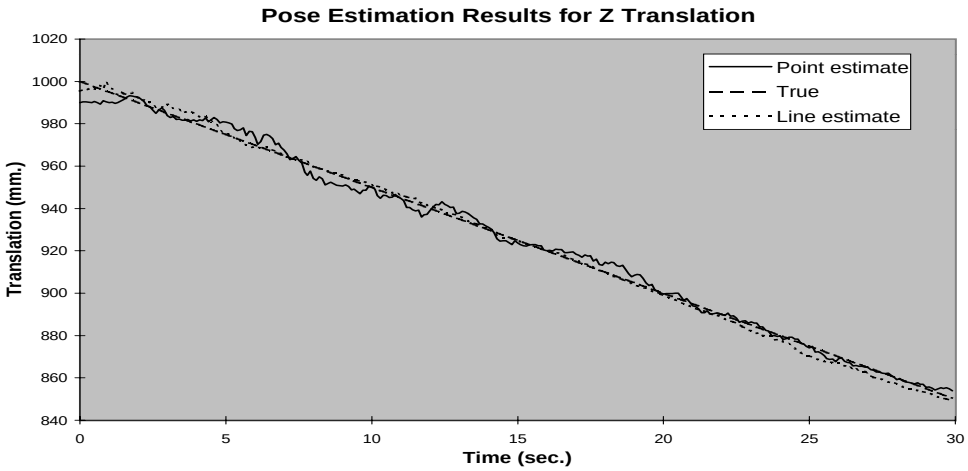

Fig. 5.5. t_z translation pose estimate simulation results with a four-point target showing linear translation over time for the line method, the comparison point method, and the true z translation value from a sample run. More deviation from the true value is seen for this axis than in x and y although the errors are not large.

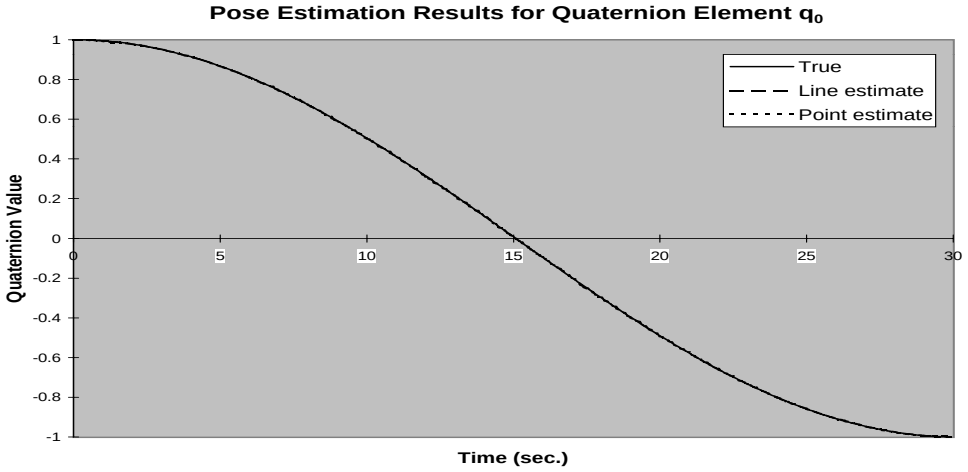

Fig. 5.6. q_0 quaternion pose estimate simulation results with a four-point target showing constant rotation over time for the line method, the comparison point method, and the true q_0 quaternion value from a sample run. Both methods are seen to give good estimation results for this state variable.

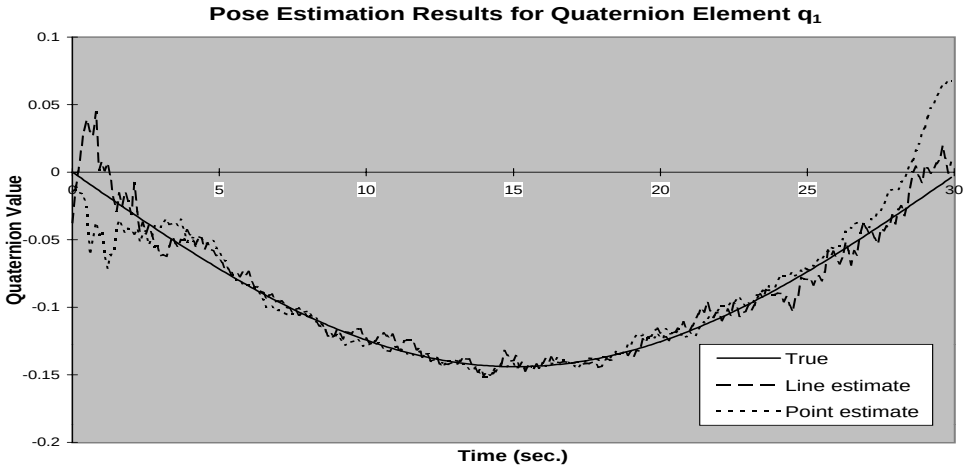

Fig. 5.7. q_1 quaternion pose estimate simulation results with a four-point target showing constant rotation over time for the line method, the comparison point method, and the true q_1 quaternion value from a sample run. More deviation from the true value is seen for this axis than in the q_0 estimate with significant errors occurring at the beginning and at the end.

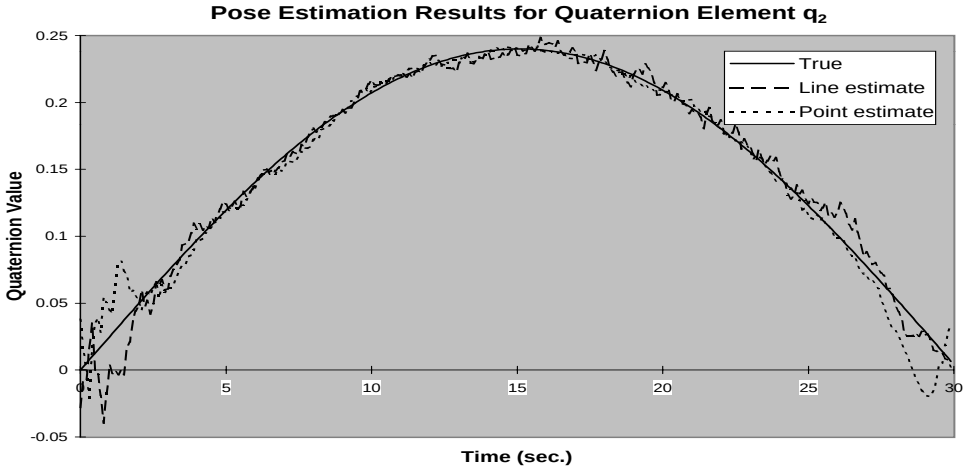

Fig. 5.8. q_2 quaternion pose estimate simulation results with a four-point target showing constant rotation over time for the line method, the comparison point method, and the true q_2 quaternion value from a sample run. More deviation from the true value is seen for this axis than in the q_0 estimate with significant errors occurring at the beginning and at the end.

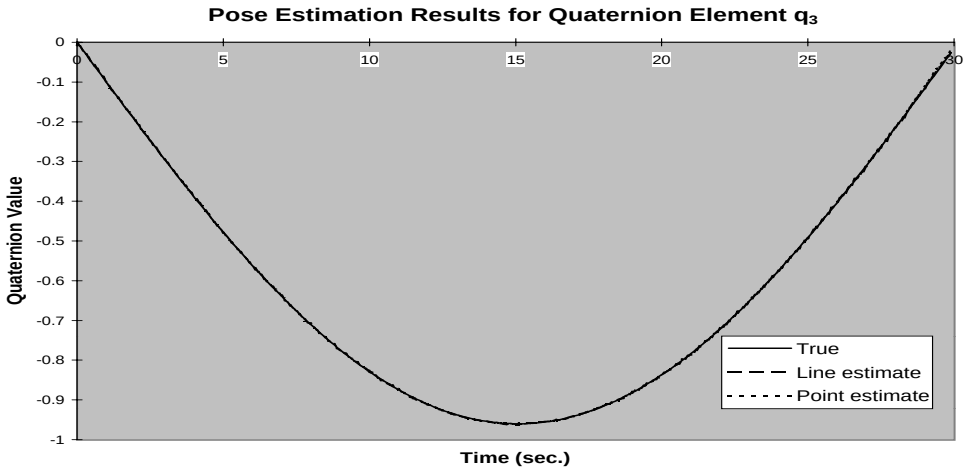

Fig. 5.9. q_3 quaternion pose estimate simulation results with a four-point target showing constant rotation over time for the line method, the comparison point method, and the true q_3 quaternion value from a sample run. Both methods are seen to give good estimation results for this state variable.

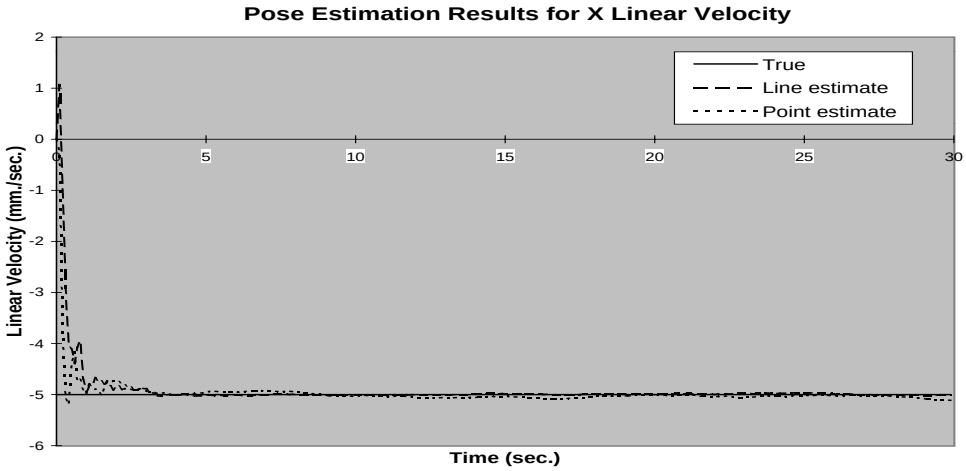

Fig. 5.10. v_x translation velocity pose estimate simulation results with a four-point target showing constant linear velocity over time for the line method, the comparison point method, and the true v_x velocity value of -5 mm./sec. from a sample run. Both methods are seen to give good estimation results for this axis after the initial settling time.

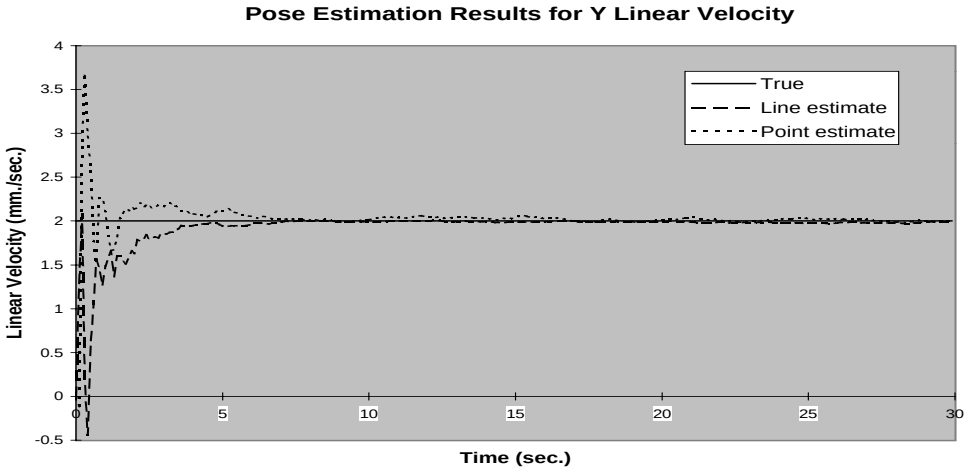

Fig. 5.11. v_y translation velocity pose estimate simulation results with a four-point target showing constant linear velocity over time for the line method, the comparison point method, and the true v_y velocity value of 2 mm./sec. from a sample run. Both methods are seen to give good estimation results for this axis after the initial settling time.

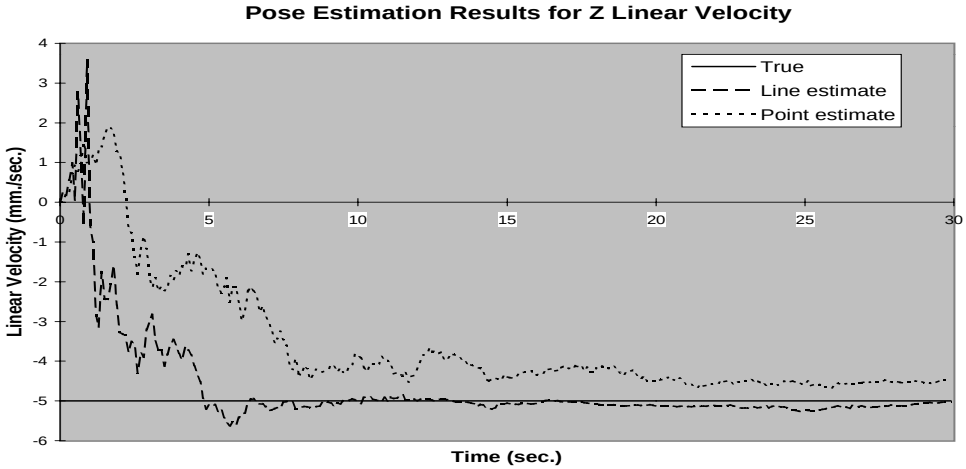

Fig. 5.12. v_z translation velocity pose estimate simulation results with a four-point target showing constant linear velocity over time for the line method, the comparison point method, and the true v_z velocity value of -5 mm./sec. from a sample run. The line method estimates are seen to settle faster to the true value and with less error than the point method.

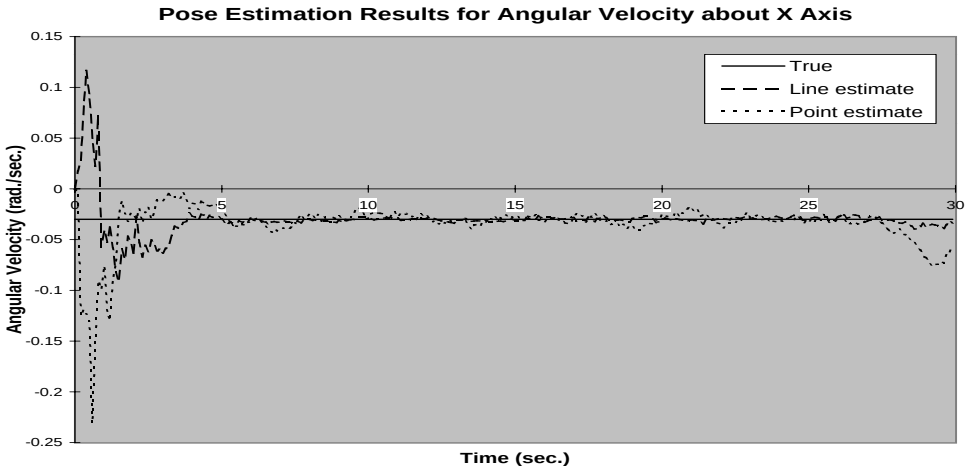

Fig. 5.13. ω_x angular velocity pose estimate simulation results with a four-point target showing constant angular velocity over time for the line method, the comparison point method, and the true ω_x angular velocity value of -0.03 rad./sec. from a sample run. Both methods are seen to give good estimation results for this rotation angle after the initial settling time, although, higher divergence is present for the point method near the end of the simulation.

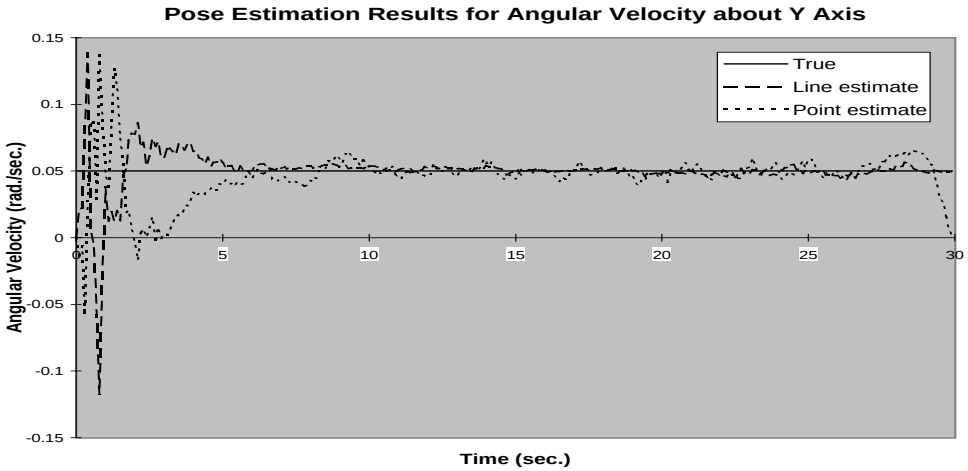

Fig. 5.14. ω_y angular velocity pose estimate simulation results with a four-point target showing constant angular velocity over time for the line method, the comparison point method, and the true ω_y angular velocity value of 0.05 rad./sec. from a sample run. Both methods are seen to give good estimation results for this rotation angle after the initial settling time, although, higher divergence is present for the point method near the end of the simulation.

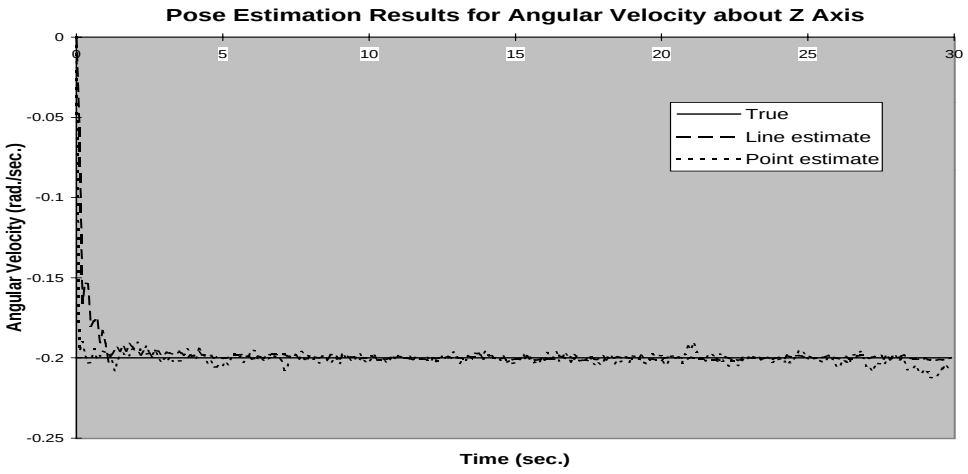

Fig. 5.15. ω_z angular velocity pose estimate simulation results with a four-point target showing constant angular velocity over time for the line method, the comparison point method, and the true ω_z angular velocity value of -0.2 rad./sec. from a sample run. Both methods are seen to give good estimation results for this rotation angle after the initial settling time.

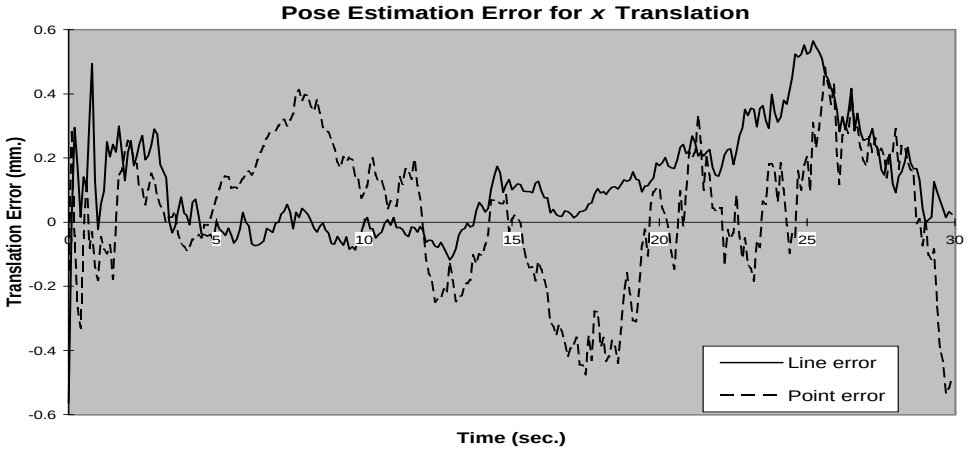

Fig. 5.16. t_x translation pose estimate simulation results for a four-point target showing linear translation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time.

