

VISION-BASED MEASUREMENTS FOR DYNAMIC SYSTEMS
AND CONTROL

by

Lili Ma

A dissertation submitted in partial fulfillment
of the requirements for the degree

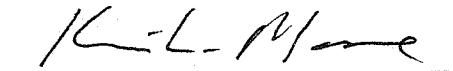
of

Doctor of Philosophy

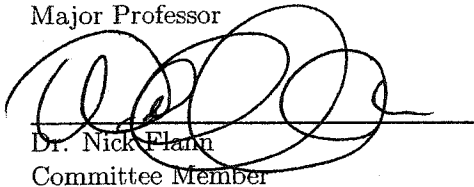
in

Electrical Engineering

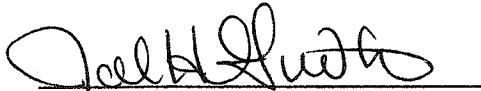
Approved:



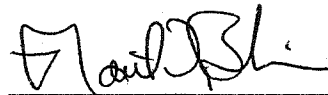
Dr. Kevin L. Moore
Major Professor



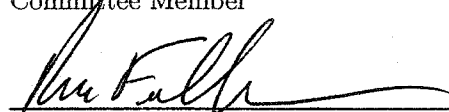
Dr. Nick Flann
Committee Member



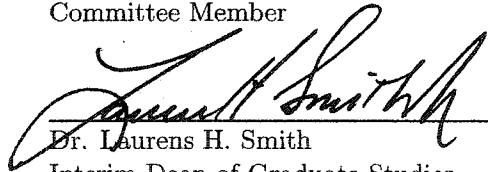
Dr. Jacob Gunther
Committee Member



Dr. Matthew Berkemeier
Committee Member



Dr. Rees Fullmer
Committee Member



Dr. Laurens H. Smith
Interim Dean of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2004

UMI Number: 3151805

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI[®]

UMI Microform 3151805

Copyright 2005 by ProQuest Information and Learning Company.

All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

Copyright © Lili Ma 2004

All Rights Reserved

Abstract

Vision-Based Measurements for Dynamic Systems
and Control

by

Lili Ma, Doctor of Philosophy

Utah State University, 2004

Major Professor: Dr. Kevin L. Moore
Department: Electrical and Computer Engineering

This dissertation presents several aspects of the use of vision for dynamic systems and control. Vision-based feedback systems have been a central problem in computer vision and control for over two decades. However, it is only in recent years that the variations of images over time have begun to be used for control. Motivated from a visual servoing task of an omnidirectional vehicle, the problem of iterative visual servoing with an uncalibrated camera is studied. Then, lens distortion modeling and correction is addressed, where a series of experimental results are given that can serve as a general guidance for evaluation of the lens distortion correction alone. Next, recent work on perspective dynamic systems (PDS), which provides a general framework to discuss vision problems, such as motion and depth estimation, using concepts and methods from controls is considered. Focused on the estimation problem of a PDS, a linear approximation-based nonlinear observer is proposed. The final section of work introduces the idea of iterative learning control of a PDS system and presents preliminary results on this problem.

(219 pages)

*This work is dedicated to
my parents, Qiang Ma and Yongqin Li,
and my brother, Weimin Ma*

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor, Dr. Kevin L. Moore, for all his help and support during my years here. I thank Dr. Moore for his instruction and direction, for his great insight and ideas, for providing a comfortable and fertile environment to work in - USU's Center for Self-Organizing and Intelligent Systems (CSOIS), and mostly for the freedom he gave me to explore. I also owe Dr. YangQuan Chen considerable thanks for his early and patient guidance, for sharing his research ideas, for the timely and detailed discussions, and for his digging out of my inner drives. I thank both Dr. Moore and Dr. Chen for their encouragement at times of my experiencing frustrations and for being more than thesis advisors, but invaluable mentors.

I would like to thank Dr. Nick Flann and Dr. Matthew Berkemeier for their early guidance and supervision in my research works. I thank the other members of my committee, Dr. Rees Fullmer and Dr. Jacob Gunther, for their time and thoughtful recommendations.

Many others have contributed to this work and deserve my thanks: the current and past CSOIS team members, Vikas Bahl, Hitesh Shah, and Zhen Song for working on the control, path planning, and license plate recognition systems on the CSOIS robots; Dr. HuiFang Dou, Mohua Ghosh, and LingHong Zhu for working on the gimbal system; Shayne Rich for installing and maintaining the ODIS camera; Monte Frandsen, Morgan Davidson, Martin Jason, and Gordon Olsen for building the CSOIS omnidirectional robots. The robots and the gimbal system serve as the experimental platforms for the algorithms presented in this dissertation.

Thanks also go to Lynette Gittins for being a wonderful, nice, and organized center secretary, who relieves us of efforts such as payment issues and travel arrangements.

Finally, I would like to thank my parents and brother for their unwavering encouragement, care, and support at all critical moments of my life path.

Lili Ma

Contents

	Page
Abstract	iii
Acknowledgments	v
List of Tables	ix
List of Figures	x
Notations	xiv
Acronyms	xv
1 Introduction	1
1.1 Research Problems	1
1.2 Current State of the Art	3
1.2.1 Iterative Visual Servoing	3
1.2.2 Camera Calibration	4
1.2.3 PDS Theory	5
1.2.4 Perspective ILC	6
1.3 Contributions of This Dissertation	6
1.4 Experimental Platforms Introduction	8
1.5 Reading the Dissertation	8
2 Iterative Visual Servoing	11
2.1 ODIS Localization via Wireless Visual Servoing	13
2.1.1 Basic Idea	13
2.1.2 Hardware Architecture	13
2.1.3 Software Architecture	15
2.1.4 Yellow Line Detection Function	15
2.2 Control Scheme and Stability Analysis	16
2.2.1 Perspective Map	17
2.2.2 Proportional Controller	18
2.2.3 Vehicle Kinematic Equations	20
2.2.4 Final Model and Stability Analysis	20
2.3 Extended Discrete-Time Proportional Controller	22
2.4 Experimental Results	24
2.5 Discussion and Conclusions	26

3	Camera Lens Distortion Correction	27
3.1	Background Introduction	27
3.1.1	Aspects That Real Cameras Deviate from Pinhole Model	27
3.1.2	Camera Parameters and Camera Calibration	29
3.1.3	Radial Distortion	32
3.2	Feature Extraction	35
3.3	Analytical Polynomial Radial Distortion Model	40
3.4	Rational Radial Distortion Models	43
3.5	Piecewise Radial Distortion Model	45
3.6	Experimental Results	46
3.6.1	Initial Model Comparison	47
3.6.2	Distortion Model Selection	48
3.6.3	Evaluation of Distortion Only	52
3.6.4	Regarding Piecewise Model	55
3.6.5	Discussions	56
3.7	Blind Detection of Camera Lens Geometric Distortions	58
3.7.1	Frequency Domain Blind Lens Detection	58
3.7.2	Experimental Results	60
3.8	Summary	68
4	Perspective Dynamic Systems and Observer Design	69
4.1	General Notation and Motion Models	71
4.2	3-D Motion Estimation Techniques	72
4.2.1	Nonlinear Optimization Formulation	73
4.2.2	Linear LS/TLS Algorithms	75
4.2.3	Using Epipolar Constraints	77
4.2.4	Lie Group Formalism	80
4.2.5	Neural Network-Based Estimation	82
4.2.6	Extended Kalman Filter Approach	82
4.3	Perspective Dynamic System Approach	84
4.3.1	Background Theory	87
4.3.2	Applicable Nonlinear Observers for PDS	91
4.3.3	Range Identification with Known Motion Parameters	95
4.4	Comparative Study of Existing Perspective Observers	95
4.4.1	Perspective Nonlinear Observers	96
4.4.2	Simulation Results and Discussions	100
4.5	Range Identification for Perspective Dynamic Systems Using Linear Approximation	104
4.5.1	Linear Approximation Technique	105
4.5.2	Observer Design for LTV Systems	107
4.5.3	LTV Observer for Each Approximation Subsystem	107
4.5.4	Discussions	111
4.5.5	Simulation Results	113
4.5.6	Concluding Remarks	119
4.6	Range Identification for Perspective Dynamic System with Single Homogeneous Observation	120

4.6.1	Nonlinear Observers for PDS with Single Homogeneous Observation	120
4.6.2	Simulation Results	123
4.6.3	Concluding Remarks	126
4.7	Perspective Systems with More General Projection Surfaces	126
4.7.1	PDS with General Planar Imaging Surface	129
4.7.2	PDS with Ball-Shape Imaging Surfaces	132
4.7.3	PDS with Paraboloid Imaging Surface	135
4.7.4	Simulation Results	137
4.7.5	Concluding Remarks	137
4.8	Estimation with Unique Solution	138
4.9	Summary	142
5	Iterative Learning Control for PDS	143
5.1	Introduction of ILC	143
5.2	ILC Control of PDS	144
5.2.1	Problem Formulation	145
5.2.2	Preliminary Study	145
5.2.3	Proof	149
5.3	Experimental Setup	150
5.4	Summary	151
6	Conclusions	152
6.1	Contributions	152
6.2	Future Directions	153
6.2.1	Perspective Dynamic Systems	153
6.2.2	Perspective ILC	154
	Appendices	172
	Appendix A Non-Vision-Based Sensing and Perception	173
A.1	Template Matching	173
A.1.1	Robot Platform Introduction	173
A.1.2	Object Manager	175
A.2	Range Sensors for Obstacle Avoidance	180
A.2.1	HIMM Map Building	182
A.2.2	Experimental Results	187
A.2.3	Decision Making Strategy	190
A.2.4	Parameters Thresholds via Testing	192
A.2.5	Discussion and Conclusions	195
	Appendix B Derivations	196
B.1	Derivations Supporting 3-D Motion Estimation in Section 4.2	196
B.2	Derivations for the PDS Theory in Section 4.3	197
B.3	Supporting 3-D Imaging Surfaces in Section 4.7	199
	Vita	201

List of Tables

Table	Page
3.1 Procedures to Extract Feature Locations	36
3.2 Scan Line Approximation Algorithm	38
3.3 Distortion Models	44
3.4 Comparison Results Using Microsoft Images	49
3.5 Comparison Results Using Desktop Images	49
3.6 Comparison Results Using ODIS Images	49
3.7 Model Selection Using the GAIC and GMDL Criteria	51
3.8 Model Selection of the Simulated Camera	54
3.9 Objective Function J of the Piecewise Model	56
3.10 GAIC and GMDL Quantities of the Piecewise Model	56
3.11 Comparison of Lens Distortion Coefficients of the Blind Removal Technique and the Target-Based Algorithm	64
3.12 Blind Detection of Geometric Distortion	67
4.1 Design Parameters of Existing Perspective Observers	100
4.2 Technologies to Achieve Wide Fields of View	127
4.3 Range Identification with Known Motion Parameters Using Single Camera	140
4.4 3-D Motion Estimation for Unknown Motion Parameters Using Single Camera	141
A.1 3×3 GRO Mask	186
A.2 Decision Logic	192
A.3 Thresholds via Testing	194

List of Figures

Figure	Page
1.1 Block diagram of vision, dynamic systems, and control.	1
1.2 The physical and mechanical layout of the ODIS robot.	9
1.3 The physical gimbal as the platform for perspective ILC.	9
1.4 The T2 and T2E vehicles.	10
2.1 Basic idea of ODIS visual servoing to yellow lines.	13
2.2 Hardware architecture of the ODIS wireless visual servoing system.	14
2.3 Signal flow of the wireless visual servoing system.	15
2.4 Image processing for ODIS visual servoing.	16
2.5 Camera frame, image plane, and parking lot line.	17
2.6 Variables for kinematic equations. Point A should be understood as on an imaginary extension of the line.	19
2.7 Iterative visual servoing control scheme.	22
2.8 Visual servoing errors vs. iteration in the image frame.	25
2.9 Another example of visual servoing: errors vs. iteration in the image frame.	25
3.1 The perspective camera model.	28
3.2 The barrel distortion and the pincushion distortion.	28
3.3 Lens camera deviates from the pinhole model in locus of convergence.	29
3.4 Skewness of two image axes.	31
3.5 Camera calibration procedures.	33
3.6 Calibration object.	35
3.7 Objects and their interior 8-boundary.	37

3.8	Illustration of scan line approximation algorithm (a) 3 sides (b) 4 sides.	39
3.9	Fitting results using partitions found by scan line approximation.	39
3.10	Feature points extraction for desktop images (1).	40
3.11	Feature points extraction for desktop images (2).	40
3.12	A smooth piecewise function (two-segment).	45
3.13	Five images of the model plane with the extracted corners (indicated by cross) for the desktop camera.	47
3.14	Five images of the model plane with the extracted corners (indicated by cross) for the ODIS camera.	48
3.15	$f(r) \leftrightarrow r$ curves for the simulated camera using the models #3, 4, 6, 7, 8, 9, 10 in table 3.3.	55
3.16	Two sample images of the model plane with the extracted corners (indicated by cross) for the desktop and ODIS cameras.	62
3.17	Blindly radially compensated desktop and ODIS images with distortion function (3.38) with $k_{uv1} = k_{uv2}$	63
3.18	Relative J values of the ODIS images using function (3.46).	66
3.19	Relative J values of the ODIS images using function (3.38).	66
4.1	Simulation results of e_3 for the three observers for Example ₁	102
4.2	Simulation results of e_3 for the three observers for Example ₂ (RIO can not be applied to this example, as can be seen from fig. 4.3).	102
4.3	Simulation results of e_3 for Example ₂ using RIO.	103
4.4	Examples _{1,2} with noise level = 10^{-3}	103
4.5	Examples _{1,2} with noise level = 10^{-2}	104
4.6	$f_{1,2}$ and $\hat{f}_{1,2}$ of IBO for Example ₁	104
4.7	Linear approximations of a nonlinear system.	113
4.8	A sequence of LTV subsystems of a nonlinear system.	115

4.9	$\hat{x}_{1,2,3}^{[2]}(t)$ of (4.91) under Motion₁ without noise.	115
4.10	Matlab simulation block diagram of the LAO observer.	116
4.11	$\hat{x}_{1,2,3}^{[2]}(t)$ under Motion₁ without noise.	116
4.12	$\hat{x}_{1,2,3}^{[2]}(t)$ under Motion₁ with uniform noise bounded by $\pm 10^{-2}$	117
4.13	$\hat{x}_{1,2,3}^{[2]}(t)$ under Motion₂ without noise.	117
4.14	$\hat{x}_{1,2,3}^{[2]}(t)$ under Motion₂ with uniform noise bounded by $\pm 10^{-2}$	117
4.15	Comparison of LAO with the other three perspective non-linear observers under Motion₁ with $x_0 = [0.4, 0.6, 1.0]^T$ and $x'_0 = [0, 0, 0]^T$: (a, b) without noise; (c, d) with uniform noise bounded by $\pm 10^{-2}$	118
4.16	Comparison of LAO with the other three perspective non-linear observers under Motion₂ with $x_0 = [0.4, 0.6, 1.0]^T$ and $x'_0 = [0, 0, 0]^T$: (a, b) without noise; (c, d) with uniform noise bounded by $\pm 10^{-2}$	118
4.17	Illustration of PDS with single observation function.	120
4.18	State estimation using IBO with $[\hat{y}_2(0), \hat{y}_3(0)] = [-1, -1]$. y_1	124
4.19	State estimation using IBO with $[\hat{y}_2(0), \hat{y}_3(0)] = [0, 0]$. y_1	124
4.20	State estimation using IBO with $[\hat{y}_2(0), \hat{y}_3(0)] = [1, 1]$. y_1	125
4.21	Estimation error comparison between IBO and IBO in the y_1+y_2 y_1 ideal case of no noise.	125
4.22	Estimation error comparison between IBO and IBO in the y_1+y_2 y_1 presence of uniform noise bounded by $\pm 10^{-2}$	126
4.23	Simulation results of e_3 for the three observers for Example ₁	129
4.24	A spherical imaging surface centered at origin with radius 1.	132
4.25	A paraboloid imaging surface centered at $[0, 0, 0]^T$	135
4.26	Range identification under general imaging surfaces.	138

4.27 Object location determined via triangulation for a stationary point using moving camera.	141
5.1 Simulation results of the ILC control of a 2-D PDS system (under IIC).	147
5.2 Simulation results of the ILC control of a 2-D PDS system (without IIC).	148
5.3 Simulation results of $\ e_k\ _2 = \ y_d - y_k\ _2$ with or without the IIC condition.	148
5.4 Experimental setup for perspective ILC using gimbal system.	151
A.1 The behavior control system architecture of ODIS.	175
A.2 Object manager architecture.	177
A.3 Rectangle model fitting algorithm using real laser data.	178
A.4 Rectangle model fitting algorithm using simulated points.	179
A.5 Flow chart of the safety agent.	181
A.6 Beam pattern of the SRF04 sonars on T2E (in inch).	183
A.7 100% and 80% sensing coverage around T2E (in meter).	183
A.8 Typical laser coverage (radius is 5 m in this illustration).	183
A.9 Sonar data updating rule.	184
A.10 Illustration of errors introduced in the map projection.	187
A.11 Experimental setup.	188
A.12 The $10 \times 10 \text{ m}^2$ HIMM maps built when T2E moves toward a wall in an indoor room.	189
A.13 Decision making based on the velocity vector.	190
A.14 State machine illustrating transitions among all states.	193

Notations

Throughout this dissertation, all entities in this dissertation are assumed to be either vectors or matrices, unless otherwise mentioned. The following conventions will be used for typesetting mathematics unless otherwise indicated.

- 1) Scalar: Can be in both lower and upper cases, such as α , k_1 , C_1 , G , M .
- 2) Matrix: In upper case and non-bold form, such as A , C , H , \mathcal{A} .
- 3) Vector: In lower case and non-bold form, such as x, y , in cases of no confusion. Vectors can also be typed in lower case bold form for emphasis, such as \mathbf{b} , \mathbf{f} , \mathbf{d} . The reason for this emphasis is these variables will be used throughout this dissertation having the same meanings. Vectors can also be represented as \mathcal{P} , \mathcal{X} .
- 4) Function: In both lower and upper cases, such as $w(\cdot)$, $A(\cdot)$, $\mathcal{F}(\cdot)$.

Notation wise, most variables have only local meanings. Those that have the same representation throughout this dissertation are listed below.

Variable	Description
$[X^w, Y^w, Z^w]^T$	3-D coordinate of a point in world reference frame
$[X^c, Y^c, Z^c]^T$	3-D coordinate of a point in camera frame
(u_d, v_d)	Distorted image point in pixels
(u, v)	Distortion-free image point in pixels
$(\alpha, \gamma, \beta, u_0, v_0)$	Five intrinsic parameters of a camera, see (3.6)
$A_{\text{intr}} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$	Intrinsic matrix of a camera
(x_d, y_d)	$[x_d, y_d, 1]^T = A_{\text{intr}}^{-1}[u_d, v_d, 1]^T$
(x, y)	$[x, y, 1]^T = A_{\text{intr}}^{-1}[u, v, 1]^T$
r_d	$r_d^2 = x_d^2 + y_d^2$
r	$r^2 = x^2 + y^2$
f	Focal length of a camera lens
λ	Scalar or eigenvalues
Ω	Skew-symmetric matrix
\mathbf{t}	Translational vector
\mathbf{k}	Distortion coefficients, see (3.10)
\mathbf{d}	Essential parameters, see (4.65)
\mathbf{f}	Vector in representation of Riccati motion, see (4.43)

Acronyms

BFCS	body-fixed coordinate system
CEF	composite energy function (related to ILC)
CSOIS	Center for Self-Organizing and Intelligent Systems
CV	certainty value (related to HIMM)
DOF	degree of freedom
EKF	extended Kalman filter
FOG	fiber optic gyro
FOV	field of view of a camera
GAIC	geometric Akaike information criterion
GMDL	geometric minimum description length criterion
GRO	growth rate operator (related to HIMM)
HIMM	histogram in-motion mapping
HOSA	higher-order spectral analysis (related to Matlab toolbox)
IBO	identifier-based observer (related to PDS)
IIC	identical initial condition (related to ILC)
ILC	iterative learning control
ICS	inertial coordinate system
LAO	linear approximation-based observer (related to PDS)
LQG	linear quadratic Gaussian
LS	least squares
LTV	linear time-varying
NN	neural network
OCS	obstacle cluster strength (related to HIMM)
ODIS	omni-directional inspection system, a robot at the CSOIS center
ODV	omni-directional vehicle

PDS	perspective dynamic systems
RIO	range identification observer (related to PDS)
SVD	singular value decomposition
TLS	total least squares

Chapter 1

Introduction

1.1 Research Problems

Visual cameras are useful sensors for systems and control since they mimic the human sense of vision and allow for non-contact measurement of the environment. Visual feedback loops have been adopted in control systems, such as robots operating in factories and robot navigation. As shown in fig. 1.1, when a camera is used as a sensor to observe a target, the image captured by the camera goes through an image processing module and outputs certain information. This information is used by a controller to design the control input to the system. Motion of the system results in changes in the image and this whole process continues until the control goals are achieved.