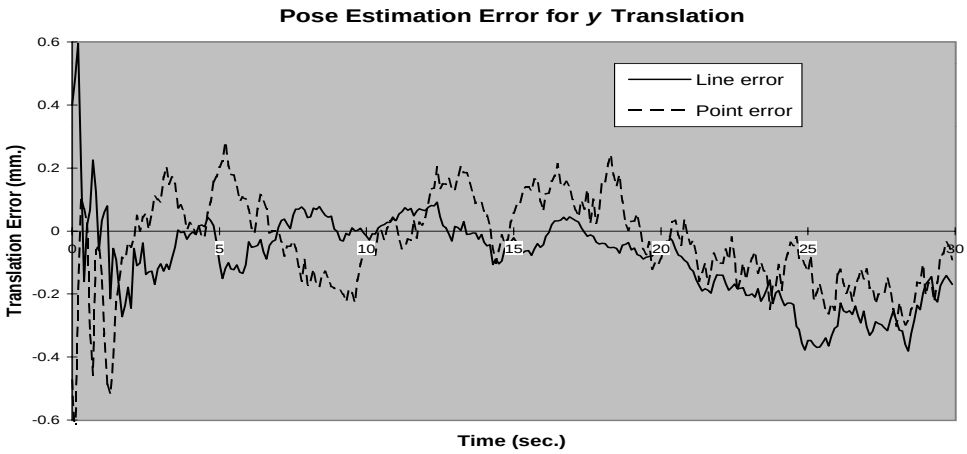

Fig. 5.17. t_y translation pose estimate simulation results for a four-point target showing linear translation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time.

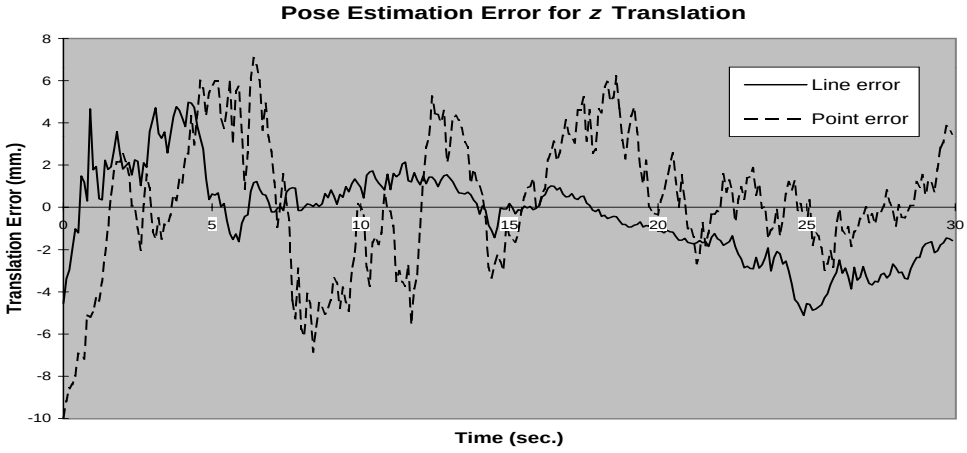

Fig. 5.18. t_z translation pose estimate simulation results for a four-point target showing linear translation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is larger than for the x and y translations. Response time is faster for the line method with somewhat smaller errors over the simulation time.

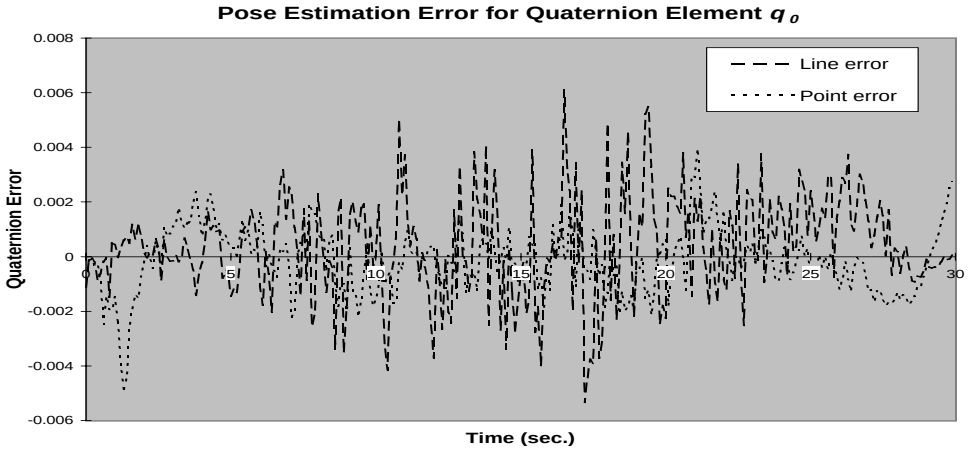

Fig. 5.19. q_0 quaternion pose estimate simulation results for a four-point target showing rotation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time.

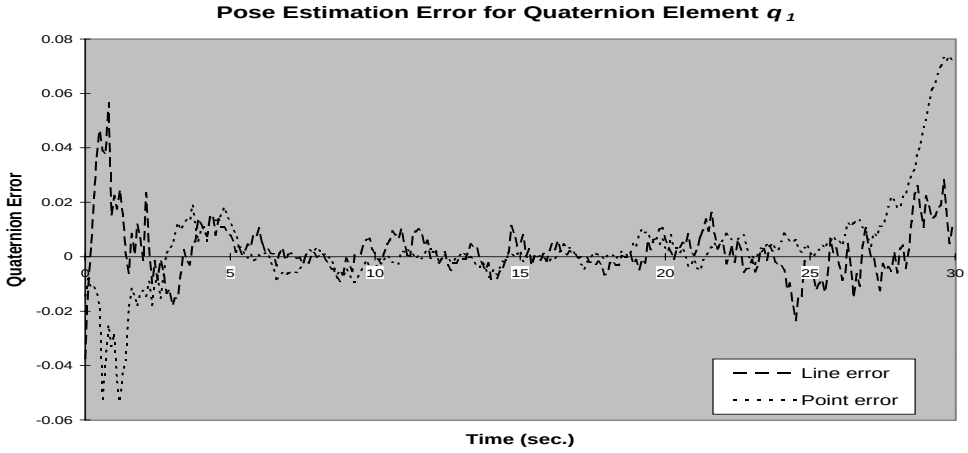

Fig. 5.20. q_1 quaternion pose estimate simulation results for a four-point target showing rotation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time except at the end where the point method has somewhat larger error.

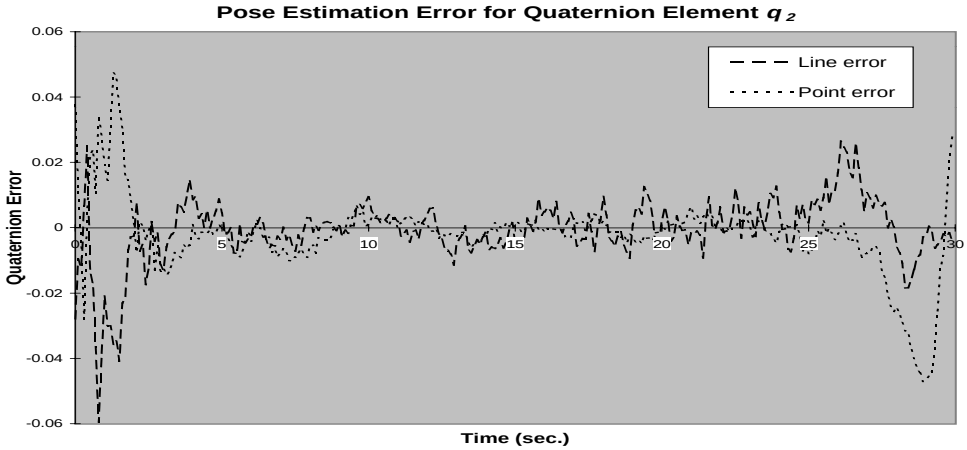

Fig. 5.21. q_2 quaternion pose estimate simulation results for a four-point target showing rotation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time except at the end where the point method has somewhat larger error.

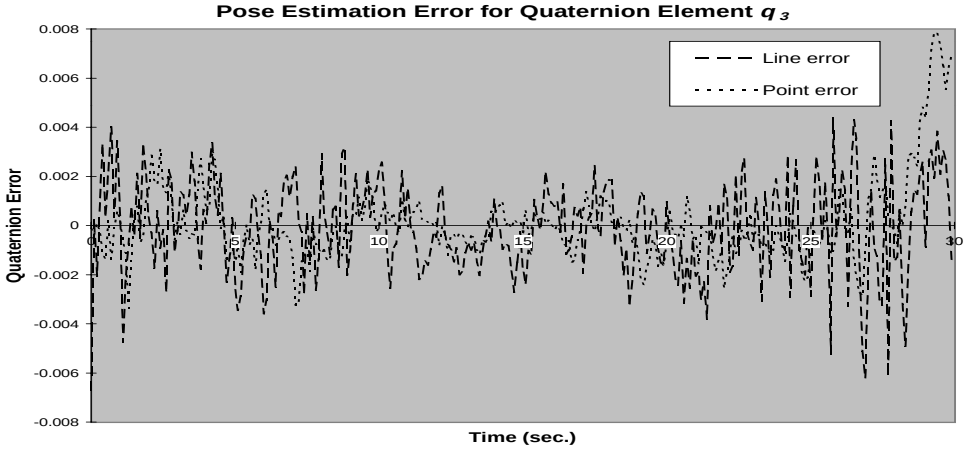

Fig. 5.22. q_3 quaternion pose estimate simulation results for a four-point target showing rotation estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small with similar errors over the simulation time.

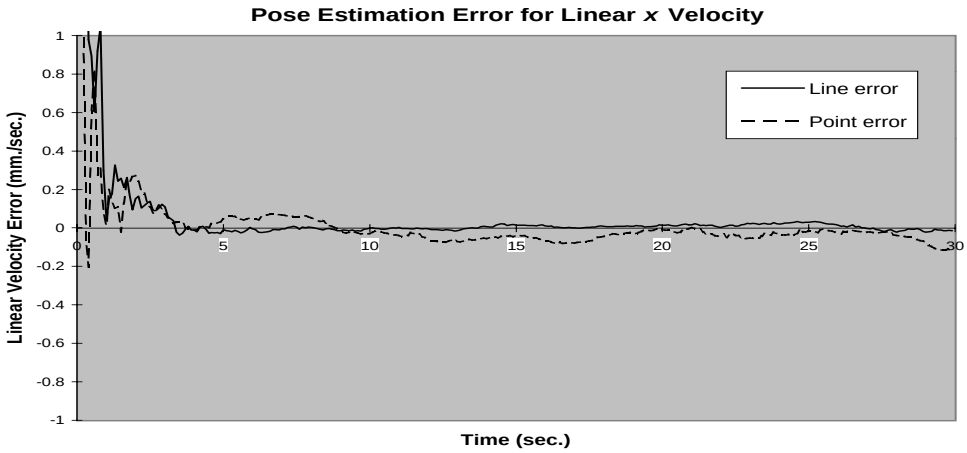

Fig. 5.23. v_x translation velocity pose estimate simulation results for a four-point target showing linear translation velocity estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small; the line method errors are somewhat less over the simulation time.

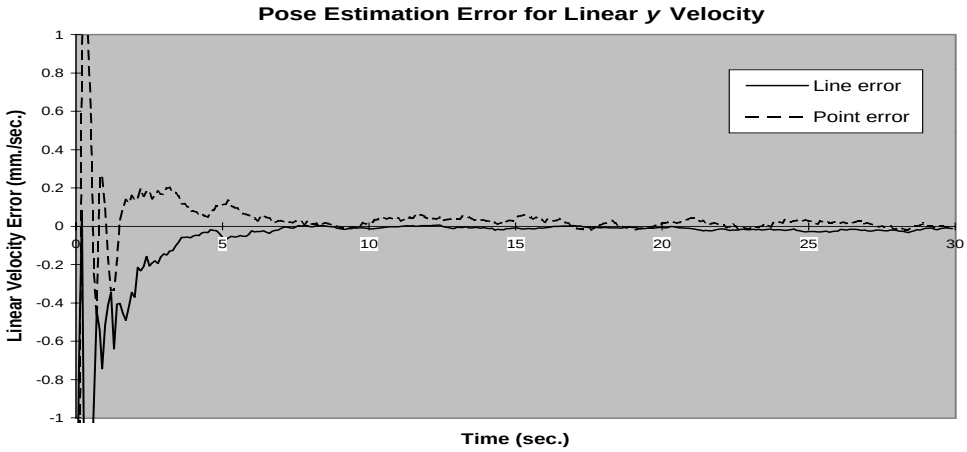

Fig. 5.24. v_y translation velocity pose estimate simulation results for a four-point target showing linear translation velocity estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be small; the line method errors are somewhat less over the simulation time.

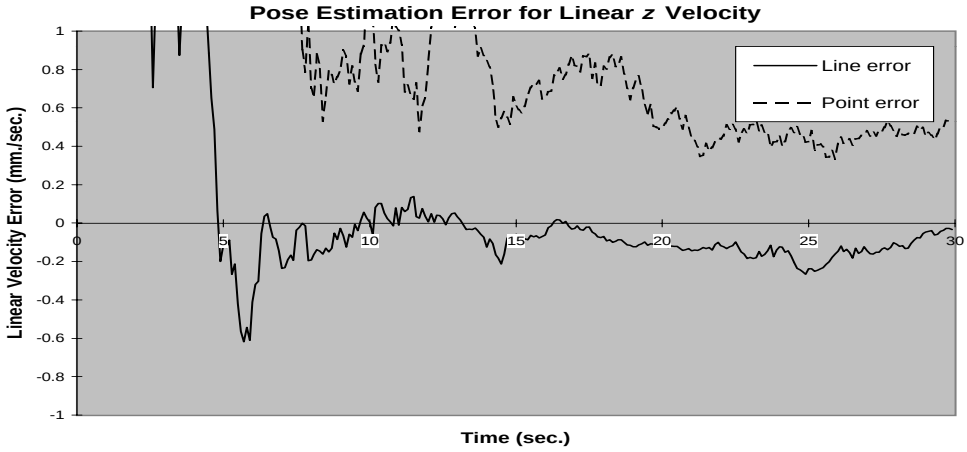

Fig. 5.25. v_z translation velocity pose estimate simulation results for a four-point target showing linear translation velocity estimation error over time for the line method and the comparison point method from a sample run. The deviation for both methods is seen to be larger than for the x and y velocity estimates. The line method has faster settling time with somewhat lower errors than the point method over the simulation time.

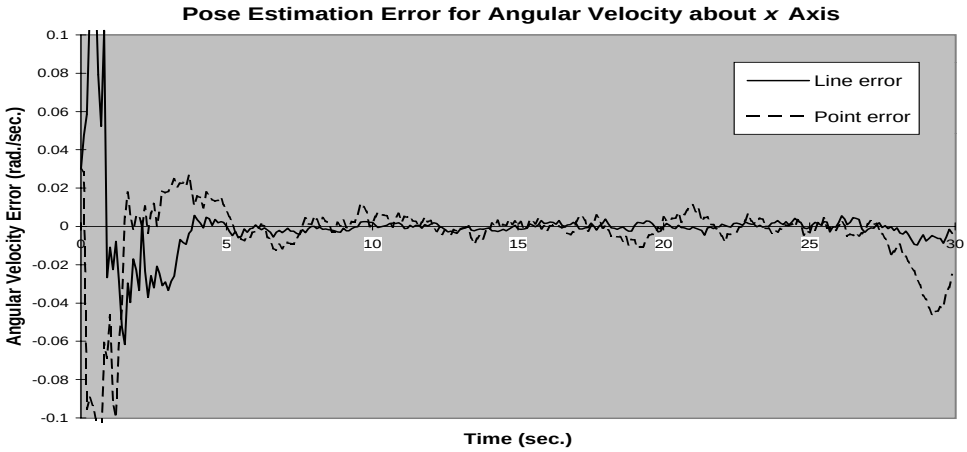

Fig. 5.26. ω_x angular velocity pose estimate simulation results for a four-point target showing constant angular velocity error over time for the line method and the comparison point method from a sample run. The deviation for both methods is relatively large compared with the true velocity, although, the mean error is near zero.

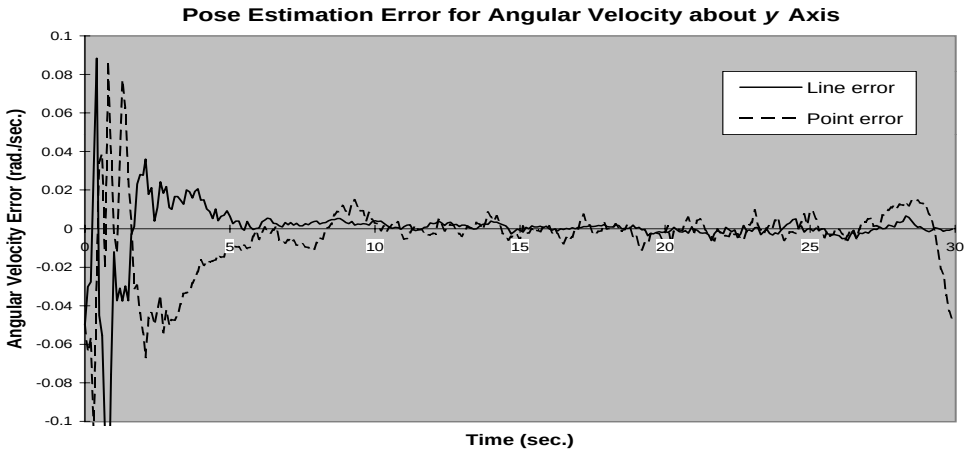

Fig. 5.27. ω_y angular velocity pose estimate simulation results for a four-point target showing constant angular velocity error over time for the line method and the comparison point method from a sample run. The deviation for both methods is relatively large compared with the true velocity, although, the mean error is near zero.

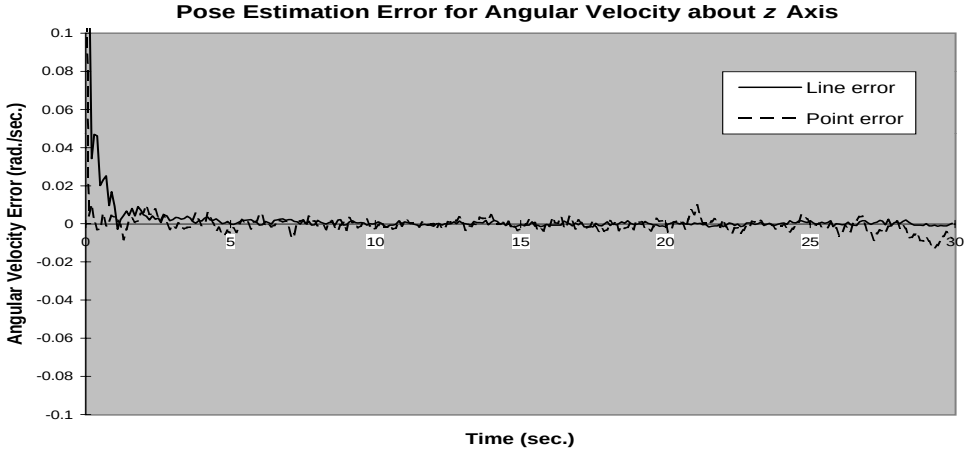

Fig. 5.28. ω_z angular velocity pose estimate simulation results for a four-point target showing constant angular velocity error over time for the line method and the comparison point method from a sample run. The deviation for both methods is small compared with the true velocity over the simulation time.