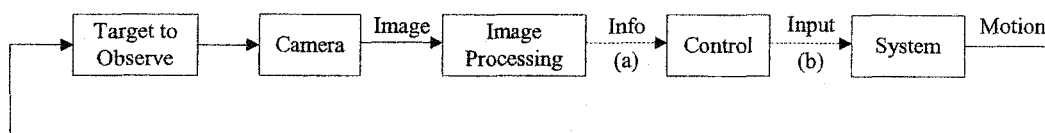


Fig. 1.1: Block diagram of vision, dynamic systems, and control.

In this dissertation, we propose to study several aspects of the use of vision for systems and control. Specifically, we consider a series of problems whose evolution is shown in the following list and depicted in fig. 1.1:

- 1) Research Problem 1 - Iterative Visual Servoing: This problem is concerned with orienting a robot to a desired pose using visual feedback. An example of this problem is to align the ODIS (short for Omnidirectional Inspection System) robot at the Center for Self-Organizing and Intelligent Systems (CSOIS)¹ with the direction of

¹CSOIS is a multidisciplinary research group at Utah State University (USU) that focuses on the design, development, and implementation of intelligent, autonomous mechatronic systems [1].

a parking lot yellow line using an uncalibrated camera. In the ODIS yellow line alignment task, the “target to observe” is the parking lot yellow line. The camera is installed on ODIS and the information extracted out of an observed image is the orientation of the yellow line. This orientation is fed into the controller to calculate the movements sent down to ODIS. Because we use an uncalibrated camera, an iterative control scheme is implemented in a start-stop manner [2, 3].

- 2) Research Problem 2 - Camera Calibration: In the ODIS yellow line alignment task described above, the iterative scheme is used to work with an uncalibrated camera, while a calibrated camera will simplify the task so that it can be performed in a non-iterative way. Camera calibration is to estimate a set of parameters, i.e. the camera’s intrinsic parameters and distortion coefficients, that describe the camera’s imaging process. With this set of parameters, a perspective projection matrix can directly link a point in the 3-D world reference frame to its projection (undistorted) on the image plane. Since virtually all imaging devices introduce a certain amount of nonlinear distortion, the observed distorted image needs to be compensated to output the corrected image. For the camera calibration problem, the “target to observe” in fig. 1.1 is the calibration target. Usually, the camera needs to observe the target at several different orientations/positions to estimate the camera parameters uniquely.
- 3) Research Problem 3 - Perspective Dynamic System (PDS): PDS theory provides a theoretical framework to study vision problems, especially 3-D motion estimation. Generally speaking, a perspective dynamic system consists of a moving target with certain motion dynamics (rigid, affine, Riccati), which is sensed using the homogeneous observations made via cameras. With a stationary camera observing a moving object, following an affine motion described by:

$$\begin{bmatrix} \dot{X}(t) \\ \dot{Y}(t) \\ \dot{Z}(t) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} X(t) \\ Y(t) \\ Z(t) \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}, \quad (1.1)$$

a typical PDS consists of the above linear dynamic system with the following homogeneous output observations:

$$y_1(t) = X(t)/Z(t), \quad y_2(t) = Y(t)/Z(t). \quad (1.2)$$

Typical problems in the PDS theory include: 1) 3-D motion estimation: estimation of the motion parameters $a_{i,j}(t)$ and $b_i(t)$ for $i, j = 1, 2, 3$ and 2) range identification: estimation of depth information, such as via nonlinear observers, when the motion parameters are known. Define:

$$y(t) = [y_1(t), y_2(t), y_3(t)]^T = [X(t)/Z(t), Y(t)/Z(t), 1/Z(t)]^T.$$

The range identification problem is to estimate $Z(t)$, or its inverse $y_3(t)$, assuming $y_1(t)$ and $y_2(t)$ are available and the motion parameters $a_{i,j}(t)$ and $b_i(t)$ for $i, j = 1, 2, 3$ are known. For the 3-D motion estimation and range identification, usually calibrated cameras are used to observe certain features on the moving target.

- 4) Research Problem 4 - Perspective Iterative Learning Control: The objective is to develop techniques to control the transient response and tracking performance of control systems whose performance is observed by cameras and whose operation is repetitive. Iterative Learning Control (ILC) can improve the transient response performance of systems that operate repetitively over a fixed time interval [4]. The control problem to be discussed is an ILC problem with perspective observations, referred to as perspective ILC hereafter.

1.2 Current State of the Art

The current state of the art of the research problems listed in section 1.1 is described next.

1.2.1 Iterative Visual Servoing

Existing eye-in-hand visual servoing approaches include [5, 6, 7]: position-based (3-D) visual servoing, image-based (2-D) visual servoing, and 2-1/2-D visual servoing. In

a position-based visual control system, the input of the control law is computed in the 3-D space. The pose of the target with respect to the camera is estimated from the observed features on the image based on a calibrated camera and the knowledge of a perfect geometric model of the object. On the other hand, in an image-based control system, the input is computed in the 2-D image plane. This usually involves the calculation of an image Jacobian which relates the rate of change of the image feature coordinates to the rate of change of the 3-D pose parameters. Both the position-based and image-based visual servoing approaches experience certain drawbacks. The main drawback of the 3-D visual servoing is that there is no control in the image. This implies that the target may leave the Field Of View (FOV) of the camera, especially if the robot or the camera is coarsely calibrated. Furthermore, a model of the target is needed to compute the pose of the camera. 2-D visual servoing does not explicitly need this model. However, the convergence is only ensured in a neighborhood of the desired position.

1.2.2 Camera Calibration

For camera calibration, lens distortion is very important for accurate 3-D measurement [8]. Among the nonlinear distortions (also called geometric distortions), radial distortion, which occurs along the radial direction from the center of distortion, is the most severe part [9, 10]. The removal or alleviation of the radial distortion is commonly performed by first applying a parametric radial distortion model, estimating the distortion coefficients, and then correcting the distortion via an inverse operation. Most of the existing works on the radial distortion models can be traced back to an early study in photogrammetry [11] where the radial distortion is governed by the following polynomial equation [11, 12, 13, 14]:

$$r_d = r + \delta_r = r(1 + k_1 r^2 + k_2 r^4 + k_3 r^6 + \dots), \quad (1.3)$$

where k_1, k_2, k_3, \dots are the distortion coefficients. r and r_d denote the ideal and the distorted version of radius, either in the camera frame or the image plane. Currently, the distortion model in (1.3) is still the most commonly used radial distortion model, typically

used with only two coefficients. However, when using two coefficients, the inverse of the polynomial function is difficult to perform analytically.

Besides the lens distortion compensation methods commonly used in the time domain, frequency domain methods have been proposed in [15] to use higher-order correlation detection to remove the lens distortion blindly. However, the accuracy of the blindly estimated lens distortion is by no means comparable to those based on a calibration target. Due to this reason, the blindly lens distortion removal approach can be useful in areas where only qualitative results are required.

1.2.3 PDS Theory

Since the 1970's, a variety of motion estimation algorithms have been developed, gaining attention from researchers and scientists in the areas of computer vision, image processing, robotics, and control. However, the goal is still far from being reached. Indeed, it opens a new and exciting avenue of research in nonlinear system theory. Appropriate tools from nonlinear estimation/identification theory are beginning to be exploited and only recently have such tools hinted at acceptable solutions [16].

A direct approach to the 3-D motion estimation problem is to formulate it as a nonlinear optimization problem, where classical optimization algorithms, such as the Gauss-Newton and the Levenberg-Marquardt optimization methods, can be used to search for the optimal solution. Though very accurate, this classical nonlinear method is computationally expensive, which prevents it from being applied to real-time applications. Thus, algorithms that are based on linearization have become prevalent. Generally, they are formulated as a minimization problem that can be solved either by Singular Value Decomposition (SVD) or recursively [16, 17, 18, 19]. Recently, exciting results have been published on what is called Perspective Dynamic System (PDS) theory, where the system's observability/identifiability property is discussed in a more theoretical way and most of the known results in the computer vision literature are revealed [20, 21]. Besides the rigid motion, nonrigid motions such as the affine and Riccati motions can be discussed in this general framework [22].

For 3-D motion estimation, theoretical analysis has been derived for the identification, to the extent possible, of rigid, affine, and Riccati motions. It can be concluded that with a single camera observing a single feature point, not all the motion parameters can be identified uniquely.

For the range identification problem, nonlinear observers have been constructed for the estimation of the depth [23, 24, 25].

1.2.4 Perspective ILC

Iterative learning control, or ILC, is a technique for improving the transient response and tracking performance of processes, machines, equipment, or systems that execute the same trajectory, motion, or operation over and over. The approach is motivated by the observation that if the system controller is fixed and if the system's operating conditions are the same each time it executes, then any errors in the output response will be repeated during each operation. These errors can be recorded during system operation and can then be used to compute modifications to the input signal that will be applied to the system during the next operation. That is, in ILC, refinements are made to the input signal after each trial until the desired performance level is reached. It is usually assumed implicitly that the initial conditions of the system are reset at the beginning of each trial to the same value. In describing the technique of ILC, the word *iterative* is used because of the recursive nature of the system, and the word *learning* is used because of the refinement of the input signal based on past performance in executing a task [26]. Most of the current ILC algorithms use the encoder readings of the plant as the feedback information. In the perspective ILC, vision measurement serves as the actual feedback. To date, there are few contribution in the ILC literature where vision feedback is used. Further, there have been no contribution that exploits the PDS theory in an ILC problem.

1.3 Contributions of This Dissertation

In this dissertation, we present contributions to each of the four areas discussed above. The original contributions are summarized below.

- 1) Iterative Visual Servoing: While most visual servoing approaches require a calibrated camera, visual servoing with an uncalibrated camera is pursued in the ODIS yellow line alignment problem. The basic idea for ODIS localization is simply the visual servoing to the yellow line painted on the ground of the parking lot, where the yellow line can be regarded as a landmark for localization. It is assumed that from a map the position and orientation of each yellow line are known. The major objective is to reset the Fiber Optic Gyro (FOG) so that the FOG drift can be kept small. Since a real-time requirement is not critical here, the task can be achieved with an iterative scheme using an uncalibrated camera. This work has been published in the IEEE International Conference on Robotics and Automation and IFAC World Congress [2, 3].

- 2) Camera Calibration: Inspired by the polynomial distortion approximation function (1.3), we have proposed a set of rational functions for both the radial and geometric distortion models. Further, we proposed a simplified geometric distortion modeling idea that allows different distortion coefficients along the two image axes. A frequency domain lens distortion removal technique was also proposed for the detection of cameras that can be better modelled by the simplified geometric distortion modeling. In all the distortion models, we emphasize the property of having an analytical inverse formula, which is a desirable feature for applications requiring real-time implementation. This work has been published in international conferences and journal [27, 28, 29].

- 3) PDS Theory: For the range identification problem, we have performed preliminary comparisons for the applied nonlinear observers. Research efforts have been conducted to apply a recently proposed linear approximation idea for the perspective nonlinear observer design and discuss the situation with single homogeneous observation. This work has been published in the IEEE International Conference on Robotics and Automation [30, 31].

- 4) Perspective ILC: Our preliminary study shows that, under the condition of Identical Initial Condition (IIC), the homogeneous output might be enough to force the states of the system to track a desired trajectory. However, when the IIC condition is not satisfied, state tracking can not be achieved simply by using the homogeneous outputs. It is at this point that the PDS theory can be helpful for the estimation of the states.

Aside: Note that during this dissertation research, some efforts were applied to non-vision-based sensing and perception for dynamic systems and control. For completeness, these results are detailed in Appendix A.

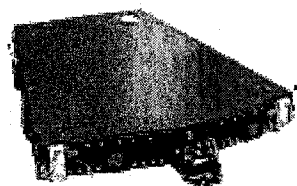
1.4 Experimental Platforms Introduction

In this dissertation, the following experimental platforms have been, or are proposed to be, used as the test bed for the sensing and perception algorithms. Figure 1.2 shows the ODIS system, which was used for experiments associated with algorithms in chapters 2 and 3. Figure 1.3 shows the gimbal system described completely in chapter 5. Figure 1.4 shows the T2/T2E system used for the non-vision-based sensing and perception algorithms described in Appendix A.

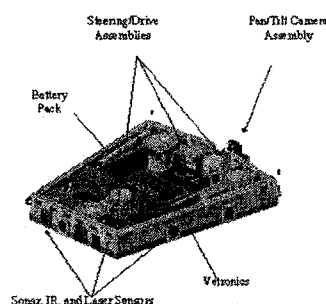
1.5 Reading the Dissertation

The linkage of the four research problems discussed in this dissertation is presented by the following application: consider an application scenario where a target is to track a 3-D trajectory repetitively and a camera is used for monitoring. Using the “iterative visual servoing” scheme, the target can be controlled to first go to the desired initial position. With a calibrated camera, from the results of “Perspective ILC,” the 3-D tracking can be achieved from 2-D observations from the camera. However, if the target tries to track the 3-D trajectory from its current position directly (which is different from the desired initial position), 3-D estimates of the moving target need to be available. In this case, results of “range identification of a PDS” can be helpful. In the above described application, the motion dynamics of the moving target is assumed known.

ODIS



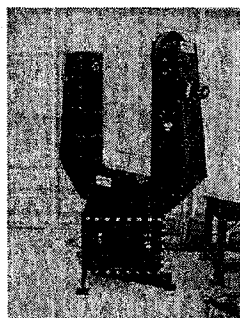
ODIS, as shown in the left, is an autonomous robot capable of surveying a parking lot and investigating the undercarriage of a suspicious vehicle. ODIS, short for Omni-Directional Inspection System, is 3-3/4-inches tall and incorporates IR, sonar, and laser range sensors, three-wheel independent drive and steering, and a color camera with transmitter. A fiber-optic gyro (FOG) is used in conjunction with wheel odometry for navigation. The ODIS vehicle has also been designed to incorporate a GPS receiver [1].



For object detection and avoidance, ODIS is equipped with three types of ranging sensors. Sonar provides ODIS with long-range, wide-angle data. Infrared sensors act as a “safety bumper” to alert ODIS that an object is too close. A laser ranging module provides very precise medium range data for accurate placement of an object in the operating environment. A color camera on a pan/tilt mechanism provides feedback to the user through a virtual reality system [1].

Fig. 1.2: The physical and mechanical layout of the ODIS robot.

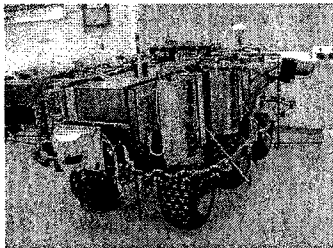
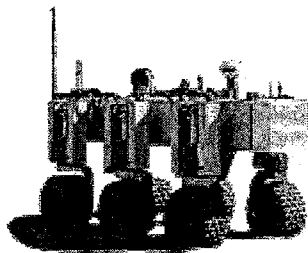
Gimbal



The gimbal system is a 2 Degree-of-Freedom (DOF) system. For vision-based tasks, a laser pointer is hooked on the gimbal. The projection of the laser ray on a 2-D object plane is observed by a stationary calibrated camera. Information on the observed image plane is a homogeneous representation of the motion on the object plane. This platform will be used in the perspective ILC in chapter 5.

Fig. 1.3: The physical gimbal as the platform for perspective ILC.

T2E



The T2 vehicle, shown in the left, follows a distributed processor architecture used by the early T-Series vehicles built at CSOIS. The T2 master node consists of a Pentium-class, single-board computer, and it communicates with the base station computer via a RF LAN, issuing set points to the wheel control nodes. The base station computer hosts the Vehicle GUI and Path Planner. The navigation node is also a Pentium-class single board computer and communicates to the master node through the LAN. The navigation node handles all communications with GPS and FOG, processing this sensor data and sending it to the master node for use in odometry and control [1].

The T2E robot is the T2 robot with enhanced sensor suit. The sensors that are equipped with the T2E include a sonar ring, a 2-D laser, and a license plate recognition system.

Fig. 1.4: The T2 and T2E vehicles.

Chapter 2

Iterative Visual Servoing

This chapter is concerned with the sensing and perception algorithm for robot localization. The sensor being used is a camera-type vision system. This chapter also works as the motivation for the study of lens distortion correction to be addressed in chapter 3. The robotic platform on which the sensing and perception algorithm has been implemented is the ODIS robot described in section 1.4.

The ODIS robot, which has been described in section A.1, is designed to perform under car inspection by sending wirelessly captured video frames to the GUI of the base station. The working period can be hours long for many parking lots. Therefore, it is important for ODIS to be able to localize and estimate its pose with respect to an internal world model (map). Although some techniques can be used to improve the FOG accuracy [32], the accumulative errors due to FOG and ODIS odometry can be progressively dominant and may significantly deflect the mobile robot pose (position and orientation in world coordinate system). Therefore, from time to time, localization is essential for a reliable task execution of ODIS. Mobile robot localization is an on-going research topic with no unique solution [33, 34] since the localization depends on the structure of the environment and the sensing capacities of the specific mobile robot. In general, localization is environment and robot specific.

In our case, the environment is the car parking area. We can use the laser or sonar to do localization based on landmarks such as lamp posts, curbs etc. However, the most commonly seen landmarks in the parking area are the yellow lines painted on the ground. In this chapter, we use visual servoing to a yellow line in a parking lot to perform the localization task. As explained above, the challenge here is that the camera is designed for the under car inspection video transmission and within ODIS there is *no* video capture card. Thus, our approach is to capture the video frames, sent wirelessly from the ODIS

camera transmitter, on the base station computer (ODIS laptop computer) and perform the visual servoing to yellow line to achieve ODIS localization by wirelessly sending the scripts to ODIS from the base station computer. Therefore, the research efforts reported in this chapter can be regarded as a *value-added block* for ODIS functionality. Based on the current architecture of ODIS and the dedicated scripting language [35], an iterative visual servoing scheme is developed to align the yellow line painted on the ground of the parking lot. The iterative scheme tolerates the uncertain time delay due to the wireless connections without introducing stability problem due to time-varying delay in real-time visual servoing. Experimental results are presented to show that for our specific application, the wireless visual servoing technique presented in this chapter is an effective approach to robot localization.

In this chapter, a “dynamic image-based look-and-move” approach is used. It is “image-based” since errors are computed in the image plane. It is “dynamic look-and-move” since the errors determine corrective velocities for the robot. In contrast to most image-based approaches, the method here does not require the use of an image Jacobian matrix. Further, although the image errors are nonlinearly related to the vehicle position errors, with significant coupling between degrees of freedom, convergence to the desired position and orientation occurs for virtually any initial condition of interest.

The chapter is organized as follows. Section 2.1 presents the basic idea and implementation architecture for ODIS wireless visual servoing in some detail, including the feature extraction of yellow lines in the image plane. Section 2.2 describes the iterative visual servoing controller scheme. Due to the limitations of the current software architecture, in section 2.3, a discrete look-and-move controller is currently implemented on the ODIS platform. Experimental results are presented in section 2.4. Some concluding remarks together with some of our planned further efforts are given in section 2.5.