These figures give the sample estimation errors over the same period of time from the same sample data. For the z-translation error, the initial convergence is seen to be faster than the point-based method with somewhat less error over time. The velocity error for linear z is significantly higher for the point method. The above results are for a single representative run and are not necessarily typical. A further measure of performance is the error over a large number of trials. Using the same initial conditions given in Table 5.1, a series of tests were performed to experimentally measure the error. Figures 5.29 through 5.41 show the calculated RMS error over 100 sample runs for both the dual quaternion line method and the quaternion point method. The difference between the two methods is shown more clearly here since in almost all state variables, the dual quaternion method has lower RMS error over the test time interval. The square root of the corresponding mean diagonal element from the calculated covariance error matrix is also shown in each figure. This matrix in the linear Kalman filter is the predicted error covariance and depends only on the initial conditions and noise models. In the nonlinear EKF, the covariance matrix depends on the measurements and does not necessarily represent the actual error covariance [92]. In these tests, the covariance matrix values correspond, generally, closely to the actual error values for the dual quaternion method while large differences are present for the point method. Specific anomalies are present in the quaternion variables. Near the end of the time sequence, the dual quaternion RMS error shows a relatively large peak not present in the previous data, which are slowly changing and have very low error.

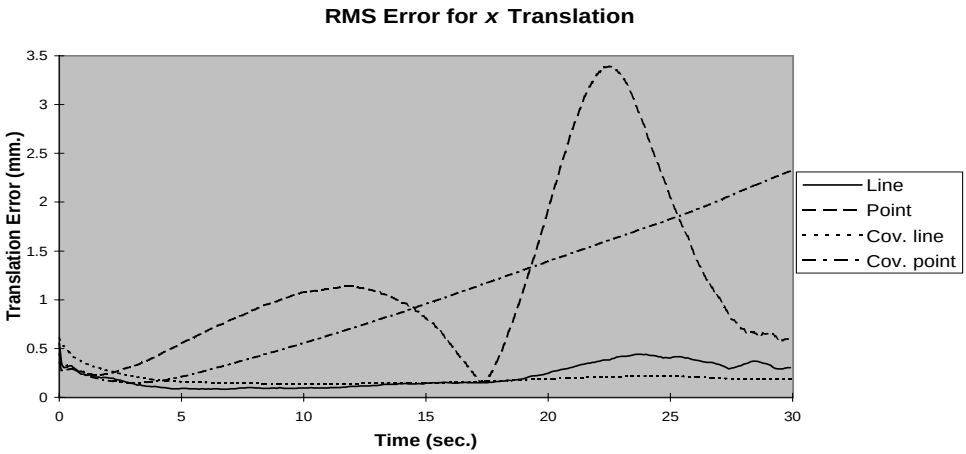

Fig. 5.29. t_x translation pose estimate simulation results for a four-point target showing linear translation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over most of the time interval. The Kalman filter predicted error is close to the actual error for the line method.

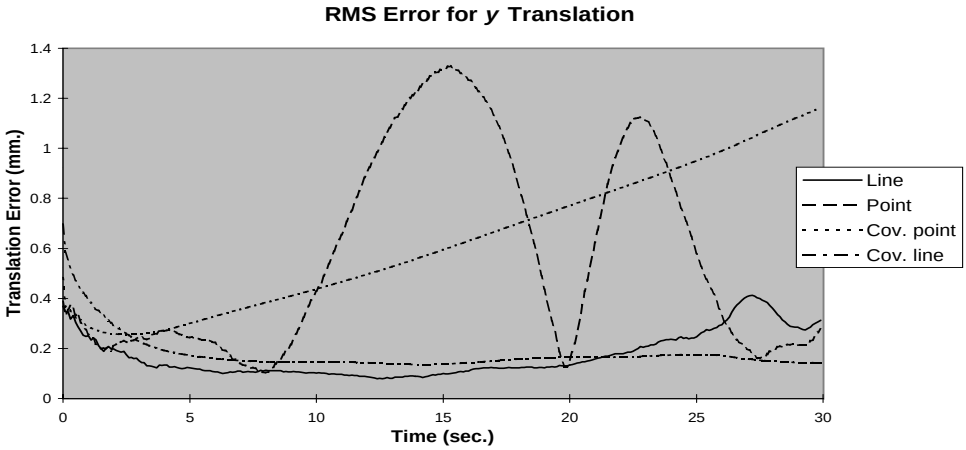

Fig. 5.30. t_y translation pose estimate simulation results for a four-point target showing linear translation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over most of the time period. The Kalman filter predicted error is close to the actual error for the line method.

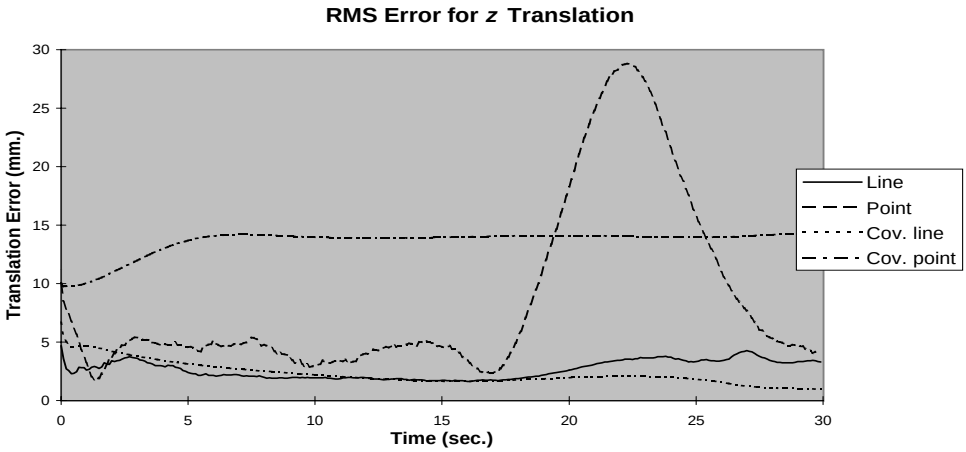

Fig. 5.31. t_z translation pose estimate simulation results for a four-point target showing linear translation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over the last half of the simulated time period. The Kalman filter predicted error is close to the actual error for the line method.

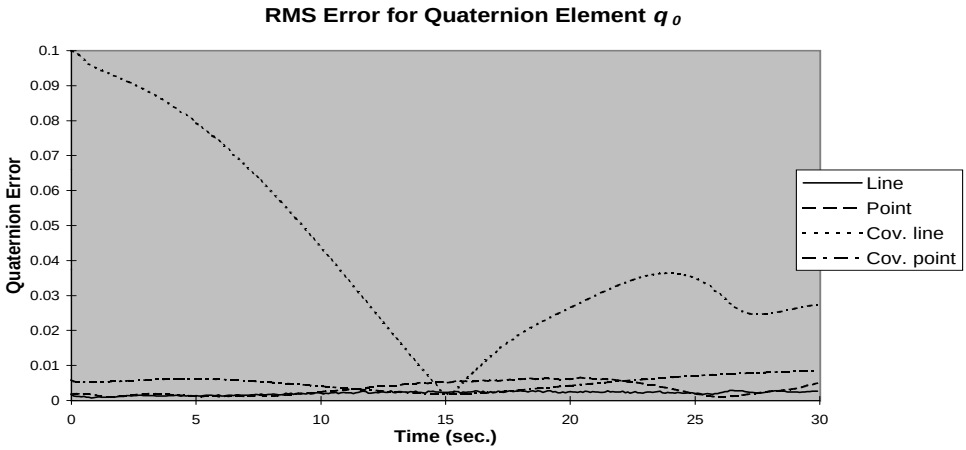

Fig. 5.32. q_0 quaternion pose estimate simulation results for a four-point target showing rotation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be small for both methods over the simulated time period. The Kalman filter predicted error for the line method greatly overestimates the actual error.

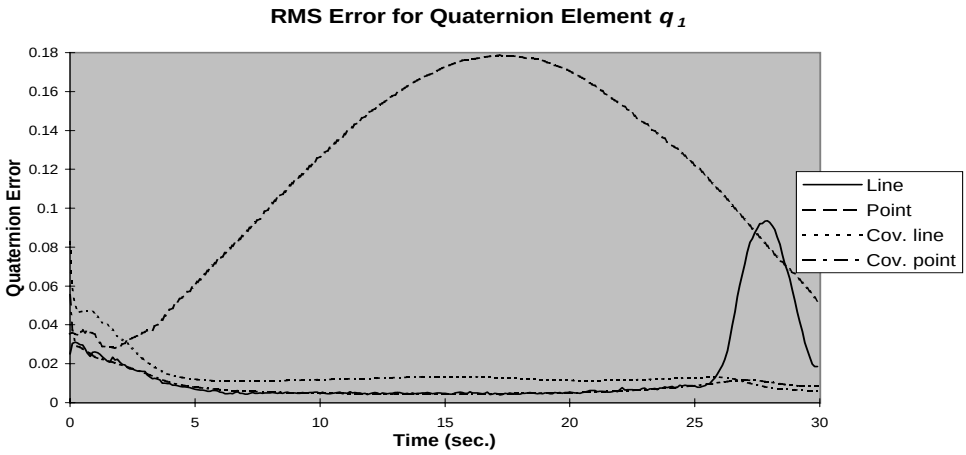

Fig. 5.33. q_1 quaternion pose estimate simulation results for a four-point target showing rotation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be considerably larger for the point method than the line method over the simulated time period. The Kalman filter predicted error for the point method greatly underestimates the actual error while the predicted error for the line method is close to the actual error except near the end of the time period.

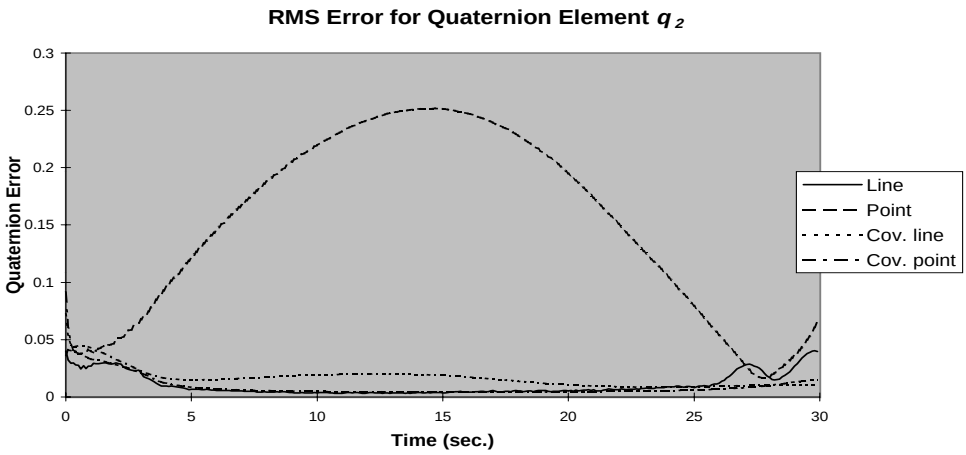

Fig. 5.34. q_2 quaternion pose estimate simulation results for a four-point target showing rotation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be considerably larger for the point method than the line method over the simulated time period. The Kalman filter predicted error for the point method greatly underestimates the actual error while the predicted error for the line method is close to the actual error except near the end of the time period.

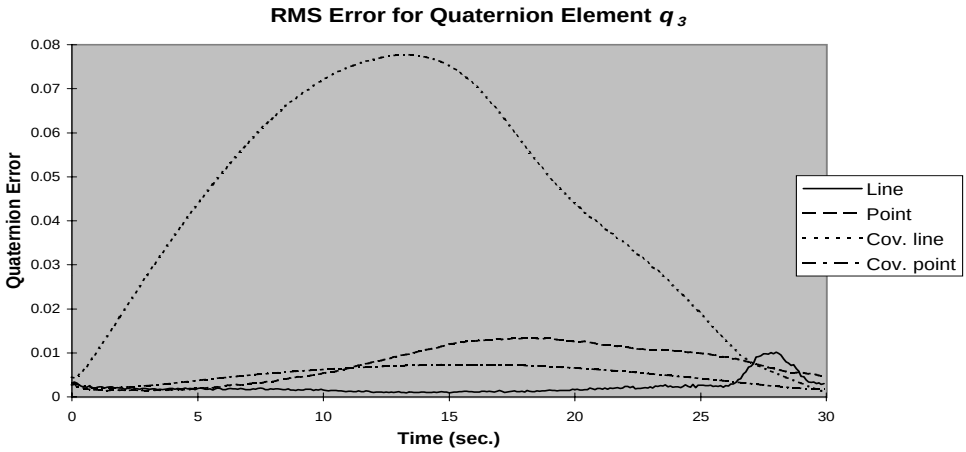

Fig. 5.35. q_3 quaternion pose estimate simulation results for a four-point target showing rotation root mean square estimation error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be small for both methods over the simulated time period. The Kalman filter predicted error for the line method greatly overestimates the actual error.

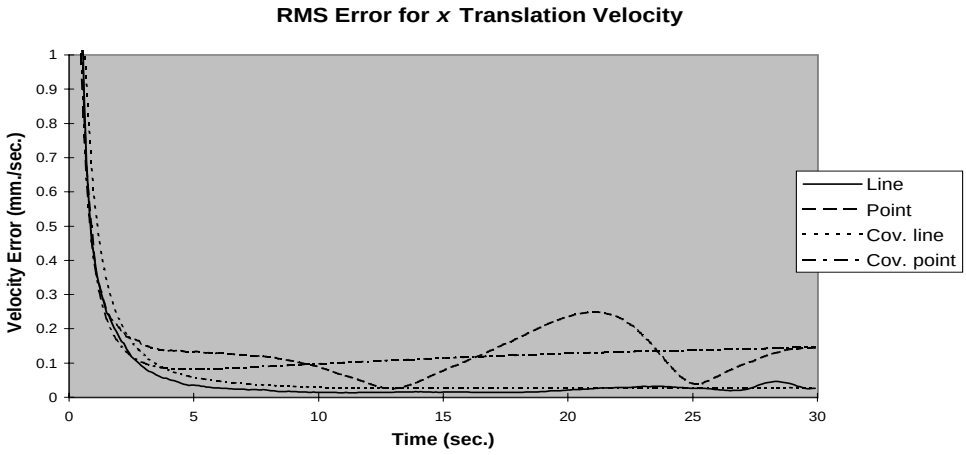

Fig. 5.36. v_x translation velocity pose estimate simulation results for a four-point target showing linear translation root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over most of the time period. The Kalman filter predicted error is close to the actual error for the line method.

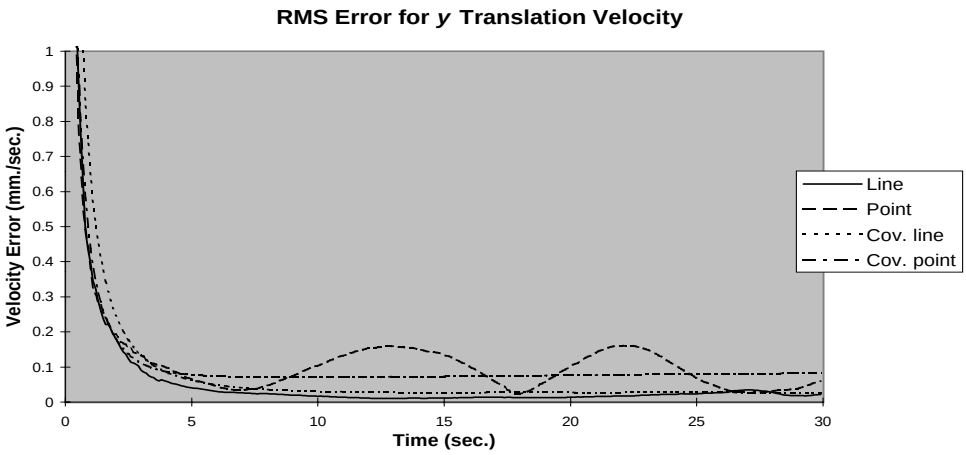

Fig. 5.37. v_y translation velocity pose estimate simulation results for a four-point target showing linear translation root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over most of the time period. The Kalman filter predicted error is close to the actual error for the line method.

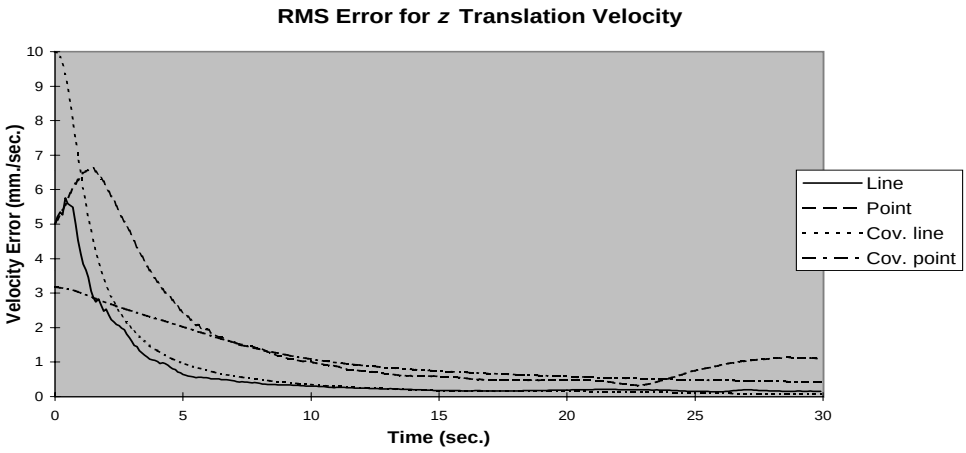

Fig. 5.38. v_z translation velocity pose estimate simulation results for a four-point target showing linear translation root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method, particularly during the initial settling time. The Kalman filter predicted error is close to the actual error for the line method but considerably underestimates the actual error for the point method.

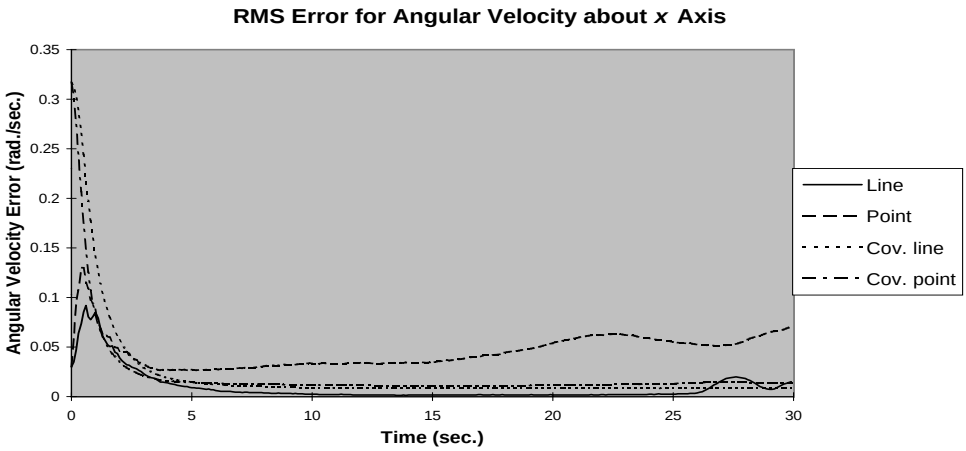

Fig. 5.39. ω_x angular velocity pose estimate simulation results for a four-point target showing angular root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over the latter portion of the simulation time. The Kalman filter predicted error is close to the actual error for the line method but considerably underestimates the actual error for the point method.

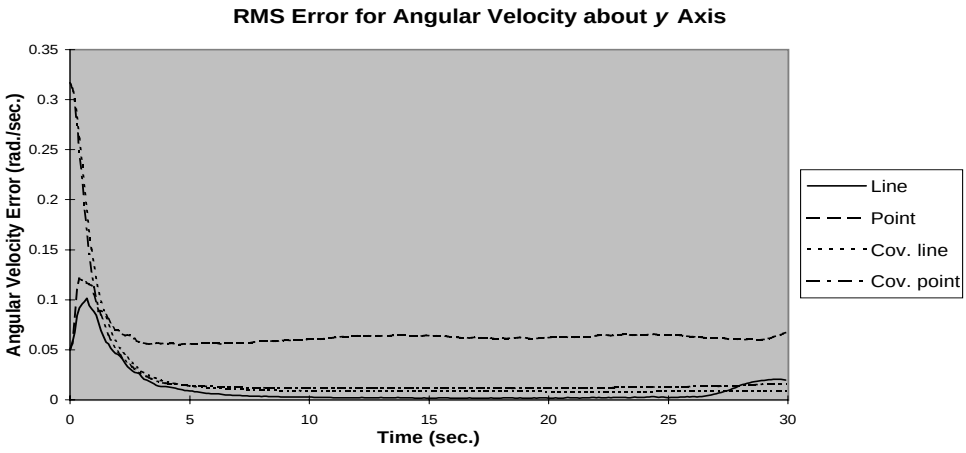

Fig. 5.40. ω_y angular velocity pose estimate simulation results for a four-point target showing angular root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over the latter portion of the simulation time. The Kalman filter predicted error is close to the actual error for the line method but considerably underestimates the actual error for the point method.

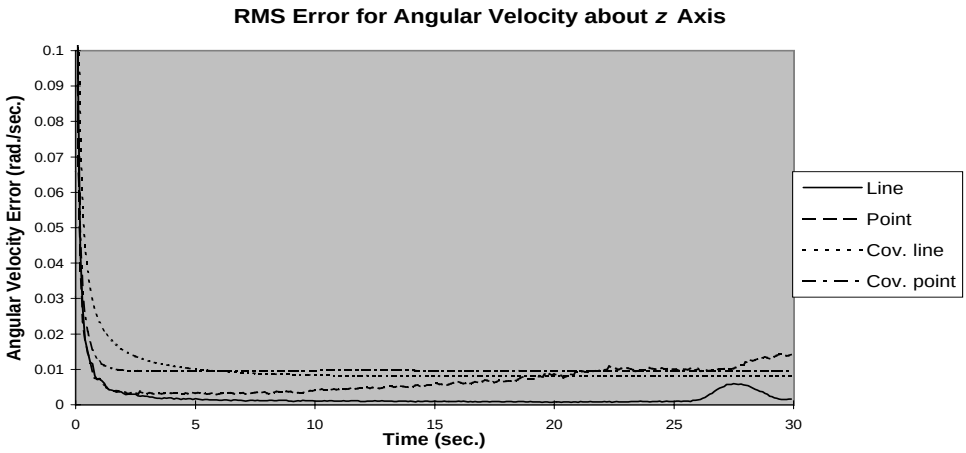

Fig. 5.41. ω_z angular velocity pose estimate simulation results for a four-point target showing angular root mean square estimation velocity error over time for the line method and the comparison point method calculated from 100 sample runs. The square root of the predicted error covariance from the Kalman filter for both methods is also shown. The root mean square error is seen to be significantly greater for the point method than for the line method over the latter portion of the simulation time. The Kalman filter predicted error is close to the actual error for the line method and for the point method.

5.1.10 Simulation Using Adaptive Noise Estimation

A simulation has been performed to compare the results using the adaptive measurement noise method in the IEKF with the nonadaptive version using a constant measurement covariance matrix. This simulation used a four-point target with a large offset from the center of the image to show the effects of larger noise covariance in the line point measurements. The point method was also simulated to compare the results for the off-center object.

Table 5.3 gives the initial state conditions for this adaptive method simulation with a four-point geometric model. Initial states are the same for all methods. Table 5.4 gives the assumed initial error covariance diagonal terms and the process noise covariance diagonal terms for both the adaptive and nonadaptive dual quaternion method and for the point-based method used for comparison. In all cases, the off-diagonal terms of the error covariance matrices are zero. For these tests, a simulated sample interval of 0.1 second is used. Note that the process noise covariance is the same for both dual quaternion methods and is different for the point method. Both process noise and measurement noise covariances were held constant for the nonadaptive dual quaternion method and for the point method during the simulation while the measurement noise covariance terms were varied based on the image line locations using the approximate real-time algorithm given in Section 4.2.

The input noise is a truncated Gaussian with a standard deviation of 0.02 mm. as described above for the measurement noise model. This noise level corresponds to approximately four percent of the image size in the image. Based on this noise level, the initial measurement error covariance matrix has diagonal elements of 0.0004 mm.^{2} for each measured variable. The off-diagonal terms are all zero. The dynamic model is the same constant linear velocity and constant angular velocity for both the adaptive and nonadaptive dual quaternion method tests as well as for the point comparison method

Table 5.3.
Actual and assumed initial states of the dual quaternion and point-based extended Kalman filter for simulation comparison with the adaptive measurement noise line method, the nonadaptive line method, and the comparison point method.

State	Translation $x\ y\ z$ (mm.)	Quaternion $q_0\ q_1\ q_2\ q_3$	Linear Velocity $x\ y\ z$ (mm./sec.)	Rotational Velocity $x\ y\ z$ (rad./sec.)
True Initial State	400 400 1000	1 0 0 0	-5 2 -5	.01 -.02 -.1
Initial State Estimate	390 390 990	.9998 .01 .01 .01	0 0 0	0 0 0

Table 5.4.
Dual quaternion method and point method initial error covariance matrix and process noise matrix diagonal terms of the extended Kalman filter for simulation comparison with the adaptive measurement noise line method, the nonadaptive line method, and the comparison point method.

Noise Covariance	Translation $x\ y\ z$ (mm.^2)	Quaternion $q_0\ q_1\ q_2\ q_3$	Linear Velocity $x\ y\ z$ $(\text{mm./sec.})^2$	Rotational Velocity $x\ y\ z$ $(\text{rad./sec.})^2$
Dual Quaternion Error Covariance	100	0.01	100	0.01
Dual Quaternion Process Noise	10^{-5}	10^{-6}	10^{-3}	10^{-4}
Point Method Error Covariance	100	0.01	100	0.01
Point Method Process Noise	0	0	10^{-2}	10^{-3}

tests. Tuning and stabilization adjustments were required for both methods and in some cases are different for each method. The resulting process noise matrix and initial error covariance matrix diagonals are given in Table 5.4. These values represent experimentally determined optimum values for the assumed dynamic model. Stability was a significant problem for all methods with this amount of x and y translation. Estimate errors occasionally became unbounded when no process noise was used. Significant improvements in errors were achieved, however, by adjustment of these values. Figures 5.42, 5.43, and 5.44 show the RMS errors for the translation state variables t_x, t_y, and t_z, respectively. The RMS errors were calculated over 100 sample runs for each of the methods. The graphs show the results from the three simulation runs of 30 seconds. These results for each variable are similar and show that the point method gives the most accurate results over most of the simulated time with much less initial overshoot. The adaptive method has less error and less overshoot than the nonadaptive method and converges near the end of the simulation to give the least error. The results are as expected from the analysis. The large x and y translations give much higher noise variances for the line points than for the individual points. The adaptive measurement method updates the measurement covariance matrix with the calculated variances to give better results than the nonadaptive method.

5.2 Robot Arm Tests

Experimental testing has been performed using a robot arm equipped with a CCD area camera and a computer system for processing images and controlling the arm. Static tests using no relative motion are used to characterize the measurement noise as well as variation of the estimates. Tests were also performed with controlled relative motion between an object and the camera.

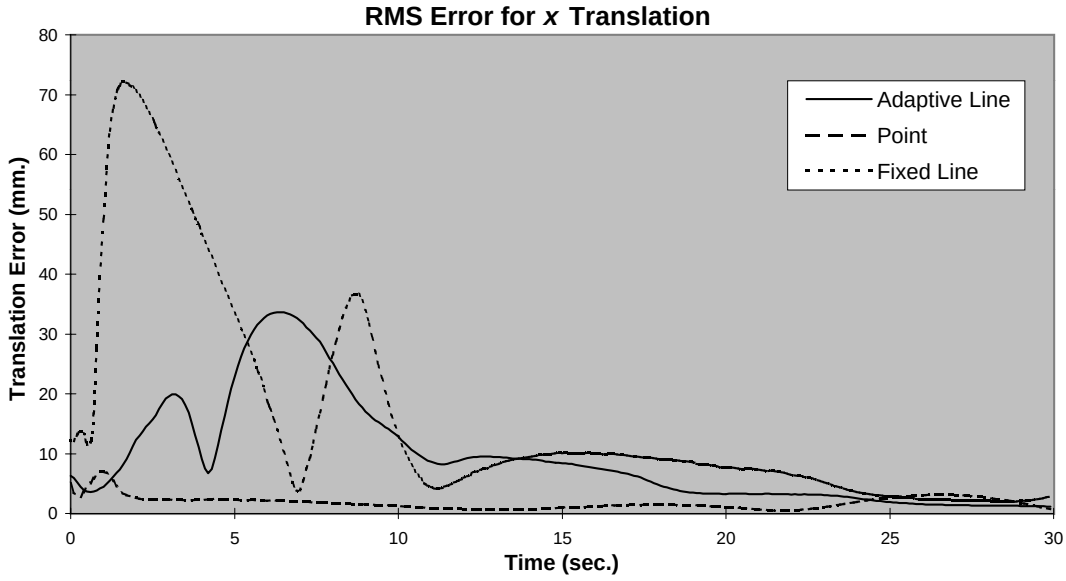

Fig. 5.42. t_x translation pose estimate simulation results for a four-point target offset from the image center. The figure shows linear translation root mean square estimation error over time for the adaptive line method, the nonadaptive line method and the comparison point method calculated from 100 sample runs. x and y translations were both initially 400 mm. The RMS error is seen to be significantly less for the point method than for the adaptive and nonadaptive line methods over most of the time period. Initial error overshoot is also much less with the point method. The adaptive line method shows less error than the nonadaptive method over most of the simulation.

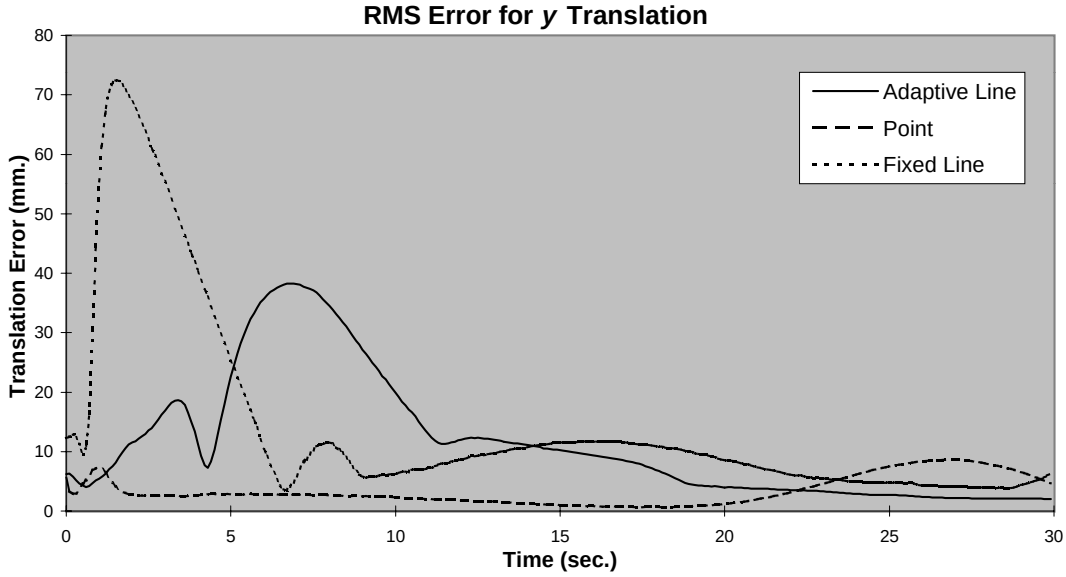

Fig. 5.43. t_y translation pose estimate simulation results for a four-point target offset from the image center. The figure shows linear translation root mean square estimation error over time for the adaptive line method, the nonadaptive line method and the comparison point method calculated from 100 sample runs. x and y translations were both initially 400 mm. The root mean square error is seen to be significantly less for the point method than for the adaptive and nonadaptive line methods over most of the time period. Initial error overshoot is also much less with the point method. The adaptive line method shows less error than the nonadaptive method over most of the simulation.

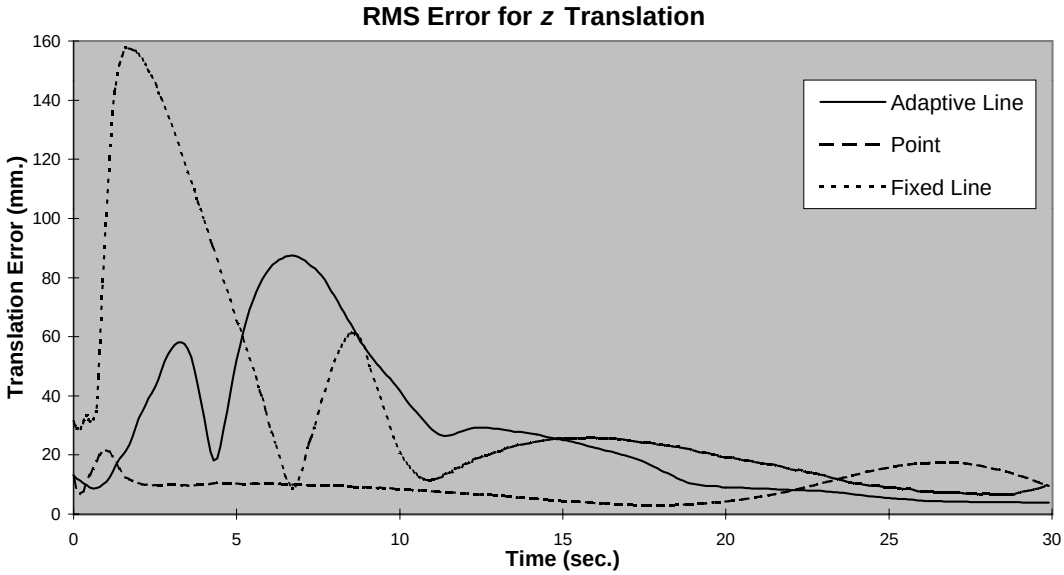

Fig. 5.44. t_z translation pose estimate simulation results for a four-point target offset from the image center. The figure shows linear translation root mean square estimation error over time for the adaptive line method, the nonadaptive line method and the comparison point method calculated from 100 sample runs. x and y translations were both initially 400 mm. The root mean square error is seen to be significantly less for the point method than for the adaptive and nonadaptive line methods over most of the time period. Initial error overshoot is also much less with the point method. The adaptive line method shows less error than the nonadaptive method over most of the simulation.

5.2.1 Experimental Setup

The experimental setup, a picture of which is shown in Figure 5.45, consists of a Mitsubishi RV-E2 six-degree-of-freedom robot arm and a Cidtec CID2250D CID camera used for image acquisition.

This camera is a progressive scan type, 30 frames per second. The image resolution is 512-by-512 square pixels where each pixel is 0.015 mm. in width. The robot arm has its own controller with a teach pendant for manually moving each of the six axes. The picture also shows the object used for the initial tests, a rectangular figure from which the edges are extracted. The pose estimation is performed on a 166 Mhz. Pentium PC. The robot is controlled remotely via a serial interface to the robot controller. A block diagram of the entire configuration is shown in Figure 5.46. The remote robot control consists of position, velocity, and acceleration commands. Low-level joint control is not accessible for this arm. As a result, high speed, dynamic control of the arm cannot be demonstrated. The functionality, however, does permit control at low update rates.

5.2.2 Dynamic Model

The system dynamic model is the same model described previously under the simulation section. Position, orientation, and their first derivatives are state estimates.

5.2.3 Measurement Model

The measurement model for the experiment consists of the functions given in Figure 5.47. First, the image of the target object is acquired by the camera and the frame grabber. Then, the object edges are extracted and the contour formed. Straight lines are calculated next from the contour with length filtering performed to eliminate short lines that could arise from extraneous features. Collinear lines are also combined into one line. The resulting lines are used as inputs to the filter after conversion to the line point

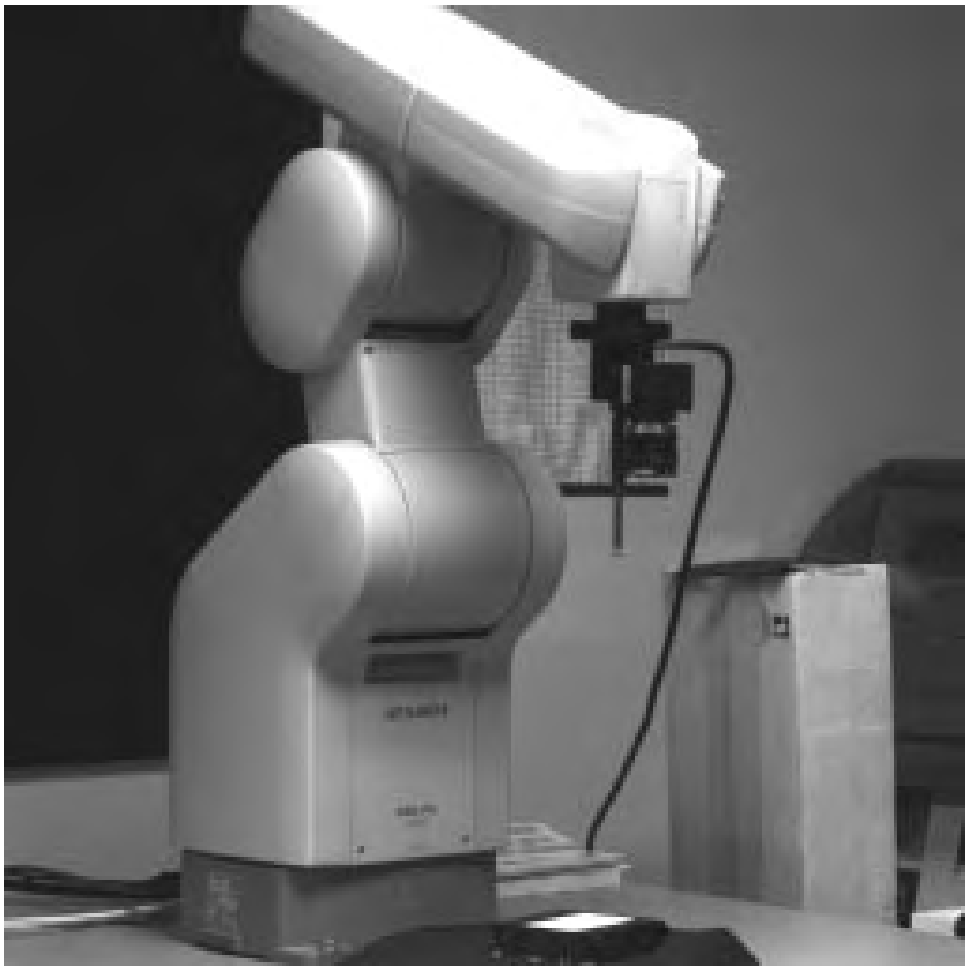

Fig. 5.45. Mitsubishi RV-E2 six-axis robot arm and the Cidtec camera used in the experimental tests.