2.1 ODIS Localization via Wireless Visual Servoing

This section describes the basic idea and implementation architecture for ODIS wireless visual servoing, together with some implementation details, including the low-level yellow line extraction and fitting.

2.1.1 Basic Idea

The basic idea for ODIS localization is simply the visual servoing to yellow lines painted on the ground of the parking lot, as illustrated in fig. 2.1. We can regard the yellow line of the parking lot as a landmark for localization. It is assumed that from the map the position and orientation of each yellow line are known. The command for ODIS to perform yellow-line alignment is from the high-level block, e.g., from either planner or just a GUI click. The major objective is to reset the FOG after inspection of several parking lots so that the drifting can be kept small. Obviously, this can be done via a dedicated utility script sent wirelessly from the base station to ODIS. Moreover, unlike the image-based path tracking [36, 37, 38], the real-time requirement is not critical here.

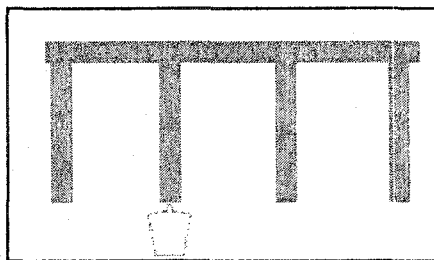


Fig. 2.1: Basic idea of ODIS visual servoing to yellow lines.

2.1.2 Hardware Architecture

Figure 2.2 shows the hardware architecture for the wireless visual servoing for ODIS localization. The ODIS laptop is a Pentium-3 notebook computer with Windows 2000 installed and a build-in image capture card. The GANZ CM3000 camera unit is a compact module including a small color camera and a video transmitter installed on ODIS.

The video signal is sent over a 900 MHz transmitter (CVR-1000) to a receiver (CVT-M) located at the base station. Both the transmitter and the receiver are made by Coherent Communications and operate in several bands within the 902 - 928 MHz range, which allows us to operate without interfering with the other onboard modems. By connecting the receiver to the video capture card inside the ODIS laptop, live images can be grabbed continuously via Microsoft Vision SDK (software development kit). Likewise, through the wireless LAN, scripts or commands can be sent to ODIS. Meanwhile, ODIS status and position/orientation can be queried from the base station computer. Therefore, a closed loop is formed via wireless connections.

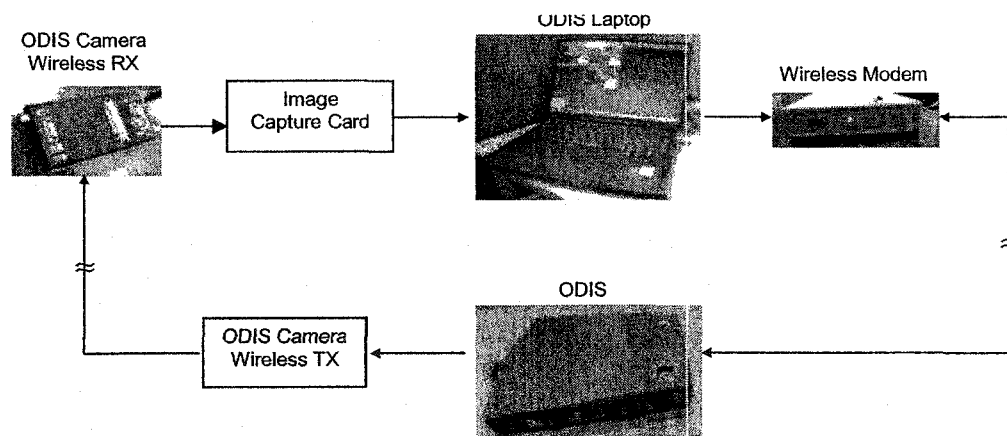


Fig. 2.2: Hardware architecture of the ODIS wireless visual servoing system.

The signal flow in the above closed-loop system is shown in fig. 2.3. ODIS continuously sends live image frames to ODIS laptop (Host). For each image frame, if there is yellow line, the yellow line detection function calculates its angle and the starting point in the image frame which serves as the output measurement of the visual servoing system. The control strategy block compares the measured output with the desired one and constructs the corresponding translation and/or rotation commands which are to be sent wirelessly down to ODIS. ODIS low-level control subsystems execute these commands (actuation).

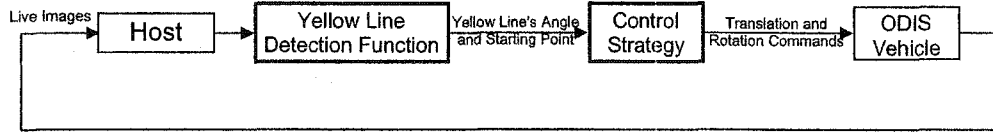


Fig. 2.3: Signal flow of the wireless visual servoing system.

2.1.3 Software Architecture

The Host (ODIS laptop) is based on the WINDOWS 2000 platform. We use Microsoft Vision SDK to acquire the live images. Microsoft Vision SDK (short for Software Development Kit) includes an MFC Application Wizard which is easily used to create MFC programs with selected functionalities of the Vision SDK. The resulting programs will attach a sequence to an image source and support a second thread capturing images in the background. One major feature of Microsoft Vision SDK is that it provides a basic image object that supports a diversity of pixel formats such as, for color images, RGBA pixel types (red, green, blue, and alpha).

Using socket programming, the visual servoing module can be independently capable of talking to ODIS bidirectionally. This independent module can be easily integrated into the base station GUI or planner as an additional function or utility script. Basically, the visual servoing module is triggered by the base station GUI or planner and returns an event completion flag and some possible exceptional flags. Central in the module are the yellow line detection function and the visual servo algorithm which will be described in some detail below.

2.1.4 Yellow Line Detection Function

The five steps in the yellow line detection function are 1) yellow color separation, 2) connected component labelling, 3) maximal region determination - line region mask, 4) points for line determination, and 5) line fitting. In step 1), we use RGB components to create a difference image by the formula $Red + Green - 2 \cdot Blue$. For each row in the difference image, a threshold value is calculated using an average of the minimum and maximum difference values in the row. Finally each row is thresholded and the result

is stored in what we call an *enhanced mask*. In step 2), connected component labelling is applied to find all equivalence classes of connected pixels in a binary image using the 8-connectivity principle [39]. Taking the *enhanced mask* as input, the output via the connected component labelling is another image called the *region map*, in which every pixel in one connected region is labelled “1” and every pixel in another connected region is labelled “2.” In step 3), with a *region map* obtained in step 2), we choose the region with the maximal number of pixels to be what we call the *line region mask*. Step 4) determines the points to form a line for line fitting in step 5). This is done by scanning the *line region mask* from both left and right until reaching the middle. The middle points are used to estimate the yellow line parameters such as the angle and the starting point position. Finally, in step 5), with the data set obtained in step 4), an isotropic line fitting algorithm is used to determine the yellow line parameters: orientation angle and the starting point position. Figure 2.4 shows the images at each step of the yellow line detection function described above.

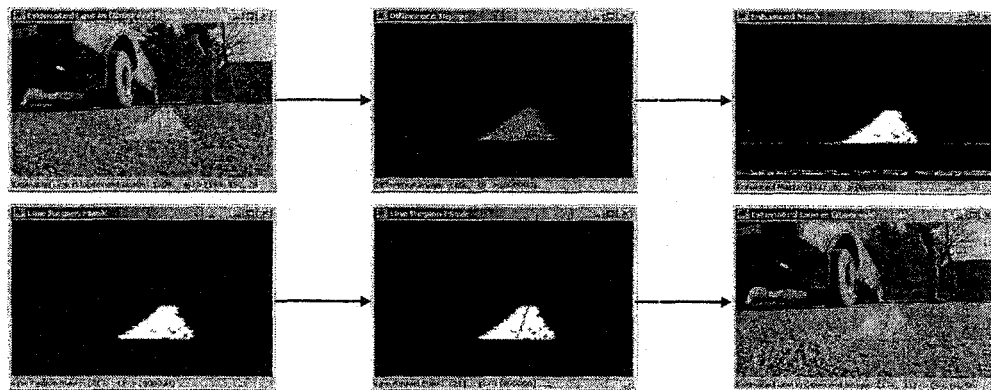


Fig. 2.4: Image processing for ODIS visual servoing.

2.2 Control Scheme and Stability Analysis

As part of the CSOIS effort on ODIS, an iterative visual servoing scheme is proposed for the robot localization. In this section, a portion of this research is described. This

material is adapted and in some cases excerpted from [2]. The original idea is credited to the first author of [2]. However, the implementation of these ideas was a contribution of this dissertation. This portion of material is provided for completeness.

2.2.1 Perspective Map

A perspective projection model is assumed for the camera. The projection map is given by:

$$(x, y) = \frac{f}{Z}(X, Y), \quad (2.1)$$

where (x, y) are **projected** projections on the image plane (not in pixels).

Figure 2.5 illustrates conventions for the camera frame, image plane, and parking lot line. Consider a point in the camera frame with coordinates (X, Y, Z) . Now, add the unit vector $(\sin \varphi, 0, \cos \varphi)$ to this point. It can be assumed that $-\pi/2 < \varphi < \pi/2$. The angle that the image of this vector makes with respect to the positive y -axis in the image plane can be easily derived.

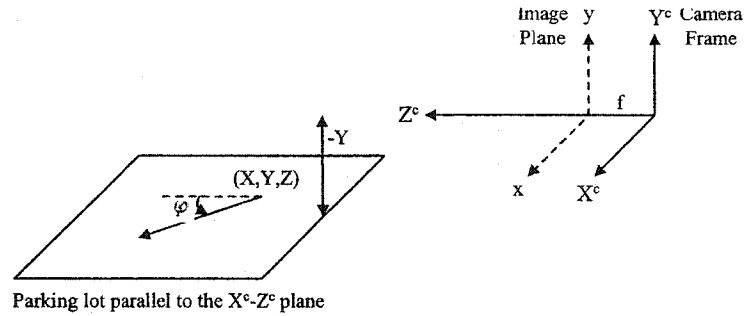


Fig. 2.5: Camera frame, image plane, and parking lot line.

The mapping of the two points, under the perspective projection, are given by:

$$(X, Y, Z) \mapsto \frac{f}{Z}(X, Y), \quad (2.2)$$

$$(X + \sin \varphi, Y, Z + \cos \varphi) \mapsto \frac{f}{Z + \cos \varphi}(X + \sin \varphi, Y). \quad (2.3)$$

Taking the difference of the two image points gives a vector in the image plane

$$\frac{f}{Z(Z + \cos \varphi)}(Z \sin \varphi - X \cos \varphi, -Y \cos \varphi). \quad (2.4)$$

Note that the vector always has a positive y component since $Y < 0$ (the camera is above the parking lot). The angle of this vector with respect to the positive y -axis is then given by:

$$\theta = \sin^{-1}\left(\frac{Z \tan \varphi - X}{\sqrt{(Z \tan \varphi - X)^2 + Y^2}}\right), \quad (2.5)$$

where the angle is positive if the vector has a positive x -component. Note that, based on fig. 2.5, the image plane should be visualized with x positive in the left direction, so positive θ is then in the counterclockwise direction.

2.2.2 Proportional Controller

Using the image processing algorithms described earlier to extract the starting point and the direction of a parking lot line, a simple controller (based on intuition) is given by:

$$\begin{aligned} \dot{X}_v &= k_1(x - x_d), \\ \dot{Z}_v &= k_2(y - y_d), \\ \dot{\varphi}_v &= k_3(\theta - \theta_d), \end{aligned} \quad (2.6)$$

where $k_i, i = 1, 2, 3$ are positive scalar controller gains, x, x_d are the horizontal coordinates of the actual and desired point in the image plane, respectively, y, y_d are the vertical coordinates of the actual and desired point in the image plane, respectively, and θ, θ_d are the actual and desired angles that the line makes with the positive y -axis in the image, respectively. \dot{X}_v is the X component of the vehicle's velocity and is positive when the vehicle moves to the left. \dot{Z}_v is the Z component of the vehicle's velocity and is positive when the vehicle moves forward. $\dot{\varphi}_v$ is the angular velocity of the vehicle and is positive when the vehicle rotates counterclockwise. On the other hand, $\dot{X}, \dot{Z}, \dot{\varphi}$ are rates associated with the movement of features with respect to the camera frame.

The motivation behind this controller is clear (see fig. 2.6). If the line appears too far to the left in the image, then the robot should move to the left. If the line appears too high in the image, the robot should move forward. Finally, if the line is rotated counterclockwise beyond the desired angle in the image, then the vehicle should rotate counterclockwise.

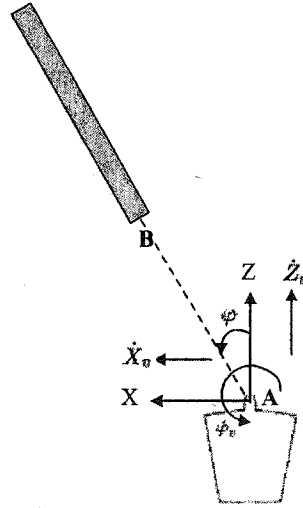


Fig. 2.6: Variables for kinematic equations. Point A should be understood as on an imaginary extension of the line.

Consider the case

$$x_d = 0, \quad \theta_d = 0. \quad (2.7)$$

Thus, the desired image is a line which is centered horizontally and oriented vertically. Setting $x = 0, y = y_d, \theta = 0$ in the projection map (2.1) gives desired values:

$$X = 0, \quad Z = fY/y_d, \quad \varphi = 0. \quad (2.8)$$

Thus, if the image line is in the desired position and orientation, the vehicle must be oriented parallel to the line, and if the line were extended toward the vehicle, the vehicle would be located on this extended line.

There are several reasons why it is not clear if the controller will work. First of all, the relationship between image and actual positions is related by the perspective map, which is nonlinear. More significantly, however, the angle of the line in the image can sometimes direct the vehicle to rotate in the wrong direction. Recall that parallel lines in the world will map to converging lines in the image. Each of these lines has a different angle with respect to the positive y -axis.

Subsequent sections will demonstrate the surprising result that the controller produces asymptotic convergence to the correct orientation and position for virtually any initial position and orientation of interest.

2.2.3 Vehicle Kinematic Equations

Figure 2.6 illustrates the important variables involved in the vehicle kinematic equations. The equations can be derived from the standard relative motion equation for rigid bodies:

$$v_B = v_A + \omega \times r_{B/A}, \quad (2.9)$$

where A and B are points on a rigid body, v_B is the velocity vector of B, v_A is the velocity vector of A, ω is the angular velocity vector of the body, and $r_{B/A}$ is the relative position vector drawn from A to B. Points A and B are labelled in fig. 2.6. All calculations will be done with respect to the camera frame. $v_B = (\dot{X}, \dot{Z})$ is the velocity of the point B as seen in the camera frame. As the vehicle moves with velocity (\dot{X}_v, \dot{Z}_v) , the point A on the line (or its imaginary extension) will appear to move with velocity $-(\dot{X}_v, \dot{Z}_v)$. Thus, $v_A = -(\dot{X}_v, \dot{Z}_v)$. Clearly, $r_{B/A} = (X, Z)$. Finally, the line will appear to rotate about the point A in the counterclockwise direction if φ is increasing. Thus, ω has magnitude $\dot{\varphi}$ and is in the counterclockwise direction when $\dot{\varphi} > 0$. Note that φ increasing means $\dot{\varphi}_v < 0$. Substituting these results into the relative motion equation and adding an equation corresponding to the previous statement gives the vehicle kinematic equations:

$$\begin{aligned} \dot{X} &= -\dot{X}_v + \dot{\varphi}Z, \\ \dot{Z} &= -\dot{Z}_v - \dot{\varphi}X, \\ \dot{\varphi} &= -\dot{\varphi}_v. \end{aligned} \quad (2.10)$$

2.2.4 Final Model and Stability Analysis

Combining equations (2.1 - 2.10) gives the following set of differential equations:

$$\begin{aligned} \dot{X} &= -k_1 f \frac{X}{Z} - k_3 Z \sin^{-1}(\cdot), \\ \dot{Z} &= -k_2 (f \frac{Y}{Z} - y_d) + k_3 X \sin^{-1}(\cdot), \\ \dot{\varphi} &= -k_3 \sin^{-1}(\cdot), \end{aligned} \quad (2.11)$$

where $\sin^{-1}(\cdot)$ denotes $\sin^{-1}(\frac{Z \tan \varphi - X}{\sqrt{(Z \tan \varphi - X)^2 + Y^2}})$ for short. For easy derivation and simplicity, assume that $f = 1$ and $y = -1$ (since the camera is above the parking lot on which the yellow line is painted). Equation (2.11) becomes

$$\begin{aligned}\dot{X} &= -k_1 \frac{X}{Z} - k_3 Z \sin^{-1}(\cdot), \\ \dot{Z} &= k_2 \left(\frac{1}{Z} + y_d \right) + k_3 X \sin^{-1}(\cdot), \\ \dot{\varphi} &= -k_3 \sin^{-1}(\cdot),\end{aligned}\tag{2.12}$$

with $\sin^{-1}(\cdot)$ denoting $\sin^{-1}(\frac{Z \tan \varphi - X}{\sqrt{(Z \tan \varphi - X)^2 + 1}})$. Equation (2.12) has a single equilibrium at

$$(0, -\frac{1}{y_d}, 0)^T.$$

Let $\mathbf{w} = [X, Z, \varphi]^T$. The system in equation (2.12) is of the form $\dot{\mathbf{w}} = \mathcal{F}(\mathbf{w})$ with the equilibrium point $\mathbf{w}_0 = [0, -\frac{1}{y_d}, 0]^T$. We will use Lyapunov's indirect method to test the stability of the equilibrium.

First, we can easily compute

$$\frac{\partial \sin^{-1}(\cdot)}{\partial \mathbf{w}} = (-1, 0, Z).\tag{2.13}$$

Then, the linear approximation to the system (2.12) can be written by

$$\begin{aligned}\frac{\partial \mathcal{F}}{\partial \mathbf{w}}|_{\mathbf{w}_0} &= \begin{bmatrix} k_3 Z - \frac{k_1}{Z} & 0 & -k_3 Z^2 \\ 0 & -\frac{k_2}{Z^2} & 0 \\ k_3 & 0 & -k_3 Z \end{bmatrix} |_{\mathbf{w}_0} \\ &= \begin{bmatrix} -\frac{k_3}{y_d} + k_1 y_d & 0 & -\frac{k_3}{y_d^2} \\ 0 & -k_2 y_d^2 & 0 \\ k_3 & 0 & \frac{k_3}{y_d} \end{bmatrix} \\ &= \begin{bmatrix} -\frac{\hat{k}_3}{y_d} + y_d & 0 & -\frac{\hat{k}_3}{y_d^2} \\ 0 & -\hat{k}_2 y_d & 0 \\ \hat{k}_3 & 0 & \frac{\hat{k}_3}{y_d} \end{bmatrix} k_1,\end{aligned}\tag{2.14}$$

where

$$\hat{k}_2 = -y_d \frac{k_2}{k_1}, \quad \hat{k}_3 = \frac{k_3}{k_1}.\tag{2.15}$$

The characteristic equation is then

$$(\lambda - \hat{k}_2 y_d)(\lambda^2 - y_d \lambda + \hat{k}_3) = 0, \quad (2.16)$$

whose eigenvalues are

$$\lambda_1 = \hat{k}_2 y_d, \quad \lambda_{2,3} = \frac{y_d \pm \sqrt{y_d^2 - 4\hat{k}_3}}{2}. \quad (2.17)$$

Given the constraints $y_d < 0$, $\hat{k}_2 > 0$ and $\hat{k}_3 > 0$, all the above three eigenvalues have negative real parts. This leads to the following:

Theorem 1 (from [2]) *If $y_d < 0$, $\hat{k}_2 > 0$ and $\hat{k}_3 > 0$, then the equilibrium $(0, -\frac{1}{y_d}, 0)^T$ of the visually servoed, omnidirectional robot equation (2.12) is locally asymptotically stable.*

2.3 Extended Discrete-Time Proportional Controller

Due to the hardware architecture (section 2.1.2), currently, the wireless visual servoing control scheme is implemented in a start-stop manner or as an iterative controller as described by fig. 2.7.