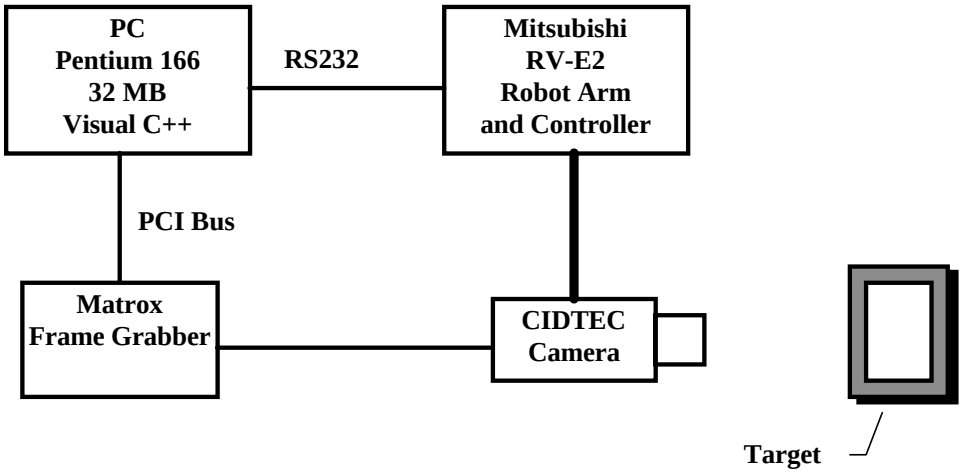

Fig. 5.46. Block diagram showing the system hardware used in the experimental setup. The hardware includes a Cidtec camera, frame grabber, PC compatible computer, and Mitsubishi robot arm and controller.

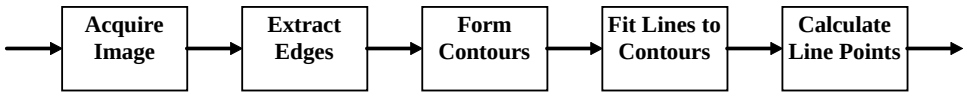

Fig. 5.47. Measurement functions applied to obtain line point measurements from the camera image in robot arm experiments.

parameterization. Correspondence to the known geometric model is established at this stage.

5.2.4 Initial Conditions

In the experimental tests, an initial estimate of the relative pose between the camera and the object is needed. This lack of knowledge is a significant difference from the simulation tests where the initial pose is known and is used as the basis from which the initial estimate is formed. To overcome this difficulty, an alternate method is used to provide an initial estimate of the pose. This method uses the four-point coplanar pose calculation method of Abidi and Chandra [4]. The initial velocities are set to zero as in the simulation tests.

5.2.5 Measurement Noise Results

Preliminary tests were performed to characterize the measurement noise present in the features extracted from the acquired image. These measurements were taken by repeatedly acquiring images of the target with no relative motion. Approximately 500 images were analyzed. The approximate distribution is given by the histograms shown in Figures 5.48 and 5.49 for a typical line point x and y coordinate.

The distribution is seen to have a single peak with no values outside an interval about the peak. The variances for all eight coordinates ranged from 2×10^{-6} to 1×10^{-5}. As expected from the results in Chapter 4, the noise approximates a Gaussian density function.

5.2.6 Target Motion Results

The tests were performed with the target moving in a circular pattern around the z axis at constant speed. A sample period is one second is used. The camera and robot arm

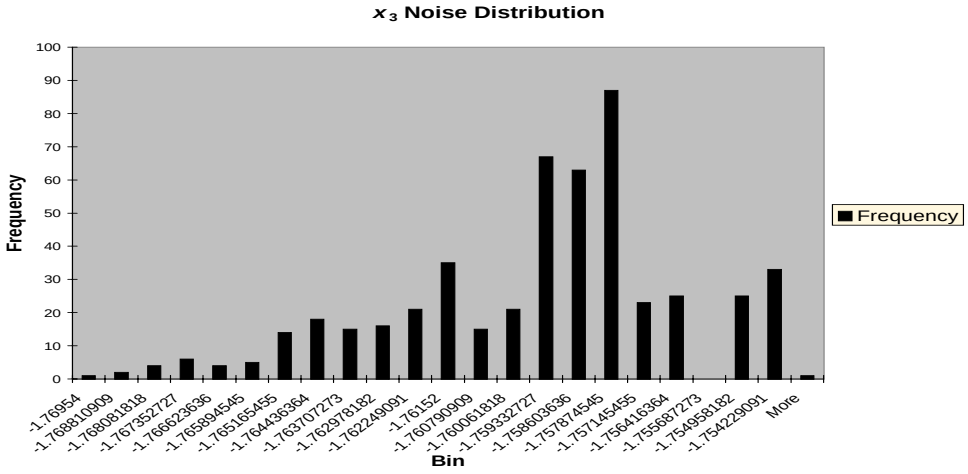

Fig. 5.48. Histogram of the distribution of x line point measurement values from 500 camera images for a stationary target.

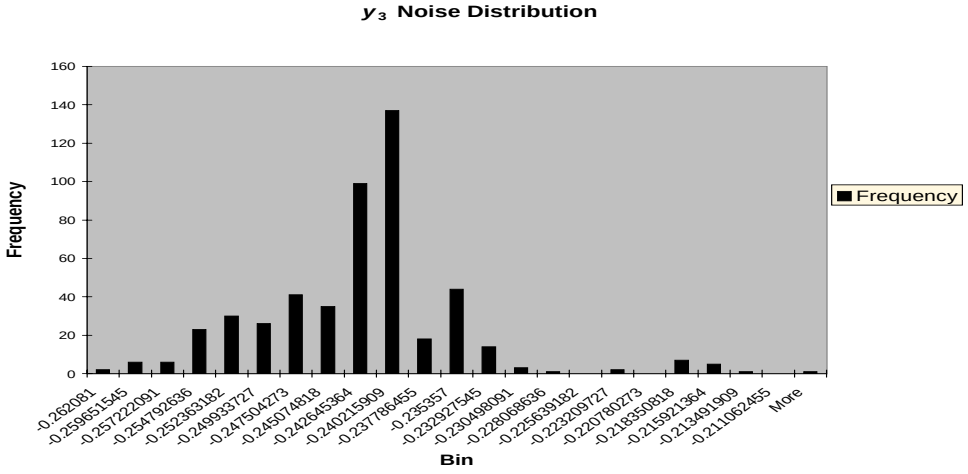

Fig. 5.49. Histogram of the distribution of y line point measurement values from 500 camera images for a stationary target.

were directly above the target and stationary. The motion is confined to the x-y plane. Figures 5.50 through 5.62 show the estimated response for each state variable over time.

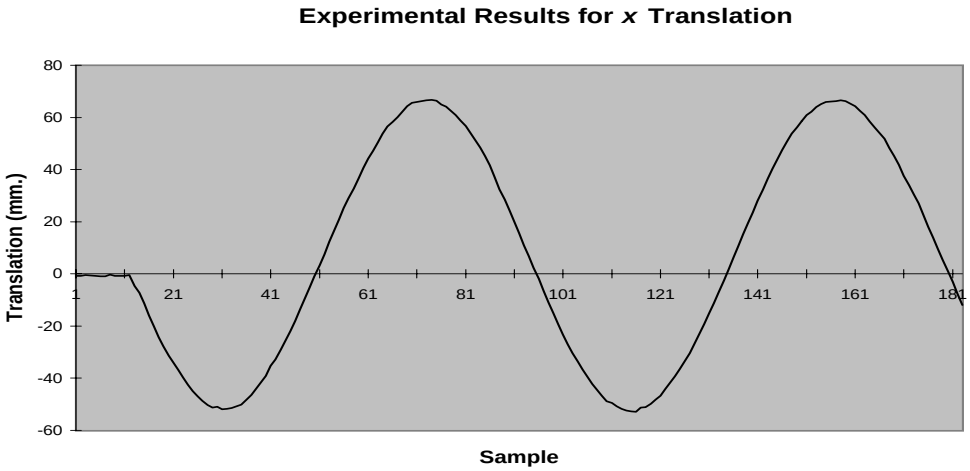

Fig. 5.50. t_x translation pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a sinusoidal variation corresponding to the true x variation of the target. Note the step response at the beginning of the target motion.

As expected, the x and y translations vary in a sinusoidal pattern consistent with constant circular motion. The z-axis position is seen to vary with a small sinusoidal component over the test interval along with apparent noise. Although the motion was nominally in the x-y plane of the target and parallel to the image plane of the camera, the setup was not calibrated precisely. Any nonparallelism would result in a small sinusoidal z variation with time. In this test, the z variation is seen to be about 5 mm. The quaternion estimates are also consistent with the actual motion. For motion about the z axis, only q_0 and q_3 should vary significantly in magnitude. The results demonstrate this since q_1 and

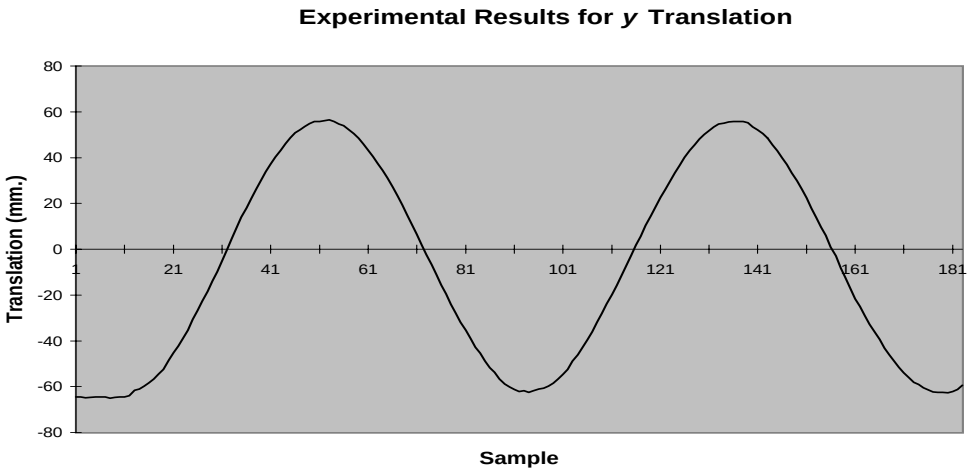

Fig. 5.51. t_y translation pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a sinusoidal variation corresponding to the true y variation of the target. Note the step response at the beginning of the target motion.

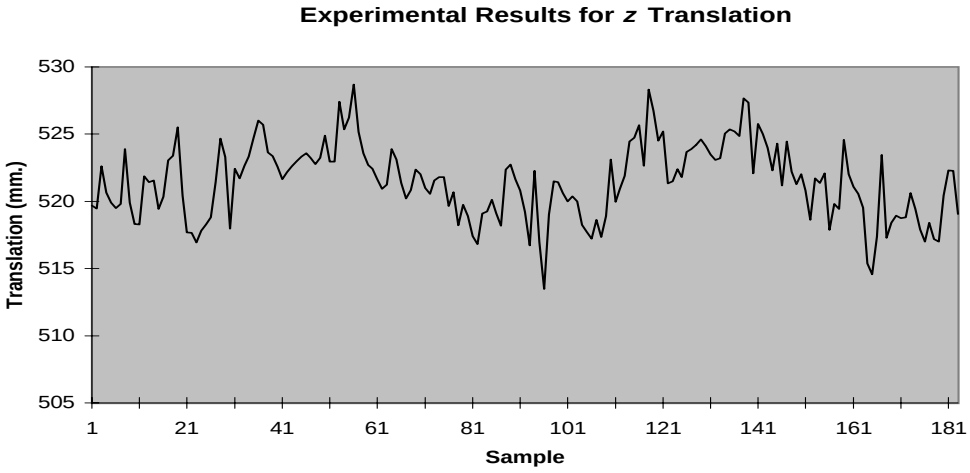

Fig. 5.52. t_z translation pose estimate from experimental testing for constant target motion about z axis. The estimate shows a noisy variation about a mean z value with a small sinusoidal component. The true target z value is nominally constant.

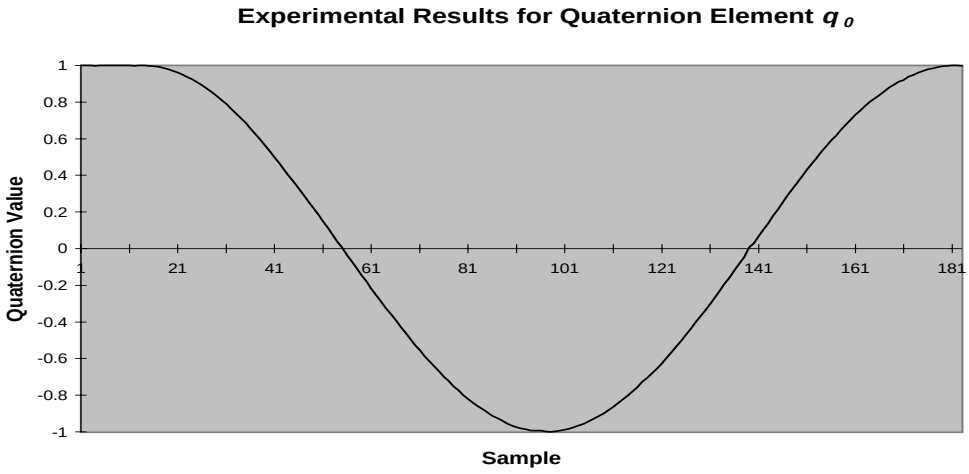

Fig. 5.53. q_0 quaternion pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a full 360 degree sinusoidal variation corresponding to the true q_0 variation of the target as it moves through two complete revolutions.

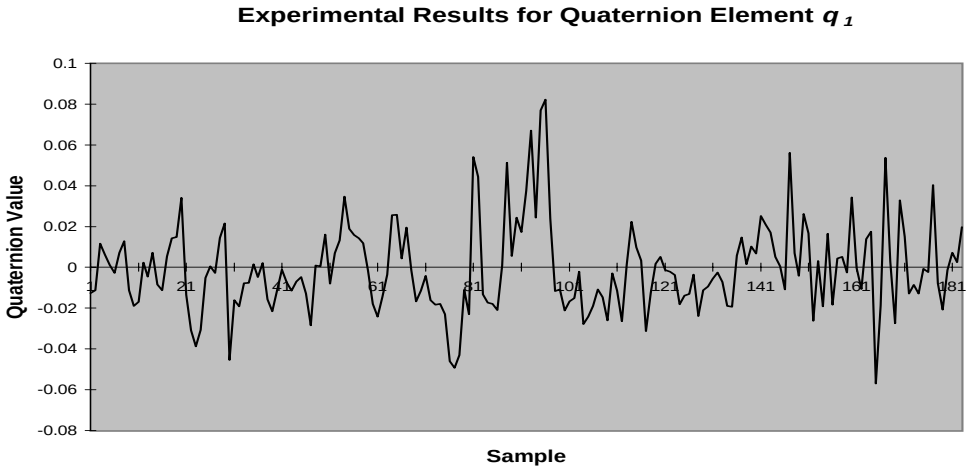

Fig. 5.54. q_1 quaternion pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a relatively small noisy variation about zero in comparison to a true q_1 constant value of zero.

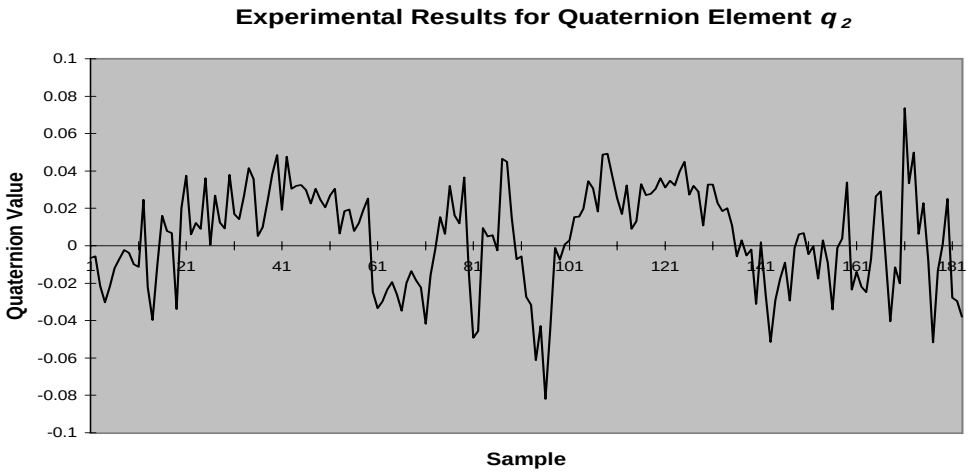

Fig. 5.55. q_2 quaternion pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a relatively small noisy variation about zero in comparison to a true q_2 constant value of zero.

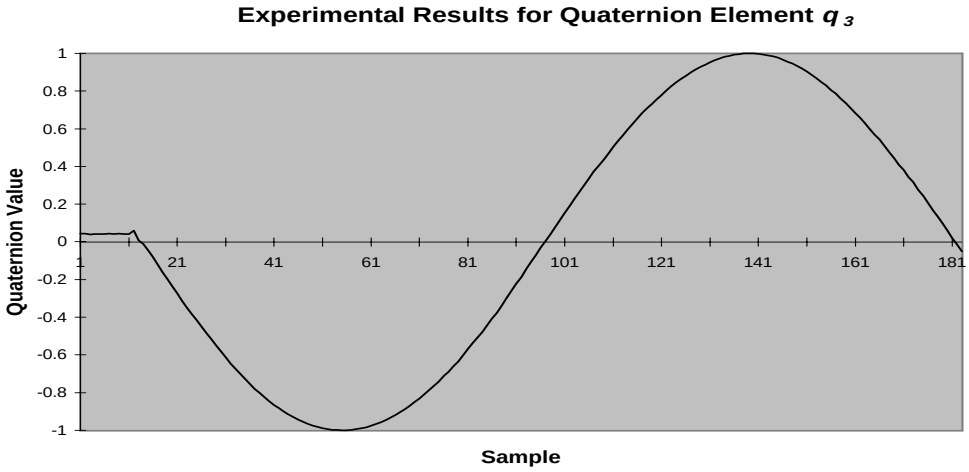

Fig. 5.56. q_3 quaternion pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a full 360 degree sinusoidal variation corresponding to the true q_3 variation of the target as it moves through two complete revolutions.

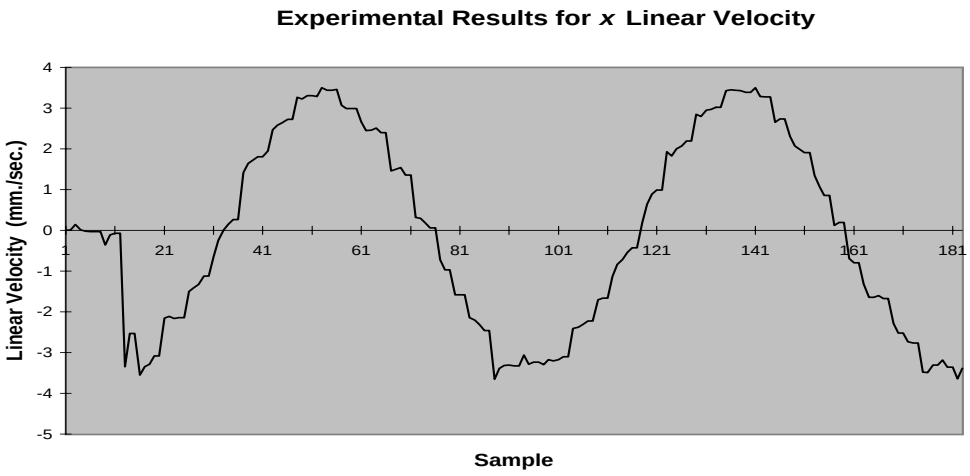

Fig. 5.57. v_x translation velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy sinusoidal variation compared to the true x sinusoidal velocity variation of the target. Note the step response at the beginning of the target motion.