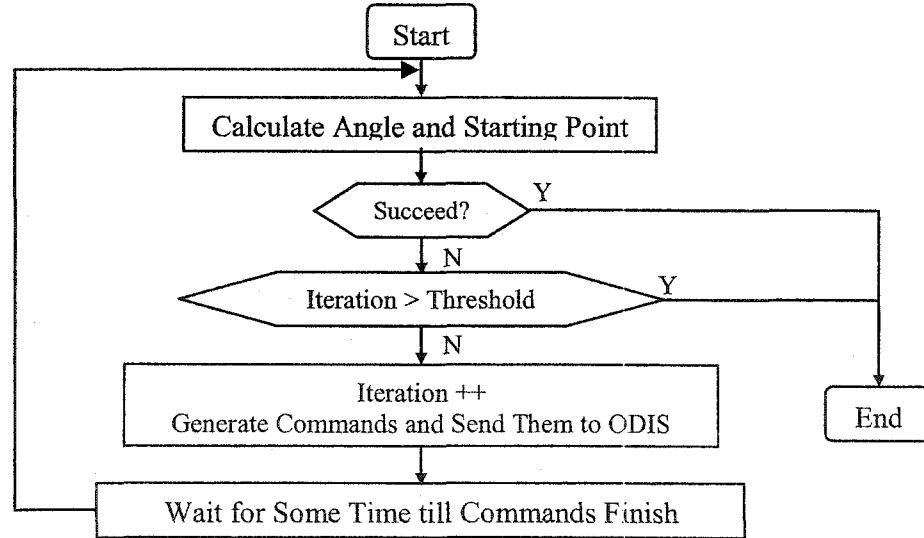


Fig. 2.7: Iterative visual servoing control scheme.

The iterative visual servoing controller is a simple proportional controller, given by

$$\Delta X_v = C_1 (u_d - u), \quad (2.18)$$

$$\Delta Z_v = C_2 (v_d - v), \quad (2.19)$$

$$\Delta \varphi_v = C_3 (\theta - \theta_d), \quad (2.20)$$

where (u, v) refers to the conventional image plane coordinates (in pixels in (2.18) - (2.20)). ΔX and ΔZ are incremental translation distances in X and Z direction in the camera frame respectively; $\Delta \varphi$ is the yaw angle increment with respect to the current ODIS orientation; the yellow line angle θ is positive if clockwise with respect to v axis in the image frame; u_d and v_d are the desired position in the image plane which correspond to the yellow line aligned vertically in the middle of the image plane (i.e., $\theta_d = 0$); and C_1, C_2, C_3 are all controller gains which were roughly determined during our tests.

During our implementation of the visual servoing control algorithm described in the above, there are several technical details that needed to be addressed. The following listed are some of the important issues.

- 1) **Yellow line searching:** Upon receipt of localization command via visual servoing, it may happen that the yellow does not lie in the field of view (FOV) of the ODIS camera. A routine is designed to automatically locate the yellow line within the FOV of camera such that the iterative visual servoing routine can start to work. If there is no yellow line found after a 360° trial, then we report an exceptional event to the GUI.
- 2) **Upper half-image plane clipping:** It may happen that a yellow colored car appears in the captured image. This will greatly affect the yellow line detection. Fortunately, since the car image is usually in the upper half of the image plane, simply clipping the upper half and using only the lower half plane for yellow line detection will solve the “yellow car disturbance problem.”

- 3) **Image frame skipping:** Due to the wireless connections, not all video frames have good quality. Occasionally, the a frame can be very bad and not usable due to e.g. E/M interference. We set a lower threshold and check the number of yellow pixels in the lower half of the image plane and then discard those images containing few yellow pixels.
- 4) **Median filtering:** A window buffer is built to store the yellow line parameters and a median filter is used to make sure that the “measurement” is not jumping due to the wireless connections.

2.4 Experimental Results

We performed tests in the parking area near outside the CSOIS “Blue Room.” We did not perform any camera calibration and the visual servoing controller gains $C_j (j = 1, 2, 3)$ are determined by experimental trials. Cautious or smaller gains can ensure the convergence of the iterative visual servoing process while suffering from slow convergence rate. On the other hand, too aggressive or larger gains may oscillate the whole system although they may make the convergence faster. A suitable design of the controller gains largely depends on the modeling efforts. Systematic and optimal design of the control gains or the development of the other control laws for the visual servoing system are our possible future efforts.

We report here a typical wireless visual servoing experimental result. The results are summarized in fig. 2.8 where we can see that the visual servoing errors in the image frame converge to near zero as the number of iterations increases. After satisfactory alignment to the yellow line with a number of visual servoing control iterations, the ODIS can be thought of being localized and then a reset FOG command can be sent to ODIS (the FOG is reset to the orientation of the yellow line as given in the assumed world map). Figure 2.9 shows another experimental result where the ODIS starts at a different initial position.

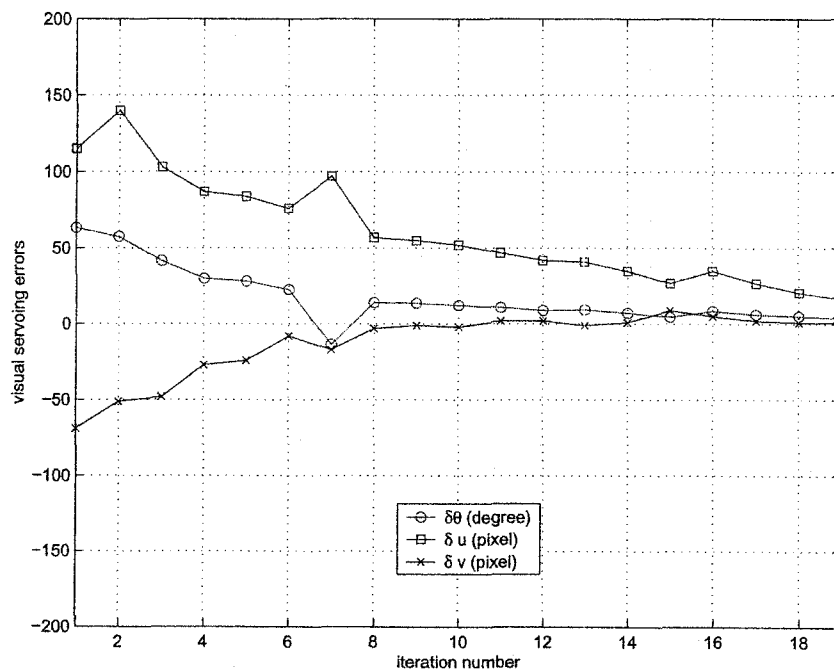


Fig. 2.8: Visual servoing errors vs. iteration in the image frame.

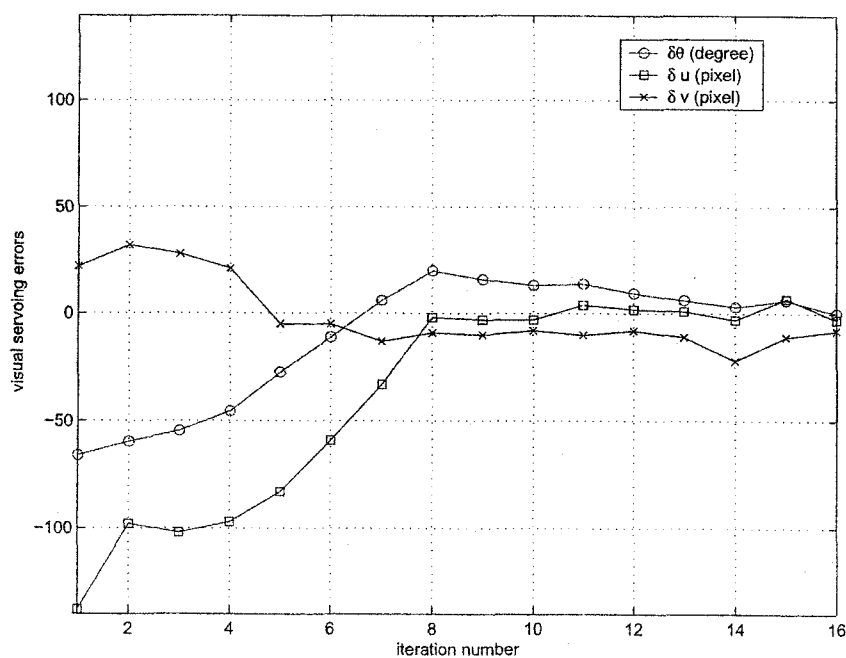


Fig. 2.9: Another example of visual servoing: errors vs. iteration in the image frame.

2.5 Discussion and Conclusions

We have presented in this section a simple robot localization technique using a wireless visual servoing technique for an autonomous ground vehicle ODIS (omnidirectional inspection system) used for under-car inspection tasks in standard parking lot environment. Based on the current architecture of ODIS and the dedicated scripting language, an iterative visual servoing scheme is proposed to align the yellow line of the parking lot. The iterative scheme tolerates the uncertain time delay due to the wireless connections without introducing stability problems due to the time-varying delays in real-time visual servoing. Experimental results were presented to show that for our specific application, the wireless visual servoing technique presented is an efficient way for robot localization.

A near future effort is to use “continuous” visual servoing instead of the “iterative scheme” presented in this section. To do this, we need to change the interface between the scripting command and the Sensor Motion Scheduler (SMS) to receive in real-time the time-varying reference signal for tracking purpose. Presently, the SMS only accepts set-point commands for motion control. With a “reference tracking mode” added to the SMS, the “continuous” visual servoing can be done easily. However, in this case, the control period and delays will be important and stability issues will be dominant.

The current yellow line alignment method applies an iterative scheme with an uncalibrated camera. If the camera is calibrated before hand, the task can be simplified by estimating the depth, either via the known motion dynamics of the vehicle, or via the known knowledge of the two parallel lines.

Chapter 3

Camera Lens Distortion Correction

Chapter 2 describes an iterative visual servoing approach with an uncalibrated camera. In this chapter, we consider the problem of lens distortion correction related to camera calibration.

3.1 Background Introduction

To use the information provided by a computer vision system, it is necessary to understand the geometric aspects of the imaging process, where the projection from 3-D world reference frame to the camera's image plane (2-D) causes direct depth information to be lost so that each point on the image plane corresponds to a ray in the 3-D space [40]. The most common geometric model of an intensity imaging camera is the *perspective* or *pinhole* model (fig. 3.1 [40]). The model consists of the image plane and a 3-D point O^c , called the *center* or *focus of projection*. The distance between the image plane and O^c is called the *focal length* and the line through O^c and perpendicular to the image plane is the *optical axis*. The intersection between the image plane and the optical axis is called the *principal point* or the *image center*. As shown in fig. 3.1, the image of P^c is the point at which the straight line through O^c and P^c intersects the image plane. The basic perspective projection [41] in the camera frame is

$$(x, y) = \frac{f}{Z^c}(X^c, Y^c), \quad (3.1)$$

where $P^c = [X^c, Y^c, Z^c]^T$ is a 3-D point in the camera frame and $p = [x, y]^T$ is its projection in the camera frame. In the camera frame, the third component of an image point is always equal to the focal length f . For this reason, we can write $p = [x, y]^T$ instead of $p = [x, y, f]^T$.

3.1.1 Aspects That Real Cameras Deviate from Pinhole Model

A real camera deviates from the pinhole model in several aspects. The most significant effect is lens distortion. Because of various constraints in the lens manufacturing process,

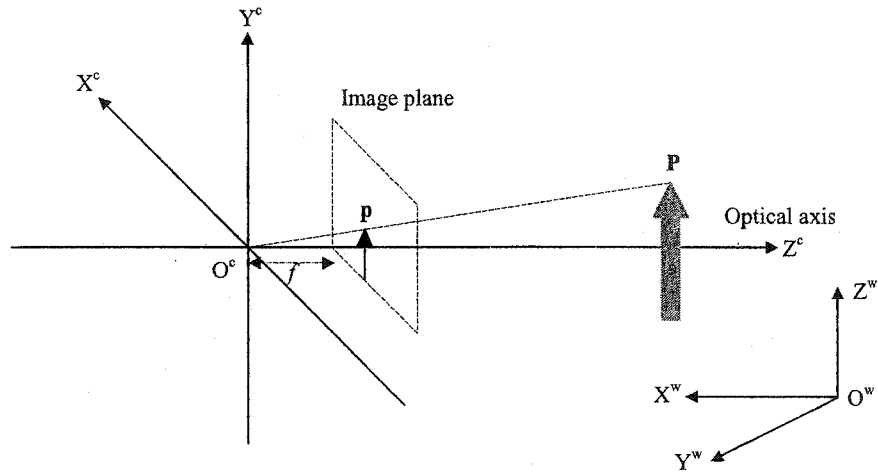


Fig. 3.1: The perspective camera model.

straight lines in the world imaged through real lenses generally become curved in the image plane. This distortion is almost always radially symmetric and is referred to as the *radial distortion*. The radial distortion that causes the image to bulge toward the center is called the *barrel distortion*, and distortion that causes the image to shrink toward the center is called the *pincushion distortion* [42] (see fig. 3.2). The center of the distortions is usually consistent with the image center.

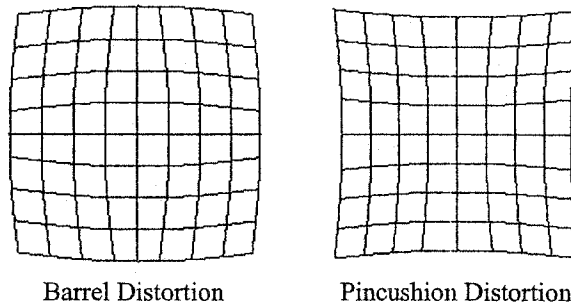


Fig. 3.2: The barrel distortion and the pincushion distortion.

The second deviation is the flatness of the imaging media. However, digital cameras, which have precisely flat and rectilinear imaging arrays, are not generally susceptible to this kind of distortion. Another deviation is that the imaged rays do not necessarily intersect

at a point, which means there is not a mathematically precise principal point as illustrated in fig. 3.3. This effect is most noticeable in extreme wide-angle lenses. But the locus of convergence is almost small enough to be treated as a point especially when the objects being imaged are large with respect to the locus of convergence.

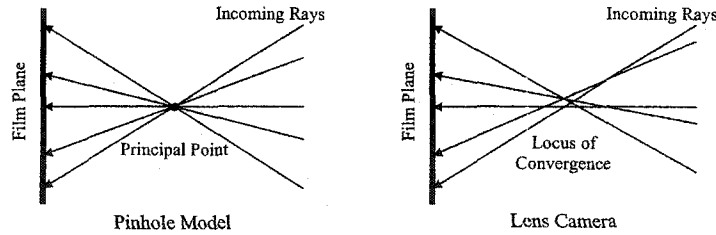


Fig. 3.3: Lens camera deviates from the pinhole model in locus of convergence.

3.1.2 Camera Parameters and Camera Calibration

Camera parameters are the parameters linking the coordinates of points in 3-D space with the coordinates of their corresponding image points. In particular, the extrinsic parameters are the parameters that define the location and orientation of the camera reference frame with respect to the world reference frame. The intrinsic parameters are the parameters necessary to link the pixel coordinates of an image point with the corresponding coordinates in the camera frame [41].

Extrinsic Parameters

The extrinsic parameters are defined as any set of geometric parameters that uniquely define the transformation between the world reference frame and the camera frame. A typical choice for describing the transformation is to use a 3×1 vector \mathbf{t} and a 3×3 orthogonal rotation matrix R such that $P^c = RP^w + \mathbf{t}$. According to Euler's rotation theorem, an arbitrary rotation can be described by only three parameters. As a result, the rotation matrix R has three degree-of-freedom (DOF) and the extrinsic parameters totally

have six DOF. Given a rotation matrix R

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix},$$

one method to get the three parameters that uniquely describe this matrix is to extract ZYZ Euler angles $(\theta_a, \theta_b, \theta_c)$, such that the rotation matrix R can be calculated by [43]:

$$R = R_z(\theta_a) R_y(\theta_b) R_z(\theta_c), \quad (3.2)$$

with

$$R_z(\theta_c) = \begin{bmatrix} \cos(\theta_c) & -\sin(\theta_c) & 0 \\ \sin(\theta_c) & \cos(\theta_c) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R_y(\theta_b) = \begin{bmatrix} \cos(\theta_b) & 0 & \sin(\theta_b) \\ 0 & 1 & 0 \\ -\sin(\theta_b) & 0 & \cos(\theta_b) \end{bmatrix}. \quad (3.3)$$

When $\sin(\theta_b) \neq 0$, the solutions for $(\theta_a, \theta_b, \theta_c)$ are [43]:

$$\begin{aligned} \theta_b &= \arctan 2(\sqrt{r_{31}^2 + r_{32}^2}, r_{33}), \\ \theta_a &= \arctan 2(r_{23}/\sin(\theta_b), r_{13}/\sin(\theta_b)), \\ \theta_c &= \arctan 2(r_{32}/\sin(\theta_b), -r_{31}/\sin(\theta_b)). \end{aligned} \quad (3.4)$$

Intrinsic Parameters

The intrinsic parameters are as follows:

- 1) The focal length f .
- 2) The parameters defining the transformation between the camera frame and the image plane. Neglecting any geometric distortion and with the assumption that the CCD array is made of a rectangular grid of photosensitive elements, we have:

$$\begin{aligned} x &= -(u - u_0) s_x, \\ y &= -(v - v_0) s_y, \end{aligned} \quad (3.5)$$

with (u_0, v_0) the coordinates in pixel of the image center and s_x, s_y the effective sizes of the pixel in the horizontal and vertical direction, respectively. Let $f_x = f/s_x$ and $f_y = f/s_y$. The current set of intrinsic parameters are u_0, v_0, f_x , and f_y .

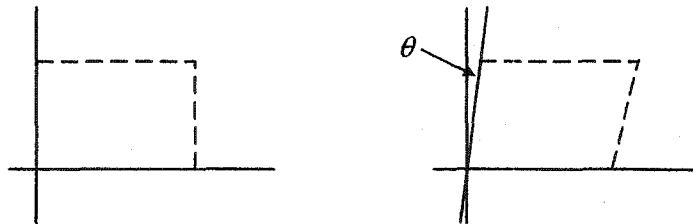


Fig. 3.4: Skewness of two image axes.

- 3) The parameter describing the skewness of the two image axes: $\gamma = f_y \tan \theta$. The skewness of two image axes is illustrated in fig. 3.4.
- 4) The parameters characterizing the lens distortion.

Projection Matrix

With the homogeneous transform and the camera parameters, we can have a 3×4 matrix, called the *projection matrix*, that directly links a point in the 3-D world reference frame to its projection in the image plane. That is [12, 41]:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A_{\text{intr}} [R | \mathbf{t}] \begin{bmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X^c \\ Y^c \\ Z^c \end{bmatrix}, \quad (3.6)$$

where (u, v) is the distortion-free image point on the image plane; the matrix A_{intr} , called the intrinsic matrix, fully depends on the camera's five intrinsic parameters $(\alpha, \gamma, \beta, u_0, v_0)$, with (α, β) being two scalars in the two image axes, (u_0, v_0) the coordinates of the principal point, and γ describing the skewness of the two image axes; $[X^c, Y^c, Z^c]^T$ denotes a point in the camera frame which is related to the corresponding point $[X^w, Y^w, Z^w]^T$ in the world reference frame by $P^c = R P^w + \mathbf{t}$ with (R, \mathbf{t}) being the rotation matrix and the translation vector. For a variety of computer vision applications where a camera is used as a sensor in the system, the camera is always assumed fully calibrated beforehand. From equations (3.1) and (3.5), we have

$$u = -\frac{f}{s_x} \frac{X^c}{Z^c} + u_0. \quad (3.7)$$

From equation (3.6), we have

$$u = \alpha \frac{X^c}{Z^c} + u_0, \quad (3.8)$$

with scaling factor $\lambda = Z^c$. From the above two equations, we get $\alpha = -f/s_x = -f_x$. In the same manner, $\beta = -f_y$.