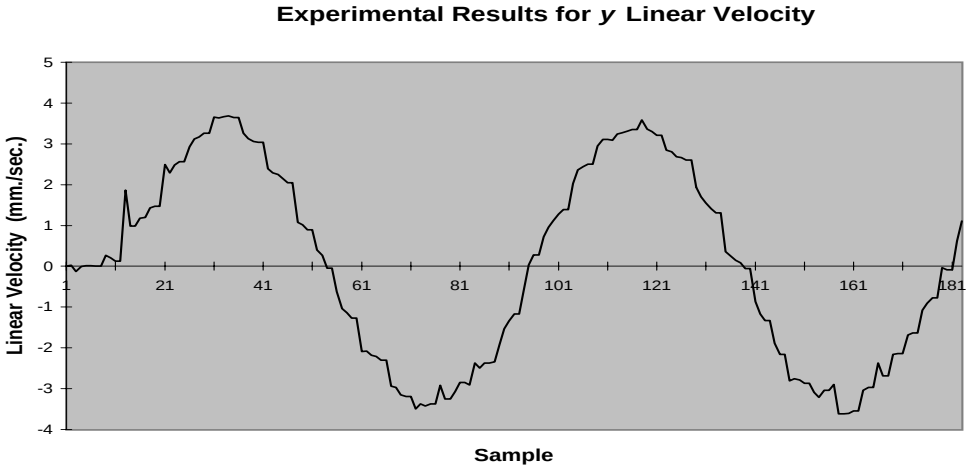

Fig. 5.58. v_y translation velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy sinusoidal variation compared to the true y sinusoidal velocity variation of the target. Note the step response at the beginning of the target motion.

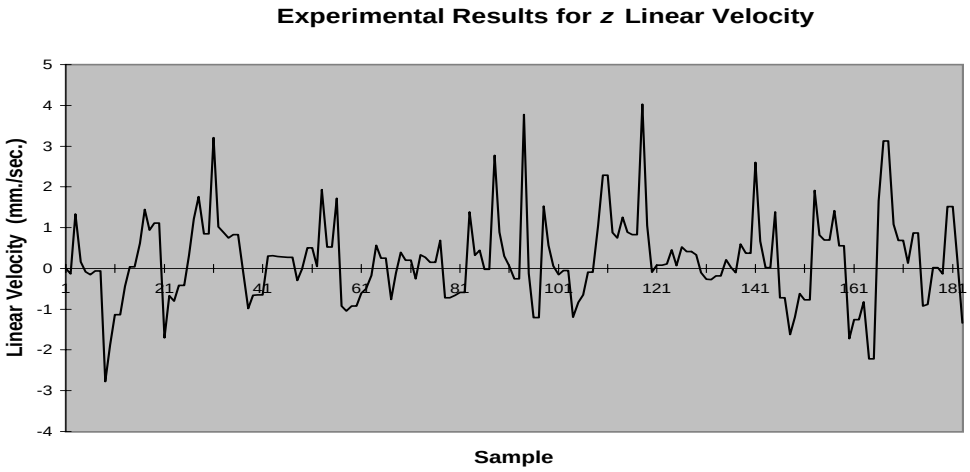

Fig. 5.59. v_z translation velocity pose estimate from experimental testing for constant target motion about z axis. The estimate shows a noisy variation about a mean z velocity value of zero. The true target v_z velocity value is nominally zero.

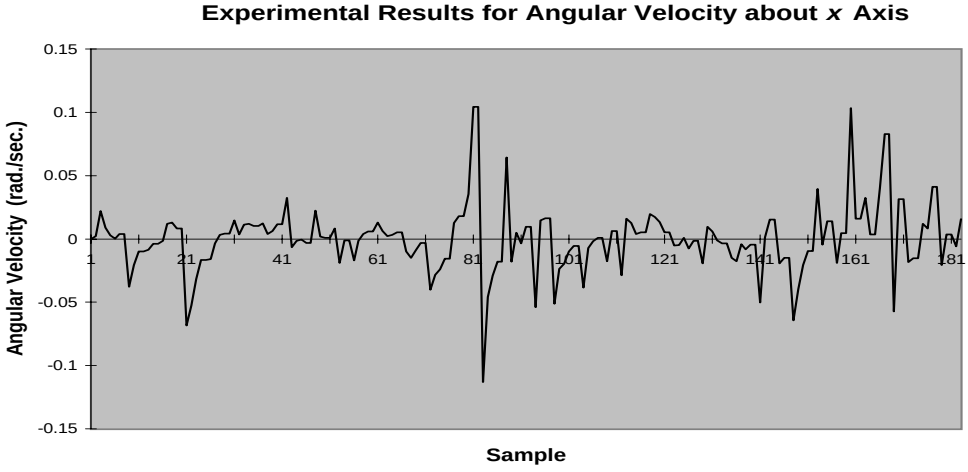

Fig. 5.60. ω_x angular velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy variation about a mean ω_x velocity value of zero. The true target ω_x velocity value is nominally zero.

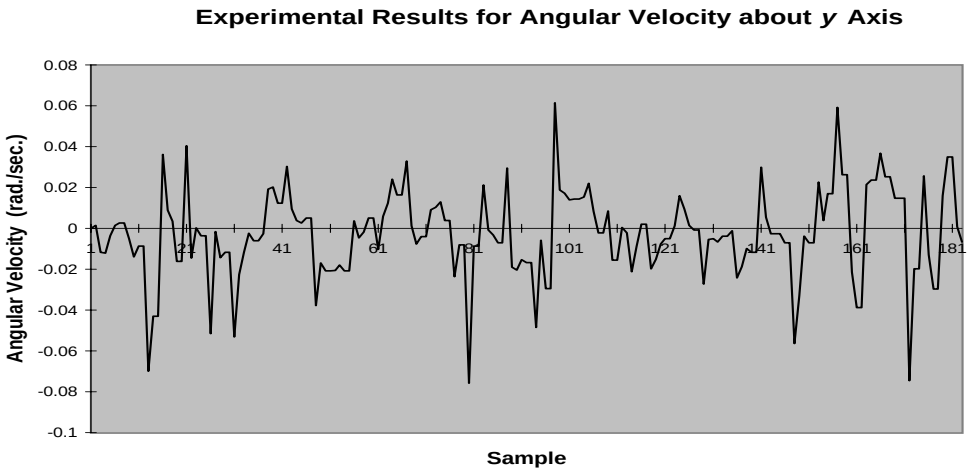

Fig. 5.61. ω_y angular velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy variation about a mean ω_y velocity value of zero. The true target ω_y velocity value is nominally zero.

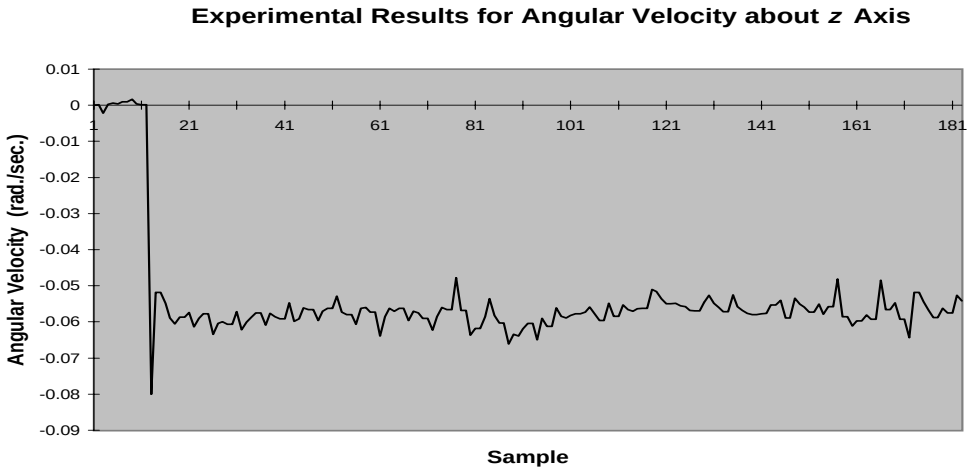

Fig. 5.62. ω_z angular velocity pose estimate from experimental testing for constant target motion about the z axis. The estimate shows a noisy variation about a mean ω_z velocity value of -0.06 rad./sec. The true target ω_z velocity value is nominally constant at -0.06 rad./sec.

q_2 vary little with no trend discernable. Linear velocity estimates similarly illustrate the sinusoidal velocity components in x and y. The z linear velocity estimate, however, shows significant noise present, masking the true velocity component which is near zero. Angular velocity estimates show significant noise in the x and y components although the mean appears to remain near zero. The angular z velocity, however, gives an accurate estimate of the true rotational speed. The 0.06 rad./sec. velocity corresponds to a revolution time of 105 seconds. This time has been verified to within one second. The z angular velocity results also show the fast transient response of the initial start-up from no motion to the constant rotational speed.

5.3 Test Results From Unknown Object Geometry

Experimental results to demonstrate structure and pose estimation are given in this section. A more realistic object than the rectangular target used in Section 5.2 was assembled from sections of PVC pipe. Figure 5.63 is a picture of the pipe object. Four sections of straight pipe were connected using joints to form the object. No dimensional information was known about the pipe prior to the experiment. The CIDTEC camera mounted on the Mitsubishi robot arm and the computer system given in Figure 5.46 were used for these tests.

5.3.1 Structure Estimation

The structure estimation is performed for linear robot arm motion and for rotational motion. For comparison, a reference structure is calculated in terms of the line parameters $\vec{l}$ and $\vec{m}$ given in Chapter 3. This reference method uses a stereo technique to calculate the true dimensions of the pipe assembly. The details involved in calculating the true 3–D line parameters from the set of stereo images obtained with the robot arm are given

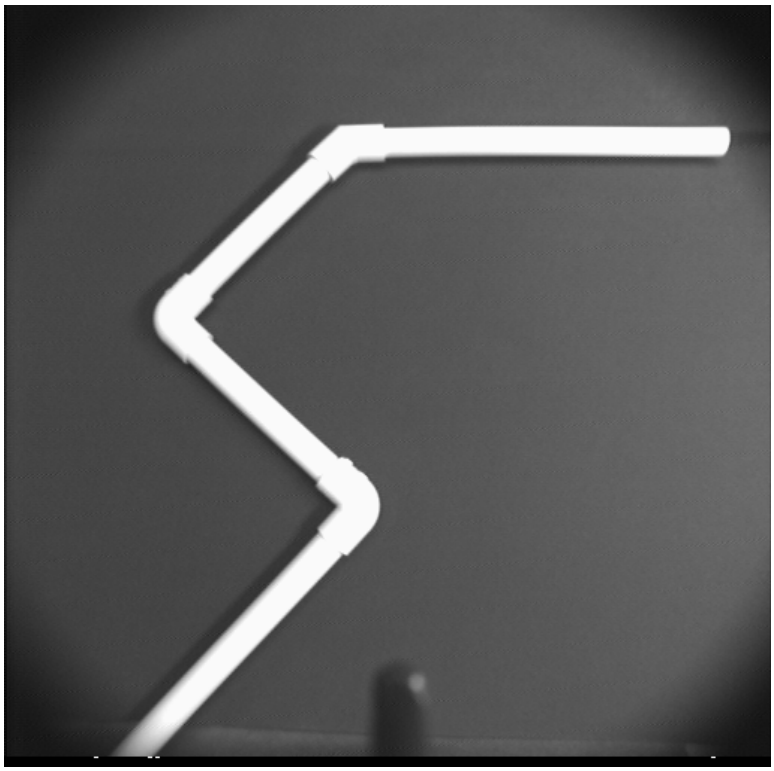

Fig. 5.63. Pipe assembly used in the structure estimation robot arm experiments. The picture shows the four segments of white PVC pipe joined together.

below. During operation, the structure estimates which are referenced to a local reference frame are transformed to the calculated reference frame and then used for comparison.

5.3.1.1 Calculation of Actual Structure for Reference

The procedure used to calculate the actual structure is to move the robot arm over the pipe, stop the motion, and then acquire 100 frames and extract the line parameters of the pipe features for each frame. The arm is then moved 50 mm. in the positive x direction; 100 frames were again taken with the line parameters calculated. The average of the line parameters from each of the left and right positions is then calculated. Each 3–D pipe line feature is computed using the following relations. First, the normalized $\overrightarrow{mn}$ values are calculated from the image line points from each set of left and right points

$$\overrightarrow{mn} = \begin{bmatrix} \frac{\lambda x_{lp}}{\rho\sqrt{\lambda^2+\rho^2}} \\ \frac{\lambda y_{lp}}{\rho\sqrt{\lambda^2+\rho^2}} \\ \frac{\rho}{\sqrt{\lambda^2+\rho^2}} \end{bmatrix} \tag{5.11}$$

where λ is the focal length and $\rho = \sqrt{x_{lp}^2 + y_{lp}^2}$.

The $\overrightarrow{mn}$'s give the normal to the plane containing the object line, the image line, and the optical center. The direction vector $\vec{l}$ is found from the cross product

$$\vec{l} = \overrightarrow{mn}_{left} \times \overrightarrow{mn}_{right}. \tag{5.12}$$

The magnitudes of the $\vec{m}$ left and right vectors are still unknown. These can be expressed in terms of the transformation between the left and right reference frames,

$$k_{left}\overrightarrow{mn}_{left} = k_{right}\overrightarrow{mn}_{right} + (\vec{t} \times \vec{l}), \tag{5.13}$$

where in this case $\vec{t}$ is the translation and is equal to $[50\ 0\ 0]^T$. An assumption is also implicitly made that no rotation occurs, so that the direction vector is the same for both the left and the right sides. Since only x translation is involved, the y and z components of the above equation can be used to solve for the scale factor k_{right}:

$$k_{right} = \frac{mn_{left_z}(\vec{t} \times \vec{l})_y - mn_{left}(\vec{t} \times \vec{l})_z}{mn_{left_y} mn_{right_z} - mn_{right_y} mn_{left_z}}. \tag{5.14}$$

An additional translation of 500 mm. in the positive z direction is then applied to place the reference origin close to the object. The resulting structure data for the pipe is given in Table 5.5 below.

Table 5.5.
Reference pipe line data as calculated by two image views separated by an x translation of 50 mm. The data show the $\vec{l}$ and $\vec{m}$ components of the 3–D lines.

Line	l_x	l_y	l_z	m_x	m_y	m_z
0	-0.512	0.859	0	47.5	28.4	-30.2
1	0.846	0.453	0.281	-6.67	-40.9	85.7
2	-0.543	0.838	0.0482	0.332	-11.1	196.5
3	-0.981	0.194	-0.0171	-4.74	-4.68	218.384

5.3.1.2 Initial State Estimate

The structure is estimated relative to a fixed reference frame set during algorithm initialization that depends on the relative position between the camera and the object. Since the scale for translation and structure is arbitrary, a normalization is applied to ensure that the scale is fixed during state and measurement updates. This fixed frame is

set by acquiring an initial image of the object before estimation begins. The state pose variables are set to the following:

$$\begin{aligned} t_x &= t_y = 0 \\ q_0 &= 1 \\ q_1 &= q_2 = q_3 = 0. \end{aligned} \tag{5.15}$$

This pose setting aligns the object reference frame's x and y axes with the camera reference frame's axes. With no relative rotation, the z-axis translation is calculated from the first image line parameters:

$$t_z = \frac{\lambda}{\rho} \tag{5.16}$$

with $\rho = \sqrt{x_{lp}^2 + y_{lp}^2}$.

Structure line initial values are normalized to the first $\overrightarrow{m0}$ line vector. These vectors have the initial values

$$\begin{aligned} \vec{m} &= \begin{bmatrix} 0 & 0 & \frac{-\rho t_z}{\lambda} \end{bmatrix}^T \\ \vec{l} &= \begin{bmatrix} \frac{-y_{lp}}{\rho} & \frac{x_{lp}}{\rho} & 0 \end{bmatrix}^T. \end{aligned} \tag{5.17}$$

The direction vectors, $\vec{l}$, are assumed to be parallel to the image plane.

5.3.1.3 Transformation of Structure Estimate Frame to Reference Frame

The object reference frame is dependent on the initial relative pose between the camera and the object and varies with each start-up. A transformation is required to compare the estimation results between different tests or to a known reference. For comparison to the reference structure obtained through the stereo technique given in Section 5.3.1.1, the estimated structure must be transformed to the base reference frame. This relationship between the estimated frame and the reference frame is given for each object line by

$$\begin{aligned} \vec{l}_{ref} &= R\vec{l}_{est} \\ \vec{m}_{ref} &= sR\vec{m}_{est} + \vec{t}\times\vec{l}_{ref} \end{aligned} \tag{5.18}$$

with rotation matrix R, translation vector $\vec{t}$, and scale s to be determined.

R for the first equation can be found by application of the method given by Horn for calculating the transformation between two 3–D reference frames [63]. For the second equation, the error squared sum is

$$\sum_{i=0}^{n-1} |e|^2 = \sum_{i=0}^{n-1} |\vec{m}_{ref_i} - sRm_{est_i} - L_{ref_i}\vec{t}|^2. \tag{5.19}$$

This expression is minimized with respect to s and $\vec{t}$ by computing the partial derivatives and setting them equal to zero. Solving for $\vec{t}$ results in

$$\vec{t} = \left[I_3 - \frac{(\sum_{i=0}^{n-1} L_{ref_i}^T L_{ref_i})^{-1}(\sum_{i=0}^{n-1} L_{ref_i}^T R\vec{m}_{est})(\sum_{i=0}^{n-1}(R\vec{m}_{est_i})^T L_{ref_i})}{\sum_{i=0}^{n-1} |R\vec{m}_{est_i}|^2} \right]^{-1}$$

$$\left(\sum_{i=0}^{n-1} L_{ref_i}^T L_{ref_i})^{-1}\right)$$
$$\left((\sum_{i=0}^{n-1} L_{ref_i}^T \vec{m}_{ref_i}) - \frac{(\sum_{i=0}^{n-1} L_{ref_i}^T R\vec{m}_{est})(\sum_{i=0}^{n-1}(R\vec{m}_{est_i})^T \vec{m}_{ref_i})}{\sum_{i=0}^{n-1} |R\vec{m}_{est_i}|^2}\right) \tag{5.20}$$

and for s results in

$$s = \frac{\sum_{i=0}^{n-1}(R\vec{m}_{est_i})^T \vec{m}_{ref_i} - \sum_{i=0}^{n-1}(R\vec{m}_{est_i})^T L_{ref_i}\vec{t}}{\sum_{i=0}^{n-1} |R\vec{m}_{est_i}|^2}. \tag{5.21}$$

These expressions provide the least squared error transformation between the two frames. This transformation is then applied to the estimated structure values to transform them to the baseline reference frame. The reference and estimated values can, as a result, be directly compared.

5.3.2 Response to Step Change in Motion

The structure estimates are obtained by introducing a step change in motion to the robot arm holding the camera. The axes changed were the x axis, z axis, and the roll axis about the z axis. In each case, only a single axis was moved at a time. Figures 5.64 through 5.71 give the results of the structure estimates when the z axis is moved.

These figures show the $\vec{l}$ and $\vec{m}$ components of the object lines. Also shown are the true structure values. The estimates have been transformed to the reference frame using the method in Section 5.3.1.3 above. Generally, good tracking of the structure values is obtained. As seen, the $\vec{l}$ estimates give somewhat more accurate results than the $\vec{m}$ estimates. Results from moving the other axes were similar.