Camera calibration is the process of estimating the camera's extrinsic parameters, intrinsic parameters, and the distortion coefficients. The early works on precise camera calibration, starting in the photogrammetry community, use a 3-D calibration object whose geometry in the 3-D space was required to be known with a very good precision. However, since these approaches require an expensive calibration apparatus, camera calibration is prevented from being carried out broadly. Aiming at the general public, the camera calibration method proposed in [12] focuses on desktop vision system and uses 2-D metric information. The key feature of the calibration method in [12] is that it only requires the camera to observe a planar pattern at a few (at least three, if both the intrinsic and the extrinsic parameters are to be estimated uniquely) different orientations without knowing the motion of the camera or the calibration object. Due to the above flexibility, the calibration method in [12] is used in this dissertation, where the detailed procedures are summarized as: 1) estimation of intrinsic parameters, 2) estimation of extrinsic parameters, 3) estimation of distortion coefficients, and 4) nonlinear optimization. The overall calibration procedures are illustrated in fig. 3.5.

3.1.3 Radial Distortion

In equation (3.6), (u, v) is not the actually observed image point since virtually all imaging devices introduce a certain amount of nonlinear distortion. Among the nonlinear distortions, radial distortion, which occurs along the radial direction from the center of distortion, is the most severe part [9, 10]. The removal or alleviation of the radial distortion is commonly performed by first applying a parametric radial distortion model, estimating the distortion coefficients, and then correcting the distortion.

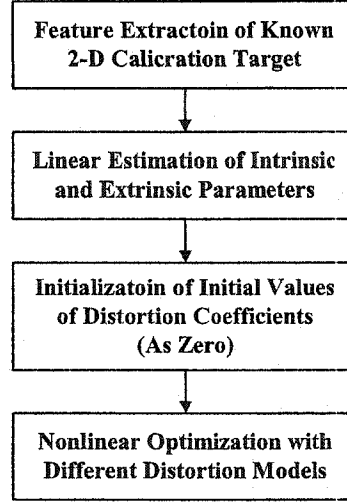


Fig. 3.5: Camera calibration procedures.

Lens distortion is very important for accurate 3-D measurement [8]. Let (u_d, v_d) be the actually observed image point and assume that the center of distortion is at the principal point. The relationship between the undistorted and the distorted radial distances is given by:

$$r_d = r + \delta_r, \quad (3.9)$$

where r_d is the distorted radial distance and δ_r the radial distortion.

Most of the existing works on the radial distortion models can be traced back to an early study in photogrammetry [11] where the radial distortion is suggested to be governed by the following polynomial equation [11, 12, 13, 14]:

$$r_d = r f(r, \mathbf{k}) = r (1 + k_1 r^2 + k_2 r^4 + k_3 r^6 + \dots), \quad (3.10)$$

where $\mathbf{k} = (k_1, k_2, k_3, \dots)$ are the distortion coefficients. It follows that

$$x_d = x f(r), \quad y_d = y f(r), \quad (3.11)$$

which is equivalent to

$$u_d - u_0 = (u - u_0) f(r), \quad v_d - v_0 = (v - v_0) f(r). \quad (3.12)$$

For the polynomial radial distortion model (3.10) and its variants, the distortion is especially dominated by the first term and it has also been found that too high an order may cause numerical instability [10, 12, 44]. When using two coefficients, the relationship between the distorted and the undistorted radial distances becomes [12]

$$r_d = r(1 + k_1 r^2 + k_2 r^4). \quad (3.13)$$

The inverse of the polynomial function (3.13) is difficult to perform analytically, but can be obtained numerically via an iterative scheme.

Besides the commonly used radial distortion modeling (3.10), the relationship between r_d and r can also be modelled as [45]

$$r_d = r f(r) = r(1 + k_1 r + k_2 r^2 + k_3 r^3 + \dots). \quad (3.14)$$

To overcome the inversion problem, the above model with two coefficients is studied in [29], which is

$$f(r) = 1 + k_1 r + k_2 r^2, \quad (3.15)$$

whose appealing feature lies in its satisfactory accuracy as well as the existence of an easy analytical undistortion formula.

Until recently, the most commonly used radial distortion models are still in the polynomial form, though other models, such as the division model [46] and the fish-eye radial distortion models (such as the fish eye transform [9]), are available in the literature. A rational model in the form of

$$r_d = \frac{\sqrt{1 + 4k}r^2 - 1}{2kr} \quad (3.16)$$

is proposed in [47] when r_d and r are expressed in the camera frame. However, it is not specified clearly in [47] why this model is sought.

In this dissertation, a new class of rational radial distortion models that are functions of simple polynomials is proposed. In the full-scale nonlinear optimization, the following objective function [12]:

$$J = \sum_{i=1}^{N_{\text{im}}} \sum_{j=1}^n \|m_{ij} - \hat{m}(A_{\text{intr}}, \mathbf{k}, R_i, \mathbf{t}_i, M_j)\|^2, \quad (3.17)$$

is used, where M_j is the j^{th} 3-D point in the world frame with $Z^w = 0$; $\hat{m}(A_{\text{intr}}, \mathbf{k}, R_i, \mathbf{t}_i, M_j)$ is the projection of point M_j in the i^{th} image using the estimated parameters; \mathbf{k} denotes the distortion coefficients; n is the number of feature points in the coplanar calibration object; and N_{im} ¹ is the number of images taken for calibration. In [12], the estimation of radial distortion is done after having estimated the intrinsic and the extrinsic parameters and just before the nonlinear optimization step. So, for different radial distortion models, we can reuse the estimated intrinsic and extrinsic parameters. The quantities M_j are known during the calibration process. Knowledge of M_j comes from the known calibration target, which is a 2-D planar surface in our study. The quantities R_i and \mathbf{t}_i are not fixed during the nonlinear optimization.

3.2 Feature Extraction

The calibration method illustrated here uses a planar calibration object shown in fig. 3.6, where 64 squares are separated evenly and the side of each square is 1.3 cm.

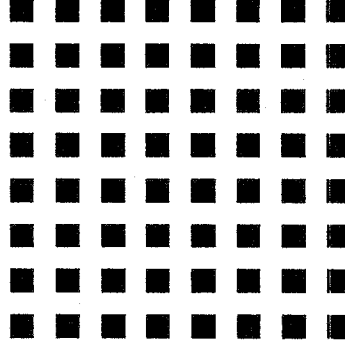


Fig. 3.6: Calibration object.

The procedures to extract the feature locations of the above calibration object are illustrated in table 3.1. The input image is an intensity image. After thresholding it with a certain value (which is 150 in our case), we can get a binary image. The binary image then goes through a Connected Component Labelling algorithm [48, 49] that outputs a *region*

¹ n and N_{im} are chosen to be $n = 256$ and $N_{\text{im}} = 5$ in all the experiments performed in this dissertation.

map, where each class of the connected pixels is given a unique label. For every class in the region map, we need to know whether or not it can be a square. In our approach, this is done by first detecting the edge of each class and then finding the number of partitions of the edge points. If the number of partitions is not equal to four, which means it is not a 4-sided polygon, we will bypass this class. Otherwise, we will fit a line using all the edge points that lie between each two adjacent partitions and thus get four line fits. The final output of this class is the intersections of these four line fits that approximate the four corners of each square. After running through all the classes in the region map, if the number of detected squares equals to the actual number of squares in the calibration object, we will record all the detected corners and arrange them in the same order as for the 3-D points in the 3-D space (for a given calibration object, assume $Z^w = 0$, we know the exact coordinates of the feature points in world reference frame and we need to arrange these feature points in a certain order so that after detecting feature points in the observed images, we can have an algorithm to seek the map from a point in the world frame to its corresponding projection in the image plane). After detecting five images, we are ready for the camera calculation.

Table 3.1: Procedures to Extract Feature Locations

Threshold input intensity image to make it binary (the threshold is 150)
Find connected components using 8-connectivity method, and output a region map
Loop for every class in the region map
Select the class whose area is < 3000 and > 20
Binary edge detection of this class
Find partitions of the detected edge points
If number of partitions = 4
Line fit between each two adjacent partitions
Output four line intersections
End if
End loop
If the total number of intersections = $4 \times$ number of squares in the calibration object
Arrange intersections in the same order as points in the world reference frame
End if

Binary Image Edge Detection (Boundary Finding)

A boundary point of an object in a binary image is a point whose 4-neighborhood or 8-neighborhood intersects the object and its complement. Boundaries for binary images are classified by their connectivity and by whether they lie within the object or its complement. The four classifications are: interior or exterior 8-boundaries and interior or exterior 4-boundaries [39]. In our approach, we use the interior 8-boundary operator, as shown in fig. 3.7, which is denoted by [39]:

$$\mathbf{b} = (1 - (\mathbf{a} \diamond N_{ei})) \mathbf{a}, \quad (3.18)$$

where

- 1) \mathbf{a} is the input binary image.
- 2) \mathbf{b} is the output boundary binary image.
- 3) N_{ei} is the 4-neighborhood: $N_{ei}(p(u, v)) = \{y : y = (u \pm j, v) \text{ or } y = (u, v \pm i), i, j \in \{0, 1\}\}$.
- 4) For each pixel $p(u, v)$ in \mathbf{a} , $p(u, v) \diamond N_{ei} = \text{minimum pixel value around } p(u, v) \text{ in the sense of 4-neighborhood.}$

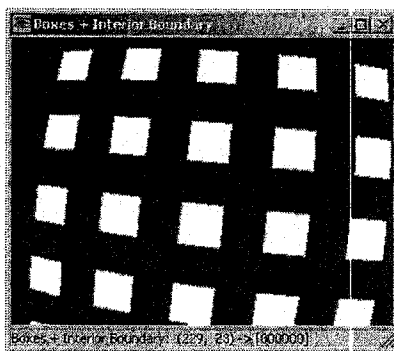


Fig. 3.7: Objects and their interior 8-boundary.

Partitions of Edge Points

Given a set of points that characterize the boundary of some object, a common question is what shape this object is, when we try to use polygons, especially the convex polygons, to denote objects in the real world. The set of points can be the output of some range finding sensors, such as laser and sonar. Or, it can come from images captured by a camera and is preprocessed by some edge detector, which is just the case we are discussing. In our problem, we know beforehand that the region of interest is a square and we can use the scan line approximation method [50, 51] to find the partitions. The scan line approximation algorithm is described in table 3.2. Figure 3.8 shows an illustration. Figure 3.9 shows the fitting results using the partitions found by the scan line approximation algorithm, where all the input data are the edge points of squares in a real image.

Table 3.2: Scan Line Approximation Algorithm

Problem Definition	
Assumption:	Object is described using a convex polygon.
Given:	A set of data points that have already been sorted in certain order.
Find:	Partitions.
Algorithm	
Scan Line Approximation (start index, end index, data points)	
Draw a line connecting the starting point and the ending point	
Calculate the maximum distance each point \in [start index, end index] to this line	
If the maximum distance is greater than a predefined threshold	
Record the index of the point that gives the maximum distance	
Set end index = the index of that point that gives the maximum distance	
Scan Line Approximation (start index, end index, data points)	
Set start index = the index of that point that gives the maximum distance	
Scan Line Approximation (start index, end index, data points)	
End if	

In table 3.2, the algorithm is implemented in a recursive way. When applying this algorithm, an important issue is how to decide the threshold. Unfortunately, this threshold is application-dependent. In our implementation, we choose 5 ~ 10 pixels. The smaller the squares or the farther that the camera is away from the calibration object, the smaller the threshold should be. Another issue to which we need to pay attention is how to choose the

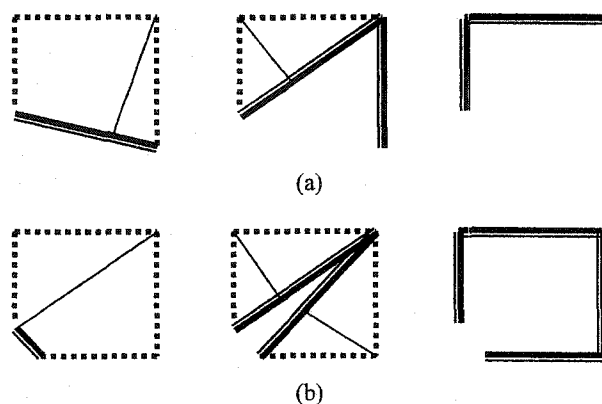


Fig. 3.8: Illustration of scan line approximation algorithm (a) 3 sides (b) 4 sides.

initial starting and ending points. They cannot be on the same side. Otherwise, due to the noise in the data, the point whose distance to the line connecting the starting and ending points is maximal might not be around the corners. This problem can be solved simply by first finding the two adjacent points whose maximal distance that all other points to this line is the biggest.

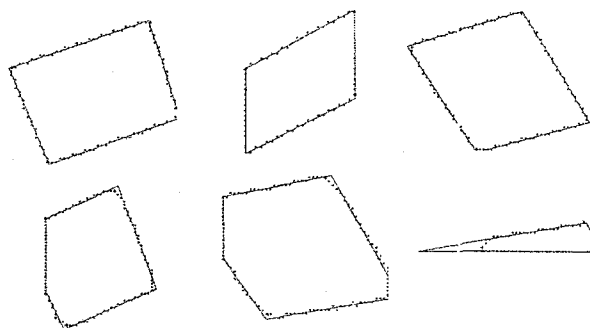


Fig. 3.9: Fitting results using partitions found by scan line approximation.

Figures 3.10 and 3.11 show two sets of processed images at all steps, where the input images are captured by a desktop camera. Notice that in fig. 3.11, in the image titled with “Binary Image + Partition Points,” the triangle in the upper right corner does not show in the next step. The reason why this happens is that after finding the partitions, the number of partitions does not equal to four and we thus bypass this region.

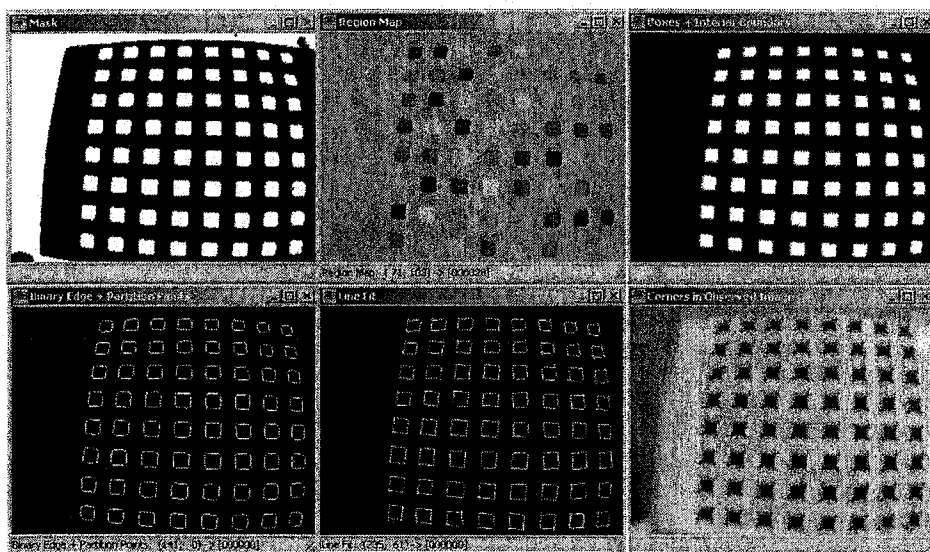


Fig. 3.10: Feature points extraction for desktop images (1).

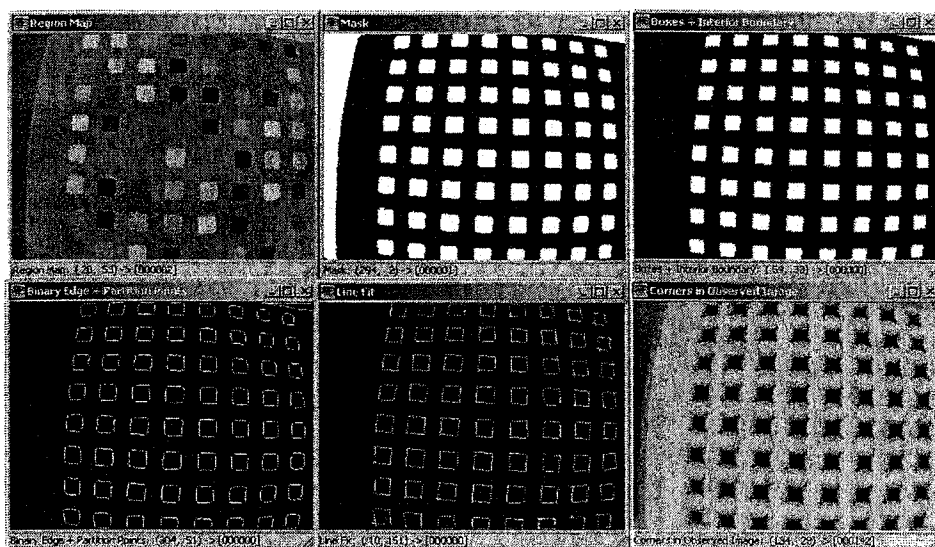


Fig. 3.11: Feature points extraction for desktop images (2).

3.3 Analytical Polynomial Radial Distortion Model

Sections 3.1 and 3.2 have described the general background of camera calibration and the preprocessing of the images for the camera calibration. In the following sections, we shall focus on lens distortion modeling, with an emphasis on the analytical inverse formula.

The conventional radial distortion model (3.13) does not have an exact inverse. Existing methods to overcome the problem of no analytical inverse function include:

- 1) Using iterative numerical methods.
- 2) Preparing a table containing the inverse relationship and searching through the table for the inverse.
- 3) Using approximations, such as [52]

$$r_d = r(1 + k_1 r^2 + k_2 r^4) \leftrightarrow r = r_d(1 - k_1 r_d^2 - k_2 r_d^4) \quad (3.19)$$

and [46]

$$r_d = r(1 + k_1 r^2) \leftrightarrow r = \frac{r_d}{1 + k_1 r_d^2}. \quad (3.20)$$

The fitting results given by the above approximations can be satisfactory when the distortion coefficients are small values. However, an extra source of error is introduced and this will inevitably degrade the calibration accuracy.

The above listed three methods introduce either additional computation time or additional approximation errors. Due to these reasons, an analytical inverse function that has the advantage of giving the exact undistortion solution is one of the main focus of this work. To overcome the shortcoming of no analytical inverse formula, but still preserving a comparable accuracy, the radial distortion model (3.15) that has the following three properties has been studied in [29]:

- 1) This function is radially symmetric around the center of distortion (which is assumed to be at the principal point (u_0, v_0) for our discussion) and it is expressible in terms of radius r only.
- 2) This function is continuous and derivable, hence $r_d = 0$ iff $r = 0$ within the operating range of r .²

²For a higher order polynomial in the form of $r_d = rf(r)$, it is obvious that $r = 0 \Rightarrow r_d = 0$. However, without a proper hypothesis, there might exist other solutions besides $r = 0$ that satisfy $r_d = 0$. To model the physical lens distortion, the condition that $r_d = 0 \Rightarrow r = 0$ has to be ensured within the operating range of r .

- 3) The resultant approximation of x_d is an odd function of x . For example, from (3.15), we have

$$\begin{cases} x_d = x f(r) = x(1 + k_1 r + k_2 r^2) \\ y_d = y f(r) = y(1 + k_1 r + k_2 r^2) \end{cases} \quad (3.21)$$

It is obvious that $x_d = 0$ iff $x = 0$. When $x_d \neq 0$, by letting $\kappa = y_d/x_d = y/x$, we have $y = \kappa x$ where κ is a constant. Substituting $y = \kappa x$ into the above equation gives:

$$\begin{aligned} x_d &= x \left[1 + k_1 \sqrt{x^2 + \kappa^2 x^2} + k_2 (x^2 + \kappa^2 x^2) \right] \\ &= x + k_1 \sqrt{1 + \kappa^2} \operatorname{sgn}(x) x^2 + k_2 (1 + \kappa^2) x^3, \end{aligned} \quad (3.22)$$

where $\operatorname{sgn}(x)$ gives the sign of x and x_d is an odd function of x . Thus, when interpreting from the relationship between (x_d, y_d) and (x, y) in the camera frame, the radial distortion function is to approximate the $x_d \leftrightarrow x$ relationship, which is intuitively an odd function.

Remark 3.3.1 *The radial distortion models discussed in this dissertation belong to the category of Undistorted-Distorted model, while the Distorted-Undistorted model also exists in the literature to correct the distortion [53]. The radial distortion models can be applied to the D-U formulation simply by defining*

$$r = r_d f(r_d).$$

Consistent results and improvement can be achieved in the above D-U formulation. While a D-U formulation relieves us the task of performing undistortion, in some cases, it is still desirable to locate the projection of a 3-D point on the image plane knowing its 3-D coordinates.

Analytical Radial Undistortion

From equations (3.14) and (3.15),

$$k_2 r^3 + k_1 r^2 + r - r_d = 0. \quad (3.23)$$

By letting $a = k_1/k_2$, $b = 1/k_2$, and $c = -r_d/k_2$, we have

$$r^3 + ar^2 + br + c = 0. \quad (3.24)$$

Further, let $\bar{r} = r + a/3$. The above equation becomes $\bar{r}^3 + p\bar{r} + q = 0$, where $p = b - a^2/3$ and $q = 2a^3/27 - ab/3 + c$. Let $\Delta = (\frac{q}{2})^2 + (\frac{p}{3})^3$. If $\Delta > 0$, there is only one solution; if $\Delta = 0$, then $r = 0$, which occurs when $\delta_r = 0$; if $\Delta < 0$, then there are three solutions [54]. In general, the middle one is what we need, since the first root is at a negative radius and the third lies beyond the positive turning point [55, 56]. After r is determined, (u, v) can be calculated from (3.12) directly.

3.4 Rational Radial Distortion Models

Based on the three criteria listed in section 3.3, a new class of radial distortion models (models #5, 6, 7, 8, 9, 10) is proposed and summarized in table 3.3, where the other four polynomial models (models #1, 2, 3, 4) are also listed. This class of six new models is our original contribution of this dissertation. Using these models, the $x_d \leftrightarrow x$ relationship complies with the property of being an odd function, as shown in the third column in table 3.3.

Clearly, all these functions in table 3.3, except the function #4, are special cases of the following function

$$f(r, \kappa) = \frac{1 + \kappa_1 r + \kappa_2 r^2}{1 + \kappa_3 r + \kappa_4 r^2 + \kappa_5 r^3}. \quad (3.25)$$

The function #4 in table 3.3 is the most commonly used radial distortion function in the polynomial approximation category. The other nine functions in table 3.3 are studied specifically with the goal to achieve comparable performance with the function #4 using the least amount of model complexity and as few distortion coefficients as possible. Since the functions #9, 10 in table 3.3 begin to show comparable calibration performance to the function #4 (as can be seen later in section 3.6), more complex distortion functions are not studied in this dissertation.

Table 3.3: Distortion Models

#	$f(r)^*$	$x_d = f(x)$
1	$1 + k r$	$x \cdot (1 + k \sqrt{1 + c^2} x \operatorname{sgn}(x))$
2	$1 + k r^2$	$x \cdot (1 + k (1 + c^2) x^2)$
3	$1 + k_1 r + k_2 r^2$	$x \cdot (1 + k_1 \sqrt{1 + c^2} x \operatorname{sgn}(x) + k_2 (1 + c^2) x^2)$
4	$1 + k_1 r^2 + k_2 r^4$	$x \cdot (1 + k_1 (1 + c^2) x^2 + k_2 (1 + c^2)^2 x^4)$
5	$\frac{1}{1 + k r}$	$x \cdot \frac{1}{1 + k \sqrt{1 + c^2} x \operatorname{sgn}(x)}$
6	$\frac{1}{1 + k r^2}$	$x \cdot \frac{1}{1 + k (1 + c^2) x^2}$
7	$\frac{1 + k_1 r}{1 + k_2 r^2}$	$x \cdot \frac{1 + k_1 \sqrt{1 + c^2} x \operatorname{sgn}(x)}{1 + k_2 (1 + c^2) x^2}$
8	$\frac{1}{1 + k_1 r + k_2 r^2}$	$x \cdot \frac{1}{1 + k_1 \sqrt{1 + c^2} x \operatorname{sgn}(x) + k_2 (1 + c^2) x^2}$
9	$\frac{1 + k_1 r}{1 + k_2 r + k_3 r^2}$	$x \cdot \frac{1 + k_1 \sqrt{1 + c^2} x \operatorname{sgn}(x)}{1 + k_2 \sqrt{1 + c^2} x \operatorname{sgn}(x) + k_3 (1 + c^2) x^2}$
10	$\frac{1 + k_1 r^2}{1 + k_2 r + k_3 r^2}$	$x \cdot \frac{1 + k_1 (1 + c^2) x^2}{1 + k_2 \sqrt{1 + c^2} x \operatorname{sgn}(x) + k_3 (1 + c^2) x^2}$

*With the risk of introducing some confusion, the same symbols k_1, k_2 , and k_3 are used in different models that might have different values.

Radial Undistortion of the Rational Models

The rational functions listed in table 3.3 are all special cases of the function (3.25). Radial undistortion of the proposed rational functions are illustrated in the next using the general function (3.25). From (3.25), we have

$$r_d = r f(r) = r \frac{1 + \kappa_1 r + \kappa_2 r^2}{1 + \kappa_3 r + \kappa_4 r^2 + \kappa_5 r^3}. \quad (3.26)$$

After simple mathematical derivations,

$$(r_d k_5 - k_2) r^3 + (r_d k_4 - k_1) r^2 + (r_d k_3 - 1) r + r_d = 0. \quad (3.27)$$

By letting

$$a = \frac{r_d k_4 - k_1}{r_d k_5 - k_2}, \quad b = \frac{r_d k_3 - 1}{r_d k_5 - k_2}, \quad c = \frac{r_d}{r_d k_5 - k_2},$$

we arrive at equation (3.24). Following the procedures described in section 3.3, radial undistortion of the general rational function (3.25) can be performed analytically with a closed-form solution.

3.5 Piecewise Radial Distortion Model

Besides the rational distortion functions as listed in table 3.3, a two-segment radial distortion function is proposed and illustrated in fig. 3.12, where each segment is a function of the form

$$\begin{cases} f_1(r) = a_0 + a_1 r + a_2 r^2, & \text{for } r \in [0, r_1] \\ f_2(r) = b_0 + b_1 r + b_2 r^2, & \text{for } r \in (r_1, r_2] \end{cases}, \quad (3.28)$$

with $r_1 = r_2/2$. We are interested in estimating the coefficients (a_0, a_1, a_2) and (b_0, b_1, b_2) such that the two polynomials are continuous and smooth at the interior point $r = r_1$. The reason for choosing a distortion function similar to (3.15) for each segment is that the radial undistortion can be performed using the analytical procedures described in section 3.3 with no iterations.

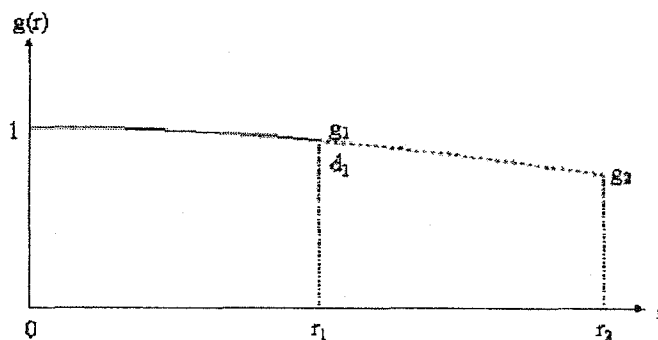


Fig. 3.12: A smooth piecewise function (two-segment).

To ensure that the overall function (3.28) is continuous and smooth across the interior point,³ the following six constraints can be applied:

$$\begin{cases} f_1(0) = 1 \\ a_0 + a_1 r_1 + a_2 r_1^2 = f_1 \\ a_1 + 2a_2 r_1 = d_1 \\ b_0 + b_1 r_1 + b_2 r_1^2 = f_1 \\ b_1 + 2b_2 r_1 = d_1 \\ b_0 + b_1 r_2 + b_2 r_2^2 = f_2 \end{cases}, \quad (3.29)$$

³Besides the requirement of continuity and smoothness at the interior point r_1 , the estimated $f(r)$ curve has to be monotonous to ensure the uniqueness in the $r_d \leftrightarrow r$ relationship in the context of lens distortion modeling. However, this additional constraint is not necessary since it is inherent in the physical lens to be modelled.

where $f_1 = f_1(r_1) = f_2(r_1)$, $f_2 = f_2(r_2)$, and $d_1 = \dot{f}_1(r_1) = \dot{f}_2(r_1)$. By enforcing that the two segments have the same value and derivative at the interior point r_1 , the resultant single function is guaranteed to be continuous and smooth over the whole range $[0, r_2]$. Since each interior point provides four constraints to make the resultant single function smooth, in order to estimate the coefficients (a_0, a_1, a_2) and (b_0, b_1, b_2) uniquely, we need another two constraints, which are chosen to be $f_1(0)$ and $f_2(r_2)$ in (3.29).

Since the coefficients (a_0, a_1, a_2) and (b_0, b_1, b_2) in (3.29) can be calculated uniquely from (f_1, d_1, f_2) by:

$$\begin{cases} a_0 = 1 \\ a_1 = (-2 - r_1 d_1 + 2f_1) / r_1 \\ a_2 = (1 + r_1 d_1 - f_1) / r_1^2 \\ b_2 = (f_2 - f_1 + r_1 d_1 - r_2 d_1) / (r_1 - r_2)^2 \\ b_1 = d_1 - 2b_2 r_1 \\ b_0 = f_1 - d_1 r_1 + b_2 r_1^2 \end{cases} \quad (3.30)$$

the radial distortion coefficients that are used in the nonlinear optimization for the piecewise radial distortion model can be chosen to be (f_1, d_1, f_2) with the initial values $(1, 0, 1)$ (equivalent to $k = 0$ for a single distortion function), which has only one extra coefficient compared with the single model (3.15). During the nonlinear optimization process, the coefficients (a_0, a_1, a_2) and (b_0, b_1, b_2) are calculated from (3.30) in each iteration.

3.6 Experimental Results

A series of experiments have been performed in this section to validate the proposed rational and piecewise distortion models. First, the final values of the objective function J defined in (3.17) of three groups of testing images are given in section 3.6.1. The model selection problem among the ten models in table 3.3 is further discussed using the geometric AIC (Akaike Information Criterion) and the geometric MDL (Minimum Description Length) criteria [57, 58] (section 3.6.2). Then, we simulate the whole imaging process by constructing a virtual camera with known camera parameters and radial distortion (section 3.6.3). We generate images with noise of a planar calibration target. At a second time, we test if the distortion coefficients are accurately estimated by observing the resultant $f(r) \leftrightarrow r$ curves.

3.6.1 Initial Model Comparison

For the proposed rational distortion functions (models #5, 6, 7, 8, 9, 10), comparisons are first made with the other four polynomial functions (models #1, 2, 3, 4) based on the final values of the objective function J in (3.17) after nonlinear optimization by the Matlab function `fminunc`, since the common approach to the camera calibration is to perform a full-scale nonlinear optimization for all parameters. Using the public domain testing images (640×480 in pixels) [59], the desktop camera images (320×240) [60] (a color camera in our CSOIS), and the ODIS camera images (320×240) [60] (the camera on ODIS robot built in our CSOIS), the final values of J , the estimated distortion coefficients, and the five estimated intrinsic parameters ($\alpha, \beta, \gamma, u_0, v_0$), are shown in Tables 3.4, 3.5, and 3.6, respectively. The extracted corners for the model plane of the desktop and the ODIS cameras are shown in figs. 3.13 and 3.14. As noticed from these images, the two cameras both experience a barrel distortion.⁴ In the experiments, we used the planar calibration target shown in fig. 3.6 images were taken for each calibration.

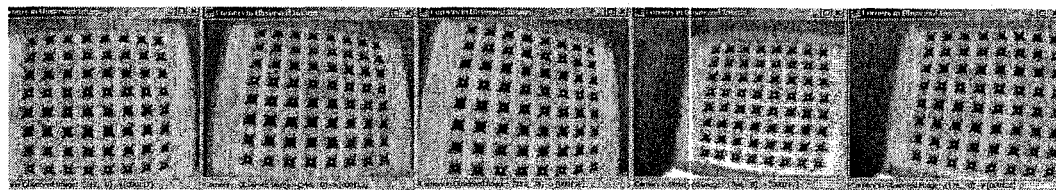


Fig. 3.13: Five images of the model plane with the extracted corners (indicated by cross) for the desktop camera.

Tables 3.4, 3.5, and 3.6 have the same format. The first column is the model number consistent with table 3.3. The second column shows the values of the objective function J defined in (3.17). The third column, the rank, sorts the distortion models by J in a [1-smallest, 10-largest] manner. From Tables 3.4, 3.5, and 3.6, we have the following observations:

⁴The plotted dots in the center of each square are only used for judging the correspondence with the world reference points.

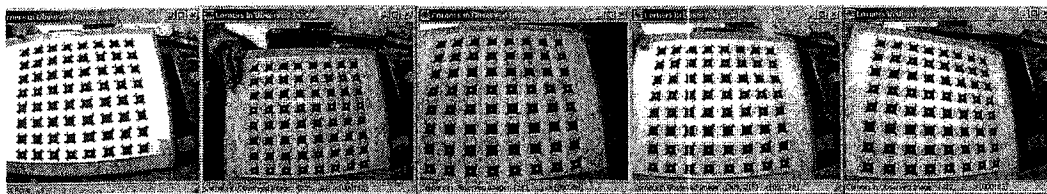


Fig. 3.14: Five images of the model plane with the extracted corners (indicated by cross) for the ODIS camera.

- 1) Using the proposed rational models, comparable, or even better, results can be achieved compared with the polynomial model #4 in table 3.3, at the cost of introducing an extra distortion coefficient.
- 2) For each category of models, either polynomial or rational, they generally follow the trend that the more complex the model, the smaller the fitting residual J .
- 3) When the distortion is significant, the performance improvement using complex models becomes more obvious.

The comparison based on J in Tables 3.4, 3.5, and 3.6 might not be fair since the distortion models use different numbers of distortion coefficients. Due to this concern, more simulations and discussions are presented in sections 3.6.2 and 3.6.3 for the validation of the proposed rational models regarding their accuracy improvement and stability. Our main point is to emphasize that, by applying the rational functions, comparable accuracy can be achieved without sacrificing the property of having an analytical undistortion function.

To make the results repeatable by other researchers for further investigation, we present the options we use for the nonlinear optimization:

```
options = optimset('Display', 'iter', 'LargeScale', 'off', 'MaxFunEvals',
8000, 'TolX', 10-5, 'TolFun', 10-5, 'MaxIter', 120).
```

3.6.2 Distortion Model Selection

Classical criteria that are used in computer vision to assess the accuracy of calibration includes the radial distortion as one part inherently [42]. However, the idea to chose among

Table 3.4: Comparison Results Using Microsoft Images

#	J	Rank	Final Values of k			Final Values of $(\alpha, \gamma, u_0, \beta, v_0)$				
1	180.5714	9	-0.0984	-	-	845.305	0.191	303.572	845.262	208.439
2	148.2789	8	-0.1984	-	-	830.742	0.216	303.948	830.798	206.557
3	145.6592	6	-0.0215	-0.1566	-	833.650	0.207	303.984	833.686	206.555
4	144.8802	3	-0.2286	0.1905	-	832.486	0.204	303.960	832.515	206.581
5	185.0628	10	0.1031	-	-	846.130	0.192	303.507	846.082	208.694
6	147.0000	7	0.2050	-	-	831.086	0.213	303.964	831.136	206.517
7	145.4682	5	-0.0174	0.1702	-	833.397	0.207	303.968	833.432	206.556
8	145.4504	4	0.0170	0.1725	-	833.384	0.206	303.971	833.419	206.544
9	144.8328	2	1.6457	1.6115	0.4054	830.941	0.204	303.957	830.970	206.583
10	144.8257	1	1.2790	-0.0119	1.5478	831.737	0.204	303.957	831.766	206.592

Table 3.5: Comparison Results Using Desktop Images

#	$J (\times 10^3)$	Rank	Final Values of k			Final Values of $(\alpha, \gamma, u_0, \beta, v_0)$				
1	1.0167	9	-0.2466	-	-	295.573	-0.819	156.610	288.876	119.852
2	0.9047	8	-0.2765	-	-	275.595	-0.666	158.201	269.230	121.525
3	0.8033	7	-0.1067	-0.1577	-	282.564	-0.619	154.491	275.901	120.092
4	0.7790	1	-0.3435	0.1232	-	277.144	-0.573	153.988	270.558	119.810
5	1.2018	10	0.3045	-	-	302.233	-1.023	160.560	295.676	120.744
6	0.7986	6	0.3252	-	-	276.252	-0.578	154.797	269.706	120.323
7	0.7876	5	-0.0485	0.2644	-	279.506	-0.588	154.173	272.882	119.956
8	0.7864	4	0.0424	0.2834	-	279.326	-0.587	154.116	272.704	119.921
9	0.7809	3	0.5868	0.5271	0.5302	275.831	-0.573	153.999	269.282	119.819
10	0.7800	2	0.2768	-0.0252	0.6778	276.450	-0.573	153.991	269.885	119.809

Table 3.6: Comparison Results Using ODIS Images

#	$J (\times 10^3)$	Rank	Final Values of k			Final Values of $(\alpha, \gamma, u_0, \beta, v_0)$				
1	0.9444	9	-0.2327	-	-	274.266	-0.115	140.362	268.307	114.391
2	0.9331	8	-0.2752	-	-	258.319	-0.516	137.215	252.685	115.930
3	0.8513	6	-0.1192	-0.1365	-	266.085	-0.367	139.919	260.313	113.241
4	0.8403	3	-0.3554	0.1633	-	260.765	-0.274	140.058	255.148	113.172
5	1.0366	10	0.2828	-	-	278.021	-0.028	139.594	271.927	116.299
6	0.8676	7	0.3190	-	-	259.494	-0.430	139.125	253.869	113.961
7	0.8450	5	-0.0815	0.2119	-	264.403	-0.350	140.052	258.680	113.144
8	0.8438	4	0.0725	0.2419	-	264.134	-0.342	140.109	258.420	113.112
9	0.8379	1	1.2859	1.1839	0.7187	259.288	-0.282	140.293	253.704	113.007
10	0.8383	2	0.4494	-0.0124	0.8540	260.937	-0.280	140.243	255.317	113.056

candidate models the one that gives the smallest residual does not work, because a model with more degrees of freedom might always be chosen since it is more likely to yield a smaller residual. To compare the distortion models fairly, the over-fit caused by more degrees of freedom in the distortion model needs to be compensated. Due to the above concern, a comparison that is solely based on the fitting residual of the full-scale nonlinear optimization, as performed in section 3.6.1, is not enough.

Model selection is one of the central subjects of statistical inference. In [57], two widely adopted criteria for statistical model selection, Akaike's AIC and Rissanen's MDL, have been generalized to be GAIC and GMDL for the geometric fitting such that the generalized GAIC and GMDL can be helpful for geometric problems considered in the computer vision. The GAIC and GMDL of a model S are defined as [57, 58]:

$$\text{GAIC}(S) = J(S) + 2(dD + p)\epsilon^2, \quad (3.31)$$

and

$$\text{GMDL}(S) = J(S) - (dD + p)\epsilon^2 \log(\epsilon/L)^2, \quad (3.32)$$

where $J(S)$ is the fitting residual when data of size D are fitted to the model S . p is the degree of freedom (DOF) of the model S . $d = m - s$ with m the dimension of the observed data and s the co-dimension of the model. L is a reference length, which is taken to be the image width in [58]. ϵ is the noise level in the data set.

In the context of lens distortion modeling, in order to apply the GAIC and GMDL, the noise level ϵ needs to be known. An unbiased estimate of ϵ can be obtained from the most commonly used candidate model (3.13), denoted by S^0 , as [57, 58]:

$$\hat{\epsilon}^2 = \frac{J(S^0)}{sD - p^0}, \quad (3.33)$$

where p^0 is the DOF of the model S^0 .

Naturally, interests arose about using the MDL criterion, since it is regarded by many as superior to the AIC criterion for statistical inference. It was thus anticipated that a criterion like MDL would outperform the geometric AIC for geometric fitting, too [57].

However, it has been reported in [57] that the above anticipation is negative. In this experiment, both the GAIC and GMDL are applied to validate the proposed model relevance, with results shown in table 3.7. The first column “#” denotes the model number consistent with table 3.3. The quantities under “GAIC” and “GMDL” are the calculated values using (3.31) and (3.32), respectively, where an unbiased estimate of ϵ is obtained from the most commonly used candidate model (3.13). To calculate the GMDL, the reference length L is chosen to be the width of the image [58]. That is, L is chosen to be 640, 320, and 320 for the public, the desktop, and the ODIS images. To facilitate an easy comparison, the rank that sorts the ten models by the GAIC and GMDL criteria in a [1-smallest, 10-largest] manner are provided as subscripts associated with each quantity.

Table 3.7: Model Selection Using the GAIC and GMDL Criteria

#	Public ($L = 640$)		Desktop ($L = 320$)		ODIS ($L = 320$)	
	GAIC	GMDL(10^3)	GAIC(10^3)	GMDL(10^4)	GAIC(10^3)	GMDL(10^4)
1	471.0120 ₉	2.3734 ₉	2.5784 ₉	1.0411 ₉	2.6289 ₉	1.1014 ₉
2	438.7195 ₈	2.3411 ₈	2.4663 ₈	1.0299 ₈	2.6176 ₈	1.1003 ₈
3	436.3266 ₆	2.3402 ₅	2.3661 ₇	1.0205 ₇	2.5371 ₆	1.0929 ₆
4	435.5476 ₁	2.3394 ₁	2.3418 ₁	1.0181 ₁	2.5261 ₃	1.0918 ₁
5	475.5034 ₁₀	2.3779 ₁₀	2.7634 ₁₀	1.0596 ₁₀	2.7211 ₁₀	1.1106 ₁₀
6	437.4406 ₇	2.3398 ₂	2.3602 ₆	1.0193 ₆	2.5521 ₇	1.0937 ₇
7	436.1356 ₅	2.3400 ₄	2.3504 ₅	1.0189 ₄	2.5308 ₅	1.0923 ₃
8	436.1178 ₄	2.3400 ₃	2.3492 ₄	1.0188 ₂	2.5296 ₄	1.0921 ₂
9	435.7269 ₃	2.3411 ₇	2.3450 ₃	1.0190 ₅	2.5250 ₁	1.0923 ₄
10	435.7198 ₂	2.3411 ₆	2.3441 ₂	1.0189 ₃	2.5254 ₂	1.0924 ₅

Table 3.7 deserves careful observation. For the three groups of testing images, the GAIC criterion generally favors the models #4, 7, 8, 9, 10 (though not necessarily in the same order), more than the other five models. When using the GMDL criterion, other than selecting models #7, 8, 9, 10 for the desktop and the ODIS camera, for the public image, it favors the models #3 and 6. In fact, this phenomenon is not surprising, since when the distortion is not significant, it is natural that the advantage gained by using more complex models can not be paid off. Further, the penalty for one degree of freedom is heavier in the

GMDL than in the GAIC (see (3.31) and (3.32)). Thus, the GAIC is more faithful to the data than the GMDL, which is more likely to choose a degenerate model [57]. In summary, the proposed rational functions, at least the functions #7, 8, 9, 10, provide comparable alternatives to the commonly used model #4 concerning the calibration accuracy and the property of having analytical inverse formulae. Notice that, in the comparison performed in table 3.7, the model #4 is treated as the ground truth model.

3.6.3 Evaluation of Distortion Only

The GAIC and GMDL based model selection, as presented in section 3.6.2, uses the fitting residual after the nonlinear optimization process with the extra DOF in the distortion model being compensated. In this section, we want to imagine the measurements that involve the distortion only. This is based on the idea that if the distortion coefficients are accurately estimated, the resultant $f(r) \leftrightarrow r$ curves should be close to the true curve. Since the true values of the intrinsic parameters and the distortion coefficients are by no means exactly known from a manufactured camera as used in sections 3.6.1 and 3.6.2, we construct a virtual camera via simulation to evaluate the distortion calibration. Besides, we also consider how the initial values of the intrinsic and extrinsic parameters affect the final selection of the distortion model.

When constructing the virtual camera, we assume that the camera has the following parameters:

- 1) Intrinsic matrix:

$$A_{\text{intr}} = \begin{bmatrix} 260 & -0.2741 & 140.0581 \\ 0 & 255.1489 & 113.1727 \\ 0 & 0 & 1 \end{bmatrix}. \quad (3.34)$$

- 2) Distortion model: $f(r) = 1 - 0.3554r^2 + 0.1633r^4$.

3) Extrinsic matrix:

$$RT = \begin{bmatrix} -1.89 & 2.83 & -1.95 & 12.16 & -12.64 & -31.90 \\ -1.26 & 2.94 & -1.29 & 10.39 & -16.31 & -33.44 \\ -1.51 & 2.78 & -1.53 & 13.03 & -12.09 & -24.50 \\ -1.46 & 2.81 & -1.47 & 12.88 & -13.07 & -27.26 \\ -0.80 & 2.59 & -0.74 & 11.16 & -11.69 & -23.99 \end{bmatrix}, \quad (3.35)$$

where each row in (3.35) denotes the transformation between the camera and the world coordinate system, and where the first three elements in each row in RT denote the ZYZ Euler angles (see equations (3.2 - 3.4)). The remaining three elements denote the translational vector. The matrix RT has five rows to simulate the usage of five images for the camera calibration. The distortion model selected in the form of (3.13) is due to its common usage and acceptance. The simulated camera is close to the ODIS camera shown in table 3.6.

First, consider the influence of different initial values for the final distortion model selection. While different methods using the same data set might give different initial values, what we simulate here uses the same method to obtain the initial values, but with corrupted simulated images. The simulated calibration process is performed when noise levels of 0, 0.25, 0.5, and 0.75⁵ are applied to the ideal simulated (u_d, v_d) that is calculated using the assumed camera parameters in (3.34) and (3.35). For each noise level, five sets of corrupted images are generated, where each set contains five images for the calibration. We average the five sets of calibration results and use the average for comparison.

Since we are mainly interested in how the initial values influence the final selection of the distortion model, detailed values of $J, (\alpha, \gamma, u_0, \beta, v_0)$, and \mathbf{k} are not given. Only the model numbers in a sorted manner are presented in table 3.8, under the above four different noise levels using the GAIC and the GMDL criteria, where the GMDL quantities are written in an Italian format.

⁵The noise is generated using the Matlab command `randn(.)` multiplied with the corresponding noise level.

Quantities in table 3.8 show a similar trend as already shown in table 3.7, in the sense that the GAIC criterion shows a consistent selection of the rational models #7,8,9,10 among the top five models (not necessarily in the same order). However, when using the GMDL criterion, as the noise level increases, that is, as the initial values deviate from their “true values,” the GMDL criterion chooses models #3,6 over #9,10 to be among the top five. Again, this is due to the tendency in the GMDL to select a degenerate model of less coefficients, model #6 (one coefficient) over #4 (two coefficients) and model #3 (two coefficients) over models #9,10 (three coefficients).

Table 3.8: Model Selection of the Simulated Camera

Noise Level	Model #									
	←	Better		Worse				→		
0	4	10	9	8	7	6	3	2	1	5
	4	10	9	8	7	6	3	2	1	5
0.25	4	10	9	8	7	3	6	2	1	5
	4	8	7	10	9	6	3	2	1	5
0.50	4	10	9	8	7	3	6	2	1	5
	6	4	8	7	3	10	9	2	1	5
0.75	4	8	7	9	10	3	6	2	1	5
	6	4	8	7	3	10	9	2	1	5

*The rows written in the Italian format are the GMDL values. Others are the GAIC values.

It is interesting to see from table 3.8 that as the noise level increases, the GMDL criterion would not even choose the model #4, though this is the assumed “true” model with noise corruption. A possible explanation of this phenomenon is that, when the feature extraction could not guarantee certain precision, the camera parameters can not be estimated accurately.

The resultant $f(r) \leftrightarrow r$ curves are illustrated in fig. 3.15 under noise level 0.25 (Fig 3.15 (a)) and 0.75 (Fig 3.15 (b)), respectively. In each figure, the $f(r) \leftrightarrow r$ curves of models #3,4,6,7,8,9,10 are plotted. It can be observed that the plotted curves are very close to each other and also to the true curve. In fig. 3.15, each curve is not specifically labelled since

the main purpose here is to demonstrate the stability property of using different distortion models, rather than selecting one specific model qualitatively from these $f(r) \leftrightarrow r$ curves.

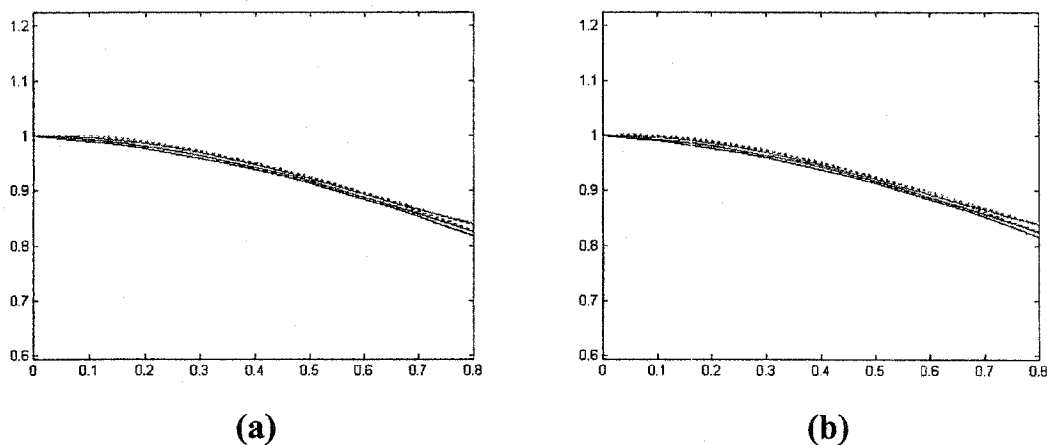


Fig. 3.15: $f(r) \leftrightarrow r$ curves for the simulated camera using the models #3, 4, 6, 7, 8, 9, 10 in table 3.3.

Until this point, it is safe to claim that the proposed class of rational functions provide an alternative for the commonly used rational distortion model (3.13), as far as both the calibration accuracy and the optimization stability are concerned. It is also worth mentioning that whether a certain distortion model best represents a lens distortion is indeed camera-dependent. Thus, it is not surprising that for some cameras, the proposed rational functions might outperform their polynomial counterparts.

3.6.4 Regarding Piecewise Model

Experiments similar to those performed in sections 3.6.1, 3.6.2, and 3.6.3 were also conducted for the piecewise model (3.28), where the values of the objective function J , along with the GAIC and the GMDL values, are shown in Tables 3.9 and 3.10, respectively. Comparing the entries in Tables 3.9 with the corresponding elements in Tables 3.4 ~ 3.6

and table 3.10 with table 3.7, focusing on models #9, 10⁶, we can observe that the piecewise model has a similar performance with the models #9, 10 in table 3.3.

Table 3.9: Objective Function J of the Piecewise Model

Images	J	Distortion Coefficients			Intrinsic Parameters $(\alpha, \gamma, u_0, \beta, v_0)$				
Public	144.8874	0.9908	-0.0936	0.9653	831.7068	0.2047	303.9738	831.7362	206.5670
Desktop	782.5865	0.9387	-0.2695	0.8066	277.4852	-0.5757	154.0058	270.9052	119.7416
ODIS	838.5678	0.9410	-0.2563	0.8270	261.9485	-0.2875	140.2521	256.3134	113.0856

Table 3.10: GAIC and GMDL Quantities of the Piecewise Model

Model	Public ($L = 640$)		Desktop ($L = 320$)		ODIS ($L = 320$)	
	GAIC	GMDL(10^3)	GAIC(10^3)	GMDL(10^4)	GAIC(10^3)	GMDL(10^4)
Piecewise	435.7814	2.3411	2.3466	1.0192	2.5257	1.0924

3.6.5 Discussions

This section presents some discussions regarding the advantage of the proposed piecewise and rational radial distortion models and how to extend these ideas to include tangential distortion.

- 1) **Why new models?** The traditional model (3.13) has been widely used in the radial distortion modeling due to its accuracy and its physical basis. Though it is also widely admitted that this model has no analytical undistortion formula, the disadvantage has not been emphasized since people tend to think that this problem can be solvable by the three methods listed at the beginning of section 3.3. The main purpose of this work is to study alternative distortion functions, either polynomials or simple functions of the polynomials, with least amount of model complexity and as few distortion coefficients as possible, to achieve a comparable (or even better) calibration accuracy compared with the most commonly used radial distortion model

⁶The models #9, 10 in table 3.3 use the same number of distortion coefficients as the piecewise model (three coefficients).

(3.13), with an additional appealing property of having analytical closed-form inverse formula. This additional appealing property might be achieved at the cost of adding one extra distortion coefficient.

- 2) **How to extend to include tangential distortion?** Though the distortion discussed is mainly focused on radial distortion modeling, it is worthy pointing out that the class of rational functions can also be applied to model geometric lens distortions by:

$$x_d = x f(r, \mathbf{k}_1), \quad y_d = y f(r, \mathbf{k}_2), \quad (3.36)$$

where $f(r, \mathbf{k})$ can be any function in table 3.3. Notice that when $\mathbf{k}_1 = \mathbf{k}_2$, the above equation reduces to the radial distortion modeling. The geometric distortion modeling in (3.36) allows the use of two different sets of distortion coefficients to model the distortions along the two image axes. Equation (3.36) denotes a lumped distortion model that aims to include all the nonlinear distortion effects.

Using quantities referring to the image plane (in pixels), lens distortion on the image plane can be modelled by: [61]

$$\begin{aligned} (u_d - u_0) &= (u - u_0) f(r_{uv}, \mathbf{k}_{uv1}), \\ (v_d - v_0) &= (v - v_0) f(r_{uv}, \mathbf{k}_{uv2}), \end{aligned} \quad (3.37)$$

where $f(r_{uv}, \mathbf{k}_{uv1,2})$ can be chosen to be any of the available distortion functions. Blind detection of lens simplified geometric distortions are presented in section 3.7 where $f(r_{uv}, \mathbf{k}_{uv1})$ and $f(r_{uv}, \mathbf{k}_{uv2})$ are chosen to be the following rational functions [27]:

$$\begin{aligned} f(r_{uv}, k_{uv1}) &= \frac{1}{1 + k_{uv1} r_{uv}^2}, \\ f(r_{uv}, k_{uv2}) &= \frac{1}{1 + k_{uv2} r_{uv}^2}, \end{aligned} \quad (3.38)$$

for its fewer number of distortion coefficients and the property of having analytical geometric undistortion formulae. From equations (3.37) and (3.38), we have

$$(u_d - u_0)^2 (1 + k_{uv1} \bar{r}_{uv})^2 + (v_d - v_0)^2 (1 + k_{uv2} \bar{r}_{uv})^2 = \bar{r}_{uv}, \quad (3.39)$$

with $\bar{r}_{uv} \triangleq r_{uv}^2$. The above equation is a quadratic function in \bar{r}_{uv} , thus having analytical inverse formula.

We have proposed a new class of rational radial distortion models and a two-segment piecewise polynomial distortion model. The appealing part of these distortion models is that they preserve high accuracy together with easy analytical undistortion formulae. Performance comparisons are made between this class of new rational models and the existing polynomial models, based on an initial observation of the fitting residue, the GAIC and GMDL criteria, and evaluation of the distortion only. Experiments results were presented to show that this new class of rational distortion models can be quite accurate and efficient, especially when the actual distortion is significant.

3.7 Blind Detection of Camera Lens Geometric Distortions

The existing camera calibration and lens distortion compensation techniques require either explicit calibration target, whose 2-D or 3-D metric information is available [12], or an environment rich in straight lines [9]. The above mentioned techniques are suitable for situations where the camera is available. For situations where direct access to the imaging device is not available, such as when down loading images from the web, a so-called blind lens removal technique has been exploited based on frequency domain criterion [15]. The fundamental analysis is based on the fact that lens distortion introduces higher-order correlations in the frequency domain, where the correlations can be detected via tools from Higher-Order Spectral Analysis (HOSA). However, it has been reported that the accuracy of blindly estimated lens distortion is by no means comparable to those based on calibration targets. Due to this reason, this approach can be useful in areas where only qualitative results are required [15].

3.7.1 Frequency Domain Blind Lens Detection

In this section, higher-order spectral analysis is first reviewed, which provides the fundamental criterion for the blind lens distortion removal technique [15, 62]. The basic approach of the blind lens distortion removal exploits the fact that lens distortion introduces higher-order correlations in the frequency domain, which can be detected using HOSA tools.

3.7.1.1 Bispectral Analysis

Higher-order correlations introduced by nonlinearities can be estimated by higher-order spectra [62]. For example, third-order correlations can be estimated by bispectrum, which is defined as

$$B(\omega_1, \omega_2) = E\{F(\omega_1)F(\omega_2)F^*(\omega_1 + \omega_2)\}, \quad (3.40)$$

where $E\{\cdot\}$ is the expected value operator and $F(\omega)$ is the Fourier transform of a stochastic one-dimensional signal in the form of

$$F(\omega) = \sum_{k=-\infty}^{\infty} f(k)e^{-j\omega k}. \quad (3.41)$$

F^* denotes the usual complex conjugate transpose. Notice that the bispectrum of a real signal is complex-valued. Since the estimate of the above bispectrum has the undesired property that its variance at each bi-frequency (ω_1, ω_2) is dependent of the bi-frequency, a normalized bispectrum, called the bicoherence, is exploited, which is defined to be [15, 62]

$$b^2(\omega_1, \omega_2) = \frac{|B^2(\omega_1, \omega_2)|}{E\{|F(\omega_1)F(\omega_2)|^2\} E\{|F(\omega_1 + \omega_2)|^2\}}. \quad (3.42)$$

The above bicoherence can be estimated as

$$\hat{b}(\omega_1, \omega_2) = \frac{\frac{1}{L} \sum_k F_k(\omega_1) F_k(\omega_2) F_k^*(\omega_1 + \omega_2)}{\sqrt{\frac{1}{L} \sum_k |F_k(\omega_1) F_k(\omega_2)|^2 \frac{1}{L} \sum_k |F_k(\omega_1 + \omega_2)|^2}}, \quad (3.43)$$

which becomes a real-valued quantity. As a measure of the overall correlations, the following quantity is employed in [15]

$$\frac{1}{L^2} \sum_{\omega_1=-L/2}^{L/2} \sum_{\omega_2=-L/2}^{L/2} \hat{b}\left(\frac{2\pi\omega_1}{L}, \frac{2\pi\omega_2}{L}\right), \quad (3.44)$$

where L is the dimension of the input one-dimensional signal.

3.7.1.2 Blind Lens Removal Algorithm

Consider a signal $f_d(x)$ that is a distorted version of $f(x)$ according to

$$f_d(x) = f(x(1 + \kappa x^2)), \quad (3.45)$$

with κ controlling the amount of distortion. It has been shown in [15] that correlation introduced by the nonlinearity is proportional to the distortion coefficient κ , where the quantity (3.44) is chosen as the measure of the correlation. Now, consider the inverse problem of recovering $f(x)$ from $f_d(x)$. It is only when κ is properly estimated that the inverted $\hat{f}(x)$ contains a least amount of nonlinearities, in which case $\hat{f}(x)$ holds a minimum bicoherence.

Based on the above discussion, an intuitive algorithm applied for the blind lens distortion removal is listed in the following [15]:

- 1) Select a range of possible values for the distortion coefficients κ .
- 2) For each κ , perform inverse undistortion to $f_d(x)$ yielding a provisional undistortion function $f_\kappa(x)$.
- 3) Compute the bicoherence of $f_\kappa(x)$.
- 4) Select the κ that minimizes all the calculated bicoherence of the undistorted signals.
- 5) Remove the distortion using the distortion coefficient obtained from step 4).

3.7.2 Experimental Results

In this section, we first verify that the blind lens removal technique, which is based on the detection of higher-order correlations in the frequency domain, can be used for the detection and compensation for lens distortion. This verification is via the comparison of the calibration coefficients of the blind removal technique with those calibrated by a planar-target based calibration method [12]. Though, the calibration results by the blind removal technique are by no means comparable to those based on a calibration target, the results shown in section 3.7.2.1 have reasonable accuracy, at least for applications where only qualitative performance is required.

The existing blind lens distortion removal method only considered a single-coefficient radial distortion model, as described in (3.46). In section 3.7.2.2, we show that cameras, which are more accurately modelled by different distortion coefficients along the two image

axes, can also be detected using higher-order correlations. As an example, the simplified geometric distortion modeling with the distortion function (3.38) is applied. Using a single-coefficient to describe the distortion along each image axis, totally two coefficients are used in (3.38). One reason for choosing the function (3.38) is for its fewer number of distortion coefficients that reduces the whole optimization duration. Another advantage is in that equation (3.38) has analytical geometric inverse formula, which further advances the optimization speed.

3.7.2.1 Verification of Blind Lens Compensation Using Calibrated Cameras

Comparisons between the distortion coefficients calibrated by the blind removal technique with those obtained by the target-based calibration method have been performed in [15] using the distortion function #2 in table 3.3, with distortion coefficient in the image plane. More specifically, the distortion is

$$r_{\text{duv}} = r_{\text{uv}}(1 + k_{\text{uv}} r_{\text{uv}}^2), \quad (3.46)$$

where r_{uv} and k_{uv} denote the radius and distortion coefficient corresponding to the image plane. r_{duv} denotes the distorted version. The above distortion modeling is used because the blind distortion removal technique currently does not consider the calibration of the camera intrinsic parameters. Thus, the lens distortion that is defined using quantities corresponding to the image plane is straightforward. Similar comparison is given here via two calibrated cameras, the desktop and the ODIS cameras, using the distortion function (3.46), along with the function (3.38) with $k_{\text{uv}1} = k_{\text{uv}2}$ for the radial distortion. We think that this double verification is needed since lens nonlinearity detection using higher-order spectral analysis is a recent development.

Original images of the desktop and the ODIS cameras are shown in fig. 3.16 (selected from fig. 3.13). Using the single-coefficient radial distortion model (3.38) (with $k_{\text{uv}1} = k_{\text{uv}2}$), the blindly compensated images of the two cameras are shown in fig. 3.17. An image interpolation operator is applied during the lens distortion compensation, since the observed original images are shrunk due to the negative distortion coefficients. The

undistorted images are only plotted in gray-level to illustrate the lens compensation results.⁷ Comparing fig. 3.16 with fig. 3.17, it can be observed that lens distortion is reduced significantly, though not completely and perfectly.

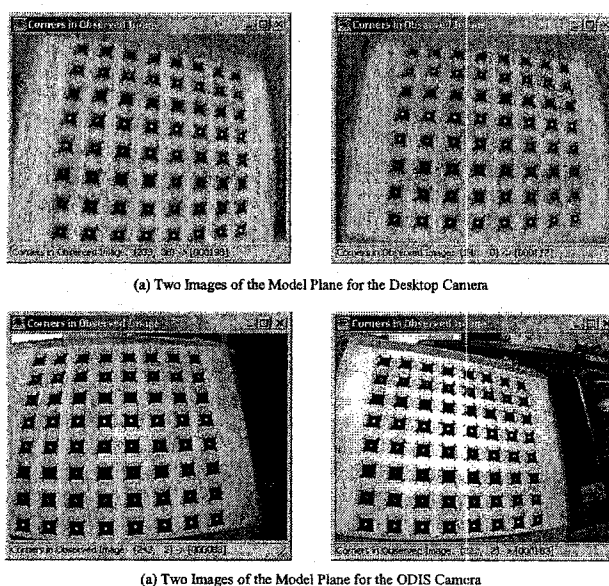


Fig. 3.16: Two sample images of the model plane with the extracted corners (indicated by cross) for the desktop and ODIS cameras.

Using the planar calibration target shown in fig. 3.6, the desktop and ODIS cameras have been calibrated in section 3.4 using the planar-based camera calibration technique described in [12]. However, the calibrated camera parameters in [27, 61] are in the normalized camera frame, while the blind removal technique deals with distortions in the image plane directly. A transformation between the lens distortion coefficients in the camera frame and those in the image plane is thus needed for a comparison purpose.

A rough transformation is illustrated in the following using the obtained intrinsic parameters from the planar-target based calibration technique. From equation (3.6), we

⁷The image interpolation operator currently applied is an average operator around each un-visited pixel in the compensated image. The resultant undistorted images might have noise and blur due to this simple operator.

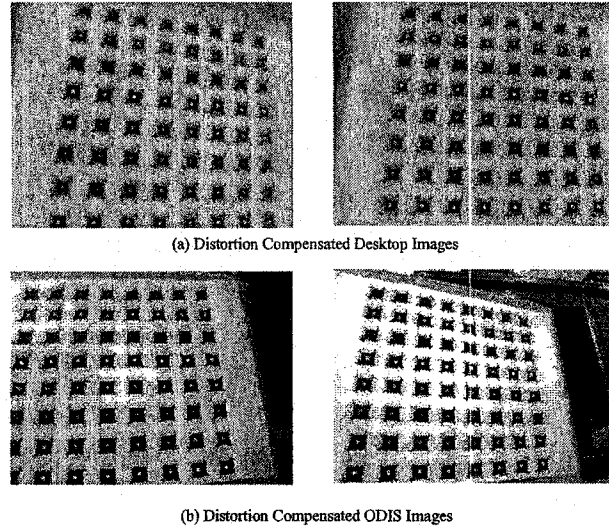


Fig. 3.17: Blindly radially compensated desktop and ODIS images with distortion function (3.38) with $k_{uv1} = k_{uv2}$.

have $(u - u_0) = \alpha x + \gamma y$, $(v - v_0) = \beta y$. Assuming that

$$\gamma \approx 0, \quad \alpha \approx \beta, \quad (3.47)$$

for a coarse approximation and using a single-coefficient radial distortion model (3.46), we have

$$r_{duv} = \alpha r_{dxy}, \quad r_{uv} = \alpha r_{xy}. \quad (3.48)$$

The relationship between k_{uv} and k_{xy} can be determined in a straightforward manner to be

$$k_{uv} = k_{xy}/\alpha^2. \quad (3.49)$$

In this section, blind lens distortion compensation is implemented via Matlab using higher-order spectral analysis toolbox following the procedures listed in section 3.7.1.2. However, instead of using equation (3.44) as the objective function to minimize, the maximum value of bicoherence is used in our implementation as the measurement criterion for

nonlinearity, which is⁸

$$J = \frac{1}{L_1 L_2} \max_{\omega_1 \in [-\frac{L_1}{2}, \frac{L_1}{2}], \omega_2 \in [-\frac{L_2}{2}, \frac{L_2}{2}]} \hat{b} \left(\frac{2\pi\omega_1}{L}, \frac{2\pi\omega_2}{L} \right), \quad (3.50)$$

for an input image of dimension $L_1 \times L_2$.

Comparison of the distortion coefficients obtained from the blind removal and the target-based [12] calibration techniques is shown in table. 3.11 using the functions (3.46) and (3.38) with $k_{uv1} = k_{uv2}$. It can be observed that, despite the deviation of the blindly calibrated results from those based on calibration targets, there is a consistency about the trend qualitatively. Notice that the distortion coefficients under the “Target” column in table 3.11 are obtained via approximations, where of course more precise transformations can be achieved without applying the assumptions in (3.47). However, since currently the blind removal method considers only the lens distortion (no calibration of the center of distortion), precise comparison can not be achieved even with precise calculation from the side of the target-based algorithm. Generally, the comparison can only be performed “quantitatively,” though the blind removal technique does provide another quantitative criterion for evaluating the calibration accuracy.

Table 3.11: Comparison of Lens Distortion Coefficients of the Blind Removal Technique and the Target-Based Algorithm

Eqn.	Camera	Blind Technique (10^{-6})		Target (10^{-6})
		Values	Mean	
(3.46)	Desktop	$-[3.5, 4.5, 3.5, 4.5, 1.5]$	-3.5	-3.73
	ODIS	$-[3.5, 2.5, 2.5, 3.5, 3.5]$	-3.1	-4.13
(3.38)*	Desktop	$[5, 6, 5, 5, 1]$	4.4	4.27
	ODIS	$[5, 3, 4, 5, 4]$	4.2	4.75

* $k_{uv1} = k_{uv2}$ for modeling the radial distortion.

One issue in the implementation is how to select the searching range for an image. While an image normalization method is commonly applied, in our simulation, the searching range is determined based on the image’s dimension and the observed judgement of

⁸The reason to use the criterion in (3.50) is more experimental. In our simulations, distortion coefficients obtained via the average sum criterion in (3.44) deviates from the previously calibrated coefficients significantly. However, when using the maximum bicoherence criterion (3.50), close and reasonable calibration results can be obtained for both the desktop and the ODIS cameras.

radial or pincushion distortions in a non-normalized way. This is the reason why in table 3.11, all the distortion coefficients are very small values. More specifically, consider the radial distortion function (3.46) with the maximum possible distortion at the image boundary. Let $r = r_{\max}$ and $r_d = \rho r_{\max}$, where r_{\max} is defined to be $r_{\max} = \sqrt{u_0^2 + v_0^2}$ on the image plane and the subscript $_{uv}$ is dropped for simplicity. We have

$$k_{uv} = (\rho - 1)/r_{\max}^2. \quad (3.51)$$

For the desktop and ODIS images, the dimension of the images is 320×240 in pixels. Further, the distortion experienced by these two cameras is a barrel distortion with $r_d < r$. Focusing on the barrel distortion by considering $\rho \in [0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1]$, we have

$$k_{uv} = \begin{cases} -6.25 \times 10^{-6}, & \text{when } \rho = \frac{3}{4}, \\ 0, & \text{when } \rho = 1. \end{cases} \quad (3.52)$$

The initial searching range for the distortion coefficients when using the radial distortion function (3.46) is chosen to be within $[-4.5 \times 10^{-6} \sim 3.5 \times 10^{-6}]$ with a step size 10^{-6} . Similarly, for the radial distortion function (3.38) with $k_{uv1} = k_{uv2}$, we have

$$k_{uv} = (\frac{1}{\rho} - 1)/r_{\max}^2, \quad (3.53)$$

and

$$k_{uv} = \begin{cases} 8.3 \times 10^{-6}, & \text{when } \rho = \frac{3}{4}, \\ 0, & \text{when } \rho = 1. \end{cases} \quad (3.54)$$

The initial searching range when using function (3.38) is $[0 \sim 9 \times 10^{-6}]$.

Relative values of J as defined in (3.50) of the five ODIS images using the distortion functions (3.46) and (3.38) with $k_{uv1} = k_{uv2}$ are shown in figs. 3.18 and 3.19. The relative J values equal to their corresponding values minus the minimum value in this group.

3.7.2.2 Blind Detection and Compensation of Lens Geometric Nonlinearity

Besides the radial distortion modelling, a simplified geometric distortion modelling method has been developed in equation (3.36). A straightforward question is whether the higher-order correlation detection in the frequency domain can help for the detection of

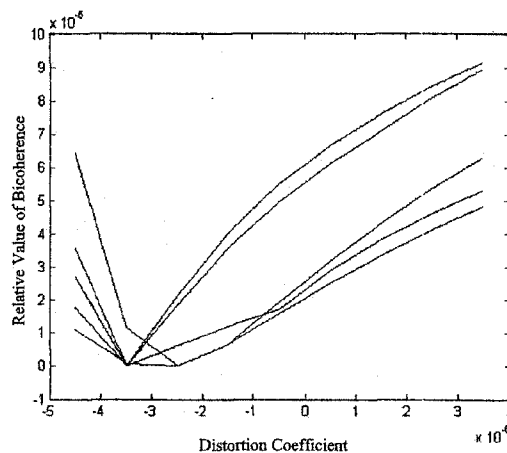


Fig. 3.18: Relative J values of the ODIS images using function (3.46).

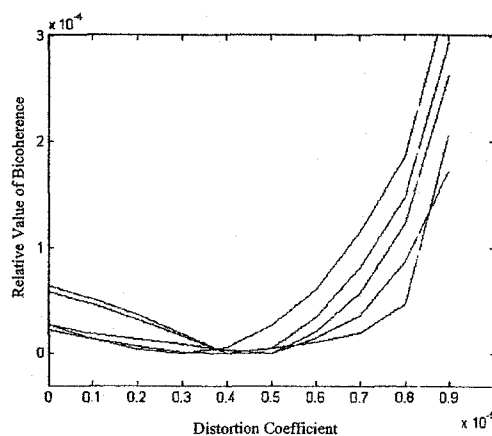


Fig. 3.19: Relative J values of the ODIS images using function (3.38).

geometric distortions, instead of just the radial one. The above problem is pursued in the next, where we first describe our conclusion. That is, the blind lens removal technique helps to detect the possible geometric distortion of a camera. By allowing the two distortion coefficients along the two image axes to be searched separately in two regions, the blindly calibrated lens distortion coefficients manifest noticeable difference for cameras that have been reported to be more accurately modelled by a geometric distortion modelling method.

Again using the two calibrated desktop and the ODIS cameras, the blindly calibrated distortion coefficients k_{uv1} and k_{uv2} using the rational distortion function (3.38) are shown in table 3.12, where the difference between the two distortion coefficients is significant for the ODIS camera, which has been studied in [61] to be better modelled by a geometric distortion model than a radial one. It is also observed that this difference between distortion coefficients along the two image axes is exaggerated using the blind removal technique.

Table 3.12: Blind Detection of Geometric Distortion

	Desktop (10^{-6}) [k_{uv1}, k_{uv2}]	ODIS (10^{-6}) [k_{uv1}, k_{uv2}]
Distortion Coefficients	[6.16, 5.28]	[2.64, 6.16]
	[2.64, 4.40]	[4.40, 6.16]
	[5.28, 6.16]	[2.64, 4.40]
	[2.64, 3.52]	[5.28, 6.16]
	[5.28, 3.52]	[3.52, 6.16]
Mean	[4.40, 4.57]	[3.70, 5.81]

In our implementation, geometric undistortion is implemented using equation (3.38) for each slice passing through the image center, which is chosen to be 12° apart for each image⁹. After (u, v) are derived from (u_d, v_d) and (k_{uv1}, k_{uv2}) , where the (k_{uv1}, k_{uv2}) are determined through the searching procedures, r_{uv} is calculate from (u, v) and the image center (which is assumed to be the center of distortion in the context of blind lens distortion removal technique). Nonlinearity detection using bicoherence is performed on this one-dimensional signal r_{uv} , which are basically the same procedures as used in the blind removal of radial distortion.

Due to the available knowledge of the distortion coefficients for the two sets of images using the radial distortion modeling with function (3.38) for $k_{uv1} = k_{uv2}$, when performing the detection for possible geometric distortions, the searching ranges are chosen to be around the distortion coefficients already determined in the radial case.

⁹Four images are reported to be enough to output unbiased distortion coefficients [15]. In this experiment, five images for each camera are used.

Concluding Remarks and Discussions

The higher-order correlation based technique is a promising method to detect lens nonlinearities in the absence of the camera. Besides the commonly used single-coefficient polynomial radial distortion model, blind detection of the geometric distortion was addressed in this section. Using the quantitative measurement criterion defined in the frequency domain, the difference between the two sets of distortion coefficients along the two image axes can be used to qualitatively detect cameras that are better modelled by a geometric distortion modelling method.

3.8 Summary

This chapter addresses lens distortion modeling functions. Rational functions and piecewise smooth functions that are either simple polynomials (and their derivations) or functions of polynomials are proposed. Extensive and detailed experimental results are presented for the model evaluation and selection. The various experiments presented in section 3.6 can serve as a general guidance for evaluation of distortion calibration alone.

Using a calibrated camera, vision tasks can be simplified and straightforward. 3-D motion estimation and range identification of a perspective dynamic system will be addressed in chapter 4.

Chapter 4

Perspective Dynamic Systems and Observer Design

Using the geometry and kinematics of the environment is a basic technique used by humans as they successfully accomplish tasks such as walking, driving, and recognizing and grasping objects. Emulation of this human skill has been one of the principal goals of artificial intelligence research, starting from the early 1970's; that is, to build machines that recognize the shape and motion of objects within the environment [16]. Since the 1970's, a variety of motion estimation algorithms have been developed, gaining attention from researchers and scientists in the areas of computer vision, image processing, robotics, and control. With the aid of fast and inexpensive computers, vision algorithms that can run in real-time are feasible and find application in robot navigation, medical imaging, and video conferencing. Even though motion estimation is not a new topic, interest in this field is increasing. Current research is aimed at making robust, accurate, and efficient algorithms that are able to detect multiple rigid motions in the presence of occlusions and discontinuities with as few *a priori* assumptions as possible [63]. Moreover, 3-D motion estimation of nonrigid motions begins to be addressed [20, 21, 22, 64]. However, the goal is still far from being reached. Indeed, it opens a new and exciting avenue of research in nonlinear system theory. Appropriate tools from nonlinear estimation/identification theory are beginning to be exploited and only recently have such tools hinted at acceptable solutions [16].

Due to the large volume of available algorithms, there is a need to classify them into appropriate categories based on certain criterion such that the whole literature becomes more organized. One objective of this chapter is to fulfill this need so as to help new researchers interested in this field get a full sense of the literature before devoting time to any particular direction.

Criteria for classification of existing algorithms can include many possible candidates, such as motion models (none, rigid, affine, Riccati), camera models (perspective, weak-perspective, orthographic), number of frames used (two, multiple, recursive), estimation capability of multiple motions, linear or nonlinear, time-domain or frequency-domain, and feature-based (point, line, curve, snake, region) or *optical flow*.¹ Classification has been made in [65] using the feature-based/optical flow criterion, where the optical flow method is based on assumptions about the image intensity while the feature-based method uses a relatively sparse set of two-dimensional features that are extracted from the images.

A very good tutorial of 2-D motion estimation algorithms is presented in [66], where different models, estimation criteria, and searching strategies are listed briefly. It also clarifies the difference between 2-D and 3-D motion estimation. While the purpose of 3-D motion estimation is to recover the motion parameters of a moving object in the 3-D world space, 2-D motion estimation considers only the corresponding movement on the image plane. 2-D motion estimation uses certain estimation criterion and searching strategies to track the movement on the image plane using image processing approaches. Motion dynamics on the image plane can thus be obtained. The corresponding 3-D motion parameters can be further estimated uniquely, or to a certain group of action [20, 21], using the available 2-D motion parameters. In this way, 2-D motion estimation can be regarded as one step for the 3-D motion estimation.

A review of 3-D motion and structure estimation algorithms using feature correspondence is presented in [67], where three categories of problems are considered, depending on whether the correspondence is 3D-3D, 2D-3D, or 2D-2D. While 3D-3D feature correspondence has application in robot navigation when the robot is equipped with range sensors, and 2D-3D correspondence is applicable to camera calibration and pose estimation tasks, the main focus of our work is on the 3-D motion estimation using 2D-2D correspondence,

¹In the computer vision literature, a distinction is often made between 2-D motion and optical flow