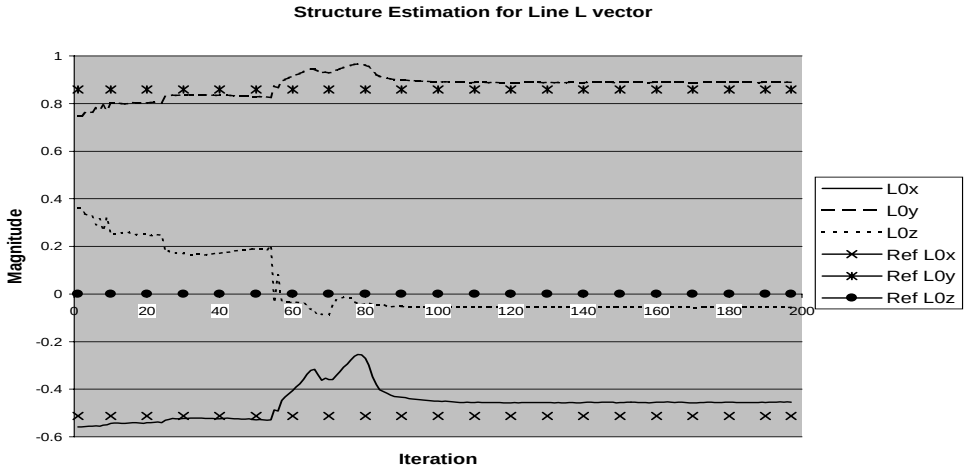

Fig. 5.64. Structure estimate of the three $\vec{l}$ line vector components for line zero with motion along the z axis. The true structure values are also shown for comparison. The x and y components are close to the true values while the z component has a constant offset error.

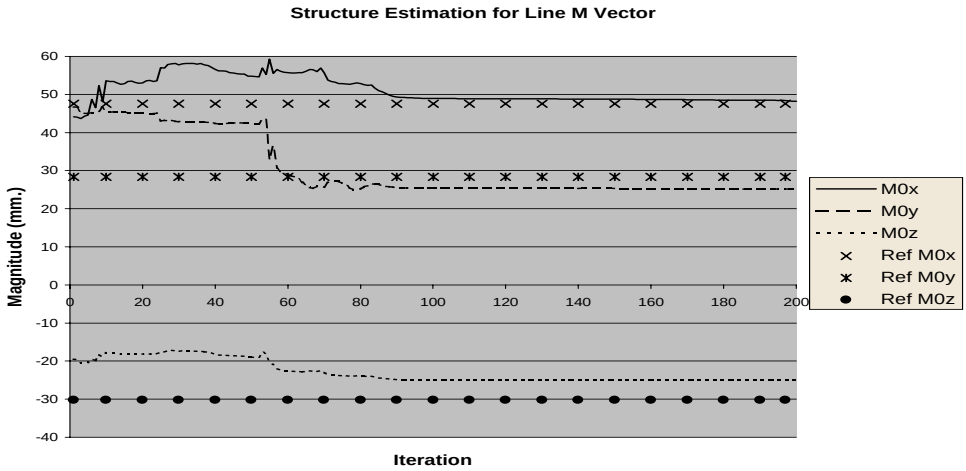

Fig. 5.65. Structure estimate of the three $\vec{m}$ line vector components for line zero with motion along the z axis. The true structure values are also shown for comparison. The y and z components are accurately estimated while the x component has significant error.

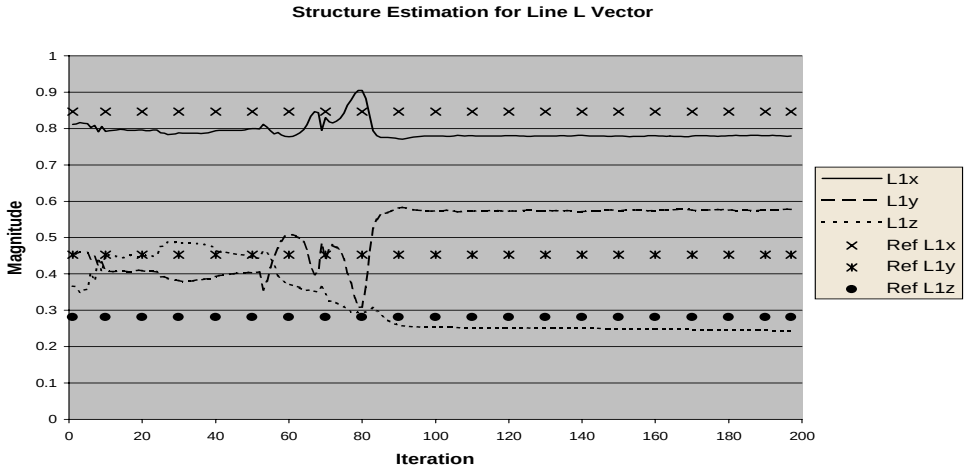

Fig. 5.66. Structure estimate of the three $\vec{l}$ line vector components for line one with motion along the z axis. The true structure values are also shown for comparison. The x and y components have small errors in the estimate while the z component is seen to converge slowly to the true value.

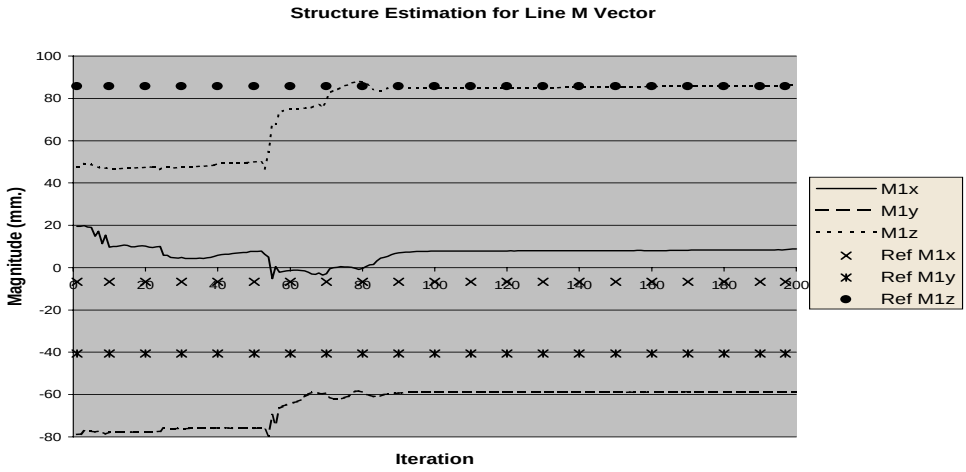

Fig. 5.67. Structure estimate of the three $\vec{m}$ line vector components for line one with motion along the z axis. The true structure values are also shown for comparison. The z component is estimated accurately while the x component is seen to converge to the correct value. Error in y, though initially small, increases with time.

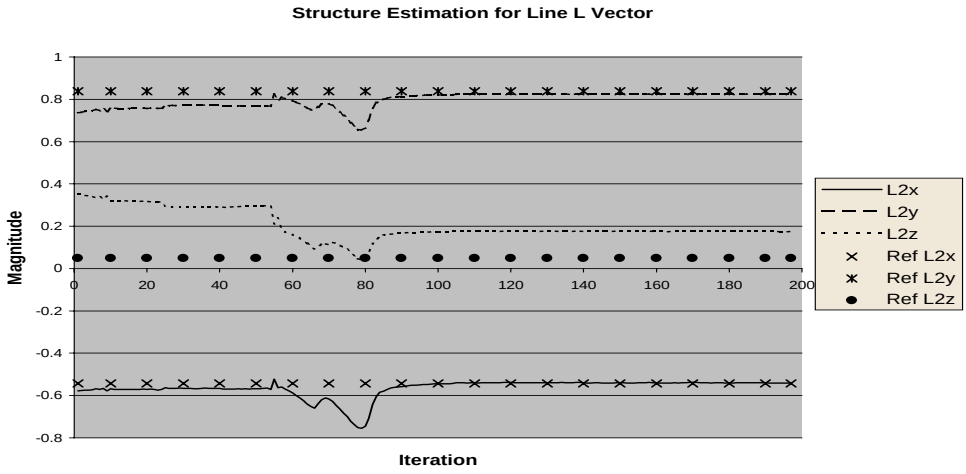

Fig. 5.68. Structure estimate of the three $\vec{l}$ line vector components for line two with motion along the z axis. The true structure values are also shown for comparison. Accurate estimates are obtained for all three components.

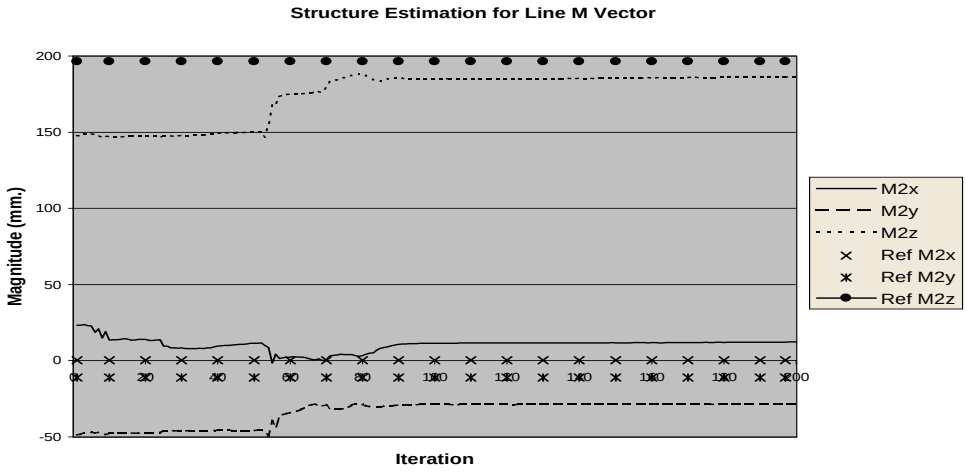

Fig. 5.69. Structure estimate of the three $\vec{m}$ line vector components for line two with motion along the z axis. The true structure values are also shown for comparison. Accurate estimates are obtained for all three components.

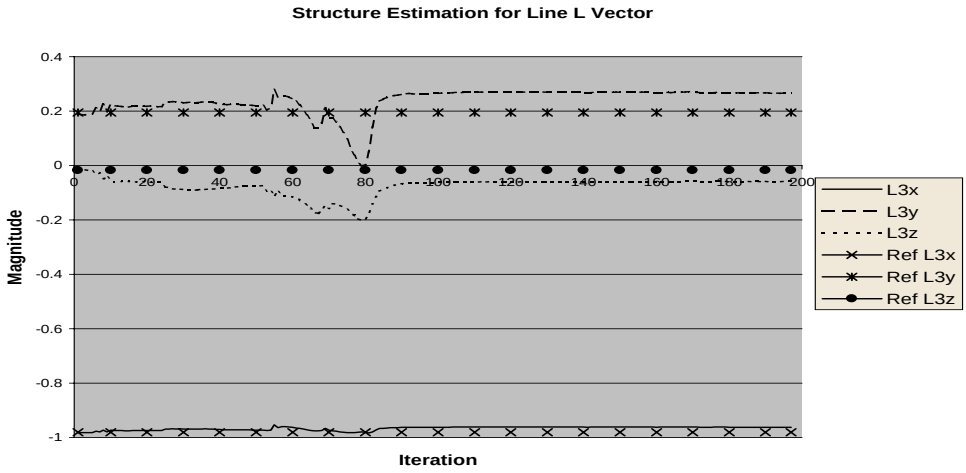

Fig. 5.70. Structure estimate of the three $\vec{l}$ line vector components for line four with motion along the z axis. The true structure values are also shown for comparison. The x and y components are close to the true values while the z component has a constant offset error.

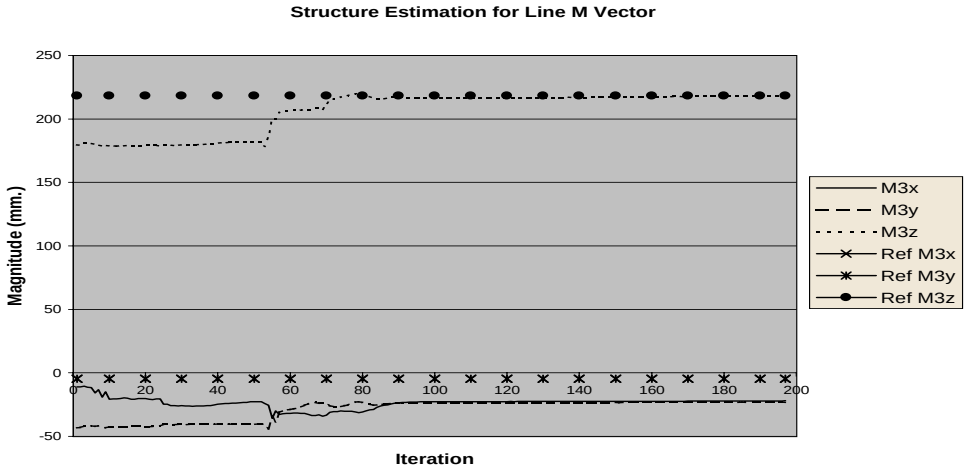

Fig. 5.71. Structure estimate of the three $\vec{m}$ line vector components for line four with motion along the z axis. The true structure values are also shown for comparison. The x and z components have relatively small errors while a constant offset error is present for the y component.

Chapter 6

Summary and Conclusions

6.1 Summary

A new pose and motion estimation method has been developed and described in this dissertation. This method uses an imaging technique with a single low-cost CCD camera along with a reference object or scene to calculate estimates for relative six-degree-of-freedom position and orientation as well as the associated velocity estimates. Intended for real-time use, this method can be applied to robot vision, control, and assembly tasks. For an object with unknown geometry, the method is also able to estimate the scaled 3–D object coordinates. The system model for the method is based on line features, a dual quaternion parameterization for the 3-D transformation, and the IEKF. The IEKF is used as an estimator for the highly nonlinear imaging model. Development of the theory and the overall approach is described in detail.

This approach provides two solutions for the 3–D estimation problem. For a rigid object or environment whose dimensions are unknown, the problem is to find the object structure in terms of the line feature coordinates in the local 3-D reference frame from the motion of the robot. Once the geometric target dimensions are estimated, the basic method may then be applied to provide the pose and motion estimation. The full method to estimate structure as well as pose and motion requires the estimation of many more state variables than when the object dimensions are known. As a result, computational requirements are also increased significantly and real-time application may not be feasible.

A potential use for this method is on a mobile robot that is attempting to navigate through a room with various kinds of obstructions such as pipes, columns, and other equipment. The experimental testing has shown the feasibility of using this method in a realistic application where objects with line features are present. The testing demonstrated the image acquisition, feature extraction, matching and feature correspondence to the model, and then the pose and motion estimation as the robot moves within the scene. Another potential application is in visual servoing whereby a camera-mounted robot arm maintains a fixed position and orientation with respect to a moving object within a scene.

An analysis of the relationship of the measurement noise in the projected 2–D image line parameters to the noise in associated point features has also been presented. Experimental testing to demonstrate and verify the performance of the method from both simulations and actual robot arm tests have been performed and presented as well. The results from these tests have been compared to the performance of a point-based method.

6.2 Conclusions and Contributions

The newly developed solution given here is unique among previous methods of real-time pose and motion estimation. This method is based on line features, 3–D transformations of lines, and extended Kalman filtering. While others have previously used extended Kalman filtering for 3–D image-based pose estimation, this previous work has used point features for the 3–D and imaging models. This new method describes the first known effort that combines line features and extended Kalman filtering to estimate pose and motion as well as 3–D structure. The EKF is particularly appropriate for this application because it permits the estimation of unknown quantities along with the derivatives of these quantities to any desired order in a format suitable for real-time application since a new estimate is produced at each iteration. The standard linear Kalman filter is also the optimal estimator where the measurement and process noise statistics are Gaussian.

Extending the filter to nonlinear applications results in a suboptimal estimator. The use of a suboptimal estimator here is justified, however, since optimal estimation of nonlinear systems is not generally amenable to analysis. No analysis pertaining to optimal estimation for the pose estimation problem was found in the literature. An analysis of the problem was examined during this work and is not believed to be analytically solvable. Good estimation results have been obtained in the past with the IEKF in other applications [99] as well as in pose estimation using point features. When the system model and measurement model satisfy the standard Kalman filter conditions, better estimation will result from the EKF. In Chapter 4, it was shown that the line-point parameters have near-Gaussian density functions when the end points have Gaussian distributions. For image feature points, a Gaussian distribution is both a common assumption and one that corresponds to actual data. This satisfies the standard Kalman filter assumption that the measurements have a Gaussian distribution.

The EKF, while well suited as an estimator for this application, requires setup and tuning to ensure stability over the expected range of values. In particular, the initial error covariance, P_k, should reasonably reflect the expected initial errors in the state variables. The filter can become unstable for error covariance values too high, while values too low will result in poor settling time or divergence from the true value. The process noise, Q_k, is also critical in obtaining good results. If the imaging model were perfect and the relative camera-to-object motion were perfectly uniform, the process noise would be zero. However, in the physical world, perspective projection is not exactly obtained due to lens aberrations and distortions as well as errors in the image sensing by a camera and the acquisition of the data. Neither is motion constant since an object may accelerate or be perturbed by external influences. The process noise must be determined empirically either through simulation or through actual testing. Even in simulations where the assumed model is correct to double precision floating point accuracy, process noise may be required

simply due to the nonlinearity of the model and the errors due to the linearization based on the truncated Taylor series expansion of the defining equations. Incorrect values, here again, may result in an unstable filter.

For a real-time application, computational requirements of an estimation method are highly important. A position control feedback control loop may require updates on the order of tens or even hundreds of times per second. The principal factors directly affecting the computation speed for the EKF used in this method are the number of states and the number of measurements. The number of computations is on the order of the square of these values. Consequently, structure estimation, since it requires many more states, is not as suitable for real-time implementation as estimation of only the pose variables. The number of measurements depends on the number of lines in the scene being used for the estimation. Accuracy improves with an increase in the number of lines but at the expense of increased computation. As the results from this method have shown, good accuracy can be obtained even with a minimal set of four feature lines. Six lines give even higher accuracy with improved stability but can still be considered within the real-time range. For example, the computation time for the EKF estimation method, written in C++ on a 166 Mhz. Pentium PC, was 100 ms. using a six-line object.

The experimental results obtained from simulation and actual robot arm testing have shown that this method estimates 3–D pose parameters more accurately than previous point-based extended Kalman filtering within the central region of a camera view. In particular, the range and range motion estimation has been shown to be measured more accurately with faster settling times than with other methods of real-time estimation. Noise analysis has shown that these results are, in part, due to the noise statistics of the line measurements. For line features around the center of the image, the noise variance is less than the corresponding noise from points used as features. Thus, the value can be estimated more accurately in this region. However, as an object moves a significant distance

from the center, the variance increases, reaching a point where the resulting estimate will be worse than the corresponding point-based estimate. Quantifying this region has not proved successful since no one distance will result in all pose variables being estimated more accurately. The behavior is that some variables will become more accurate while others will not. Tuning by way of the process noise will also affect the point at which one method will show better results.

This work has produced a number of new developments in the area of pose estimation as a result of the analysis and experimental testing. The primary contributions resulting from this work include the application of the EKF to pose, motion, and structure estimation based on line features from a scene or object. The combining of line features with the EKF has not been previously seen. The pose estimation method has been developed and implemented in C++ software and demonstrates the method is more accurate and has faster response when compared to point-based methods over the central region of an image. Parameterization of the line transformations in terms of dual quaternions as an aid in understanding and developing the Kalman filter measurement model is also a new contribution. While others have used dual quaternions to perform 3–D transformations of lines, this approach applies dual quaternions within the Kalman filter measurement model to transform from the object reference frame to the camera reference frame. Also new is a two-parameter 2–D line representation that is continuously differentiable in the image plane with respect to the image plane x and y axes. The partial derivatives of the parameters with respect to each state variable are also continuous. Analysis of the noise statistics of the 2–D image line-point parameters in relation to point feature noise statistics was also performed. The pdf of the line point as a function of the end point density functions was derived, requiring numerical integration for solution. As part of this analysis, the variance and covariance of the line point values were also derived in terms of the end points. Since

the exact solution also requires numerical integration, an approximate solution was developed suitable for real-time implementation. These estimated covariances are applied to the measurement covariance matrix of the EKF at each iteration so that during relative motion, the measurement covariances accurately reflect the statistics of the measurement data processed by the filter. The result is improved state estimates with less error. Another significant contribution is the method developed to compare structure estimates. This method calculates the least squared error transformation of 3–D lines from a local reference frame to a base reference frame. Software for the pose estimation method has been developed and fully implemented in C++. It has been integrated on a computer system with a robot arm and camera and successfully demonstrated.

6.3 Future Work

While the basic method has been developed and successfully tested, much additional work remains in terms of analysis and extension of the concepts. One application of the method is for visual servoing of a robot arm. This requires implementation of a feedback control loop for moving the arm in relation to scene motion. The system implemented and demonstrated for this work had iteration times on the order of one second and a robot command interface that permitted only point-to-point movements. While the basic operation could be observed, a better implementation on fast hardware would be a control loop executing ten or more times per second and directly controlling the joint angles of the arm. Another area is to extend the physical model to estimate higher order derivatives (acceleration, etc.) and provide an evaluation of performance in comparison to this method. Also of interest would be an analysis of the iteration count for the IEKF. During experimentation, it was found that iteration counts of three or more sometimes gave results with less error than an iteration count of one while under different conditions the reverse was true. Usually a higher iteration count improved estimates

from the point method while the line method gave the best estimates with an iteration count of one. Other research areas include the analysis of the EKF in terms of stability and optimum performance. Analyzing the effects of and aids to assist in setting the filter parameters for near optimum performance would be of great benefit. Another research area would be to analyze the computational requirements of this method in comparison to the computational requirements of the point method.

REFERENCES

REFERENCES

[1] J. S.-C. Yuan, "A general photogrammetric method for determining object position and orientation," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 2, pp. 129–142, April 1989.

[2] R. M. Haralick and H. Joo, "2d-3d pose estimation," in *Proceedings of the International Conference on Pattern Recognition*, 1988, pp. 385–391.

[3] R. J. Holt and A. N. Netravali, "Camera calibration problem: Some new results," *CVGIP: Image Understanding*, vol. 54, no. 3, pp. 368–383, November 1991.

[4] M. A. Abidi and T. Chandra, "Pose estimation for camera calibration and landmark tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation*, May 1990, pp. 420–426.

[5] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 4, pp. 323–344, August 1987.

[6] V. Torre, A. Verri, and A. Fiumicelli, "Stereo accuracy for robotics," in *Robotics Research, The Third International Symposium*, G. Giralt O. D. Faugeras, Ed., pp. 5–9. MIT Press, 1986.

[7] D. F. DeMenthon and L. S. Davis, "Model-based object pose in 25 lines of code," in *Computer Vision– ECCV '92*, G. Sandini, Ed., pp. 335–343. Springer-Verlag, 1992.

[8] M. A. Abidi and T. Chandra, "A new efficient and direct solution for pose estimation using quadrangular targets: Algorithm and evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 5, pp. 534–538, May 1995.

[9] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, June 1981.

[10] L. G. Trabasso and C. Zielinski, "Semi-automatic calibration procedure for the vision-robot interface applied to scale model decoration," *Robotica*, vol. 10, pp. 303–308, 1992.

[11] M. W. Walker, L. Shao, and R. A. Volz, "Estimating 3-d location parameters using dual number quaternions," *CVGIP: Image Understanding*, vol. 54, no. 3, pp. 358–367, November 1991.

[12] T. Q. Phong, R. Horaud, A. Yassine, and D. T. Pham, "Optimal estimation of object pose from a single perspective view," in *International Conference on Computer Vision*, February 1993, pp. 534–539.

[13] T. Phong, R. Horaud, A. Yassine, and D. Pham, "Object pose from 2-d to 3-d point and line correspondences," *International Journal of Computer Vision*, vol. 15, pp. 225–243, 1995.

[14] M. A. Abidi and R. C. Gonzalez, "The use of multisensor data for robotic applications," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 2, pp. 159–177, April 1990.

[15] N. Chen, J. R. Birk, and R. B. Kelley, "Estimating workpiece pose using the feature points method," *IEEE Transactions on Automatic Control*, vol. AC-25, no. 6, pp. 1027–1041, December 1980.

[16] R. M. Haralick, "Pose estimation from corresponding point data," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 6, pp. 1426–1446, November 1989.

[17] C. J. Ho and N. H. McClamroch, "Automatic spacecraft docking using computer vision-based guidance and control techniques," *Journal of Guidance, Control, and Dynamics*, vol. 16, no. 2, pp. 281–288, March 1993.

[18] R. Horaud, B. Conio, and O. Leboulleux, "An analytic solution for the perspective 4-point problem," *Computer Vision, Graphics, and Image Processing*, vol. 47, pp. 33–44, 1989.

[19] R. T. Howard and M. L. Book, "Video guidance sensor for autonomous capture," in *NASA Automated Rendezvous and Capture Review*, November 1991.

[20] K. Hung, C. Shyi, J. Lee, and T. Lee, "Robot location determination in a complex environment by multiple marks," *Pattern Recognition*, vol. 21, no. 6, pp. 567–580, 1988.

[21] Y. Hung, P. Yeh, and D. Harwood, "Passive ranging to known planar point sets," Tech. Rep. CAR-TR-65, Center for Automation Research, University of Maryland, June 1984.

[22] S. Kamata, R. O. Eason, M. Tsuji, and E. Kawaguchi, "A camera calibration using 4 point-targets," in *Proceedings 11th IAPR International Conference on Pattern Recognition*, August 1992, vol. 1, pp. 550–553.

[23] D. H. Kite and M. Magee, "Reasoning about camera positioning relative to rectangular patterns," in *Proceedings SPIE Sensor Fusion: Spatial Reasoning and Scene Interpretation*, November 1988, vol. 1003, pp. 222–233.

[24] R. Krishnan, H. J. Sommer III, and P. D. Spidaliere, "Monocular pose of a rigid body using point landmarks," *Computer Vision, Graphics, and Image Processing*, vol. 55, no. 3, pp. 307–316, May 1992.

[25] F. E. Veldpaus, "An analytic solution for the perspective 4-point problem," *CVGIP*, vol. 47, pp. 33–44, 1989.

[26] K. Mandel and N. A. Duffie, "On-line compensation of mobile robot docking errors," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 6, pp. 591–598, December 1987.

[27] Y. Liu, T. S. Huang, and O. D. Faugeras, "Determination of camera location from 2-d to 3-d line and point correspondences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 28–37, January 1990.

[28] J. C. Tietz and L. M. Germann, "Autonomous rendezvous and docking," in *Proceedings of the 1982 American Control Conference*, 1982, vol. 2, pp. 460–465.

[29] W. B. Jatko, J. S. Goddard, R. K. Ferrell, S. S. Gleason, J. S. Hicks, and V. K. Varma, "Crusader automated docking system phase iii report," Tech. Rep. ORNL/TM-13177, Oak Ridge National Laboratory, March 1996.

[30] D. G. Lowe, *Perceptual Organization and Visual Recognition*, Kluwer, Boston, MA, 1985.

[31] J. S. Goddard, W. B. Jatko, R. K. Ferrell, and S. S. Gleason, "Robust pose determination for autonomous docking," in *ANS Sixth Topical Meeting on Robotics and Remote Systems*, February 1995, pp. 767–774.

[32] W. J. Wolfe, G. K. White, and L. J. Pinson, "A multisensor robotic locating system and the camera calibration problem," in *SPIE Intelligent Robots and Computer Vision*, September 1985, vol. 579, pp. 420–431.

[33] H. F. L. Pinkney, C. I. Perratt, and S. G. MacLean, "Canex-2 space vision system experiments for shuttle flight sts-54," in *SPIE Close-Range Photogrammetry Meets Machine Vision*, 1990, vol. 1395, pp. 374–381.

[34] H. F. L. Pinkney and C. I. Perratt, "A flexible machine vision guidance system for 3-dimensional control tasks," in *Proceedings of the International Society for Photogrammetry and Remote Sensing, Commission V Symposium*, June 1986, pp. 401–411.

[35] J. Ponce, A. Hoogs, and D. J. Kreigman, "On using cad models to compute the pose of curved 3d objects," *CVGIP: Image Understanding*, vol. 55, no. 2, pp. 184–197, March 1992.

[36] S. Chen and W. Tsai, "Determination of robot locations by common object shapes," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 1, pp. 149–156, February 1991.

[37] R. Safaee-Rad, I. Tchoukanov, B. Benhabib, and K. C. Smith, "Accurate parameter estimation of quadratic curves from grey-level images," *CVGIP: Image Understanding*, vol. 54, no. 2, pp. 259–274, September 1991.

[38] Y. Han, W. E. Snyder, and G. L. Bilbro, "Pose determination using tree annealing," in *Proceedings IEEE International Conference on Robotics and Automation*, May 1990, pp. 427–432.

[39] J. Ohya, L. Davis, and D. DeMenthon, "Recognizing 3-d objects from range images and finding their six pose parameters," Tech. Rep. CAR-TR-538, Center for Automation Research, University of Maryland, February 1991.

[40] B. Wirtz and C. Maggioni, "3-d pose estimation by an improved kohonen-net," in *Visual Form: Analysis and Recognition*, C. Arcelli, Ed., pp. 593–602. Plenum Press, New York, 1992.

[41] S. Ma, S. Si, and Z. Chen, "Quadric curve based stereo," in *Proceedings 11th International Conference on Pattern Recognition*, The Hague, The Netherlands, August 1992, vol. 1, pp. 1–4.

[42] H. H. Chen, "Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6, pp. 530–541, June 1991.

[43] R. Consales, A. D. Bimbo, and P. Nesi, "On the perspective projection of 3-d planar-faced junctions," in *Proceedings 11th International Conference on Pattern Recognition*, The Hague, The Netherlands, August 1992, vol. 1, pp. 616–619.

[44] M. Dhome, M. Richetin, J. LaPreste, and G. Rives, "Determination of the attitude of 3-d objects from a single perspective view," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 12, pp. 1265–1278, December 1989.

[45] T. Ellis, A. Abbood, and B. Brillaut, "Ellipse detection and matching with uncertainty," *Image and Vision Computing*, vol. 10, no. 5, pp. 271–276, June 1992.

[46] N. J. Foster and A. C. Sanderson, "Determining object orientation using ellipse fitting," in *SPIE Intelligent Robots and Computer Vision*, 1984, vol. 521, pp. 33–43.

[47] R. Safaee-Rad, I. Tchoukanov, B. Benhabib, and K. C. Smith, "3d pose estimation from a quadratic-curved feature in two perspective views," in *Proceedings 11th IAPR International Conference on Pattern Recognition*, The Hague, The Netherlands, August 1992, vol. 1, pp. 341–344.

[48] R. Safaee-Rad, I. Tchoukanov, B. Benhabib, and K. C. Smith, "Accurate parameter estimation of quadratic curves from grey-level images," *CVGIP: Image Understanding*, vol. 54, no. 2, pp. 259–274, September 1991.

[49] R. Safaee-Rad, I. Tchoukanov, K. C. Smith, and B. Benhabib, "Three-dimensional location estimation of circular features for machine vision," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 624–640, October 1992.

[50] M. Ferri, F. Mangili, and G. Viano, "Projective pose estimation of linear and quadratic primitives in monocular computer vision," *CVGIP: Image Understanding*, vol. 58, no. 1, pp. 66–84, July 1993.

[51] R. Glachet, M. Dhome, and J. Lapreste, "Finding the perspective projection of an axis of revolution," *Pattern Recognition Letters*, vol. 12, pp. 693–700, November 1991.

[52] R. R. Goldberg and D. G. Lowe, "Hessian methods for verification of 3d model parameters from 2d image data," in *Proceedings of SPIE: Sensor Fusion: Spatial Reasoning and Scene Interpretation Conference*, 1989, vol. 1003, pp. 63–67.

[53] M. Han and S. Rhee, "Camera calibration for three-dimensional measurement," *Pattern Recognition*, vol. 25, no. 2, pp. 155–164, 1992.

[54] R. M. Haralick and Y. H. Chu, "Solving camera parameters from the perspective projection of a parameterized curve," *Pattern Recognition*, vol. 17, no. 6, pp. 637–645, 1984.

[55] R. M. Haralick, "Determining camera parameters from the perspective projection of a rectangle," *Pattern Recognition*, vol. 22, no. 3, pp. 225–230, 1989.

[56] M. R. Kabuka and A. E. Arenas, "Position verification of a mobile robot using standard pattern," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 6, pp. 505–516, December 1987.

[57] B. Hussain and M. R. Kabuka, "Real-time system for accurate three-dimensional position determination and verification," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 1, pp. 31–43, February 1990.

[58] D. G. Lowe, "Three-dimensional object recognition from single two-dimensional images," *Artificial Intelligence*, vol. 31, pp. 355–393, 1987.

[59] J. Neira, L. Montano, and J. D. Tardos, "Constraint-based object recognition in multisensor systems," in *Proceedings IEEE International Conference on Robotics and Automation*, May 1993, vol. 3, pp. 135–142.

[60] D. Kriegman, B. Vijayakumar, and J. Ponce, "Constraints for recognizing and locating curved 3d objects from monocular image features," in *Computer Vision–ECCV '92*, G. Sandini, Ed., pp. 829–833. Springer-Verlag, 1992.

[61] R. Kumar, "Determination of camera location and orientation," in *Proceedings: Image Understanding Workshop*, May 1989, pp. 870–879.

[62] R. Kumar and A. R. Hanson, "Robust methods for estimating pose and a sensitivity analysis," *CVGIP: Image Understanding*, vol. 60, no. 3, pp. 313–342, November 1994.

[63] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America*, vol. 4, no. 4, pp. 629–642, April 1987.

[64] S. Linnainmaa, D. Harwood, and L. S. Davis, "Pose determination of a three-dimensional object using triangle pairs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 5, pp. 634–647, September 1988.

[65] L. P. Ray, "Estimation of modeled object pose from monocular images," in *Proceedings IEEE International Conference on Robotics and Automation*, May 1990, pp. 408–413.

[66] Y. Roth, A. S. Wu, R. H. Arpaci, T. Weymouth, and R. Jain, "Model-driven pose correction," in *Proceedings IEEE International Conference on Robotics and Automation*, May 1992, pp. 2625–2630.

[67] D. D. Sheu and A. H. Bond, "A generalized method for 3d object location from single 2d images," *Pattern Recognition*, vol. 25, no. 8, pp. 771–786, 1992.

[68] M. A. Abidi, "A regularized multi-dimensional data fusion technique," in *Proceedings IEEE International Conference on Robotics and Automation*, April 1991, pp. 2738–2744.

[69] N. Ayache and O. Faugeras, "Building, registrating, and fusing noisy visual maps," *International Journal of Robotics Research*, vol. 7, no. 6, pp. 45–65, December 1988.

[70] W. M. Wells III, "Posterior marginal pose estimation," in *Proceedings DARPA Image Understanding Workshop*, January 1992, pp. 745–751.

[71] A. Singh, "Robust computation of image-motion and scene-depth," in *Proceedings IEEE International Conference on Robotics and Automation*, April 1991, pp. 2370–2737.

[72] L. Matthies, T. Kanade, and R. Szeliski, "Kalman filter-based algorithms for estimating depth from image sequences," *International Journal of Computer Vision*, vol. 3, no. 8, pp. 209–236, 1989.

[73] J. Weng, P. Cohen, and M. Herniou, "Camera calibration with distortion models and accuracy evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 10, pp. 965–980, October 1992.

[74] A. Azarbayejani and A. P. Pentland, "Recursive estimation of motion, structure, and focal length," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 6, pp. 562–575, June 1995.

[75] 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*, vol. 26, no. 4, pp. 639–656, July 1990.

[76] S. Lee and Y. Kay, "A kalman filter approach for accurate 3-d motion estimation from a sequence of stereo images," *CVGIP: Image Understanding*, vol. 54, no. 2, pp. 244–258, September 1991.

[77] S. Lee and Y. Kay, "An accurate estimation of 3-d position and orientation of a moving object for robot stereo vision: Kalman filter approach," in *Proceedings IEEE International Conference on Robotics and Automation*, May 1990, pp. 414–419.

[78] D. B. Westmore and W. J. Wilson, "Direct dynamic control of a robot using an end-point mounted camera and kalman filter position estimation," in *Proceedings IEEE International Conference on Robotics and Automation*, April 1991, pp. 2376–2384.

[79] J. Wang and W. J. Wilson, "3d relative position and orientation estimation using kalman filter for robot control," in *Proceedings IEEE International Conference on Robotics and Automation*, May 1992, pp. 2638–2645.

[80] B. K. P. Horn, *Robot Vision*, MIT Press, Cambridge, MA, 1986.

[81] J. B. Burl, "A reduced order extended kalman filter for sequential images containing a moving object," *IEEE Transactions on Image Processing*, vol. 2, no. 3, pp. 285–295, July 1993.

[82] P. K. Allen, A. Timcenko, B. Yoshimi, and P. Michelman, "Automated tracking and grasping of a moving object with a robotic hand-eye system," Tech. Rep. CUCS-034-91, Department of Computer Science, Columbia University, November 1991.

[83] N. P. Papanikolopoulos, P. K. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 1, pp. 14–35, February 1993.

[84] W. J. Wilson, C. C. W. Hulls, and G. S. Bell, "Relative end-effector control using cartesian position based visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 684–696, October 1996.

[85] J. C. K. Chou, "Quaternion kinematic and dynamic differential equations," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 1, pp. 53–64, February 1992.

[86] W. K. Clifford, "Preliminary sketch of bi-quaternions," *Proceedings of the London Mathematical Society*, vol. 4, pp. 381–395, 1873.

[87] E. Study, *Geometrie der Dynamen*, Leipzig, 1903.

[88] R. C. Gonzalez and P. Wintz, *Digital Image Processing*, Addison-Wesley, Reading, MA, 1977.

[89] K. Daniilidis and E. Bayro-Corrochano, "The dual quaternion approach to hand-eye calibration," in *Proceedings IEEE International Conference on Pattern Recognition*, 1996, p. A7E.6.

[90] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering (ASME)*, vol. 82D, pp. 35–45, March 1960.

[91] A. Gelb, Ed., *Applied Optimal Estimation*, The MIT Press, Cambridge, Massachusetts, 1974.

[92] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*, Academic Press Inc., Orlando, Florida, 1988.

[93] E. Kreyszig, *Advanced Engineering Mathematics*, John Wiley and Sons, New York, 1983.

[94] W. L. Brogan, *Modern Control Theory*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[95] R. Talluri and J. K. Aggarwal, "Mobile robot self-location using model-image feature correspondence," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 1, pp. 63–77, February 1996.

[96] P. Z. Peebles Jr., *Probability, Random Variables, and Random Signal Principles*, McGraw-Hill, New York, NY, 1980.

[97] J. B. Thomas, *An Introduction to Applied Probability and Random Processes*, John Wiley and Sons, New York, NY, 1971.

[98] C. C. W. Hulls, *Dynamic Real-Time Multisensor Fusion Using an Object Model Reference Approach*, Ph.D. thesis, University of Waterloo, Waterloo, Ontario, 1996.

[99] R. G. Brown and P. Y. C. Hwang, *Introduction to random signals and applied Kalman filtering*, John Wiley and Sons, New York, second edition, 1986.

VITA

James Samuel Goddard, Jr., was born in Sparta, Tennessee, on November 13, 1949. After attending public schools in Tennessee, he graduated from White County High School, Sparta, Tennessee, in June 1967. He received a Bachelor's degree in Electrical Engineering from Georgia Institute of Technology, Atlanta, Georgia, in June 1971. After graduation, he held an engineering position with the National Aeronautics and Space Administration, Goddard Space Flight Center, Greenbelt, Maryland, from 1971 to 1976. During this period, he also attended graduate school at the University of Maryland, College Park, Maryland, receiving a Master of Science Degree with a major in Electrical Engineering in December 1974. Since 1976, he has held several engineering positions at the Department of Energy facilities in Oak Ridge, Tennessee. For the past seven years, he has been a development engineer in the Image Science and Machine Vision group at the Oak Ridge National Laboratory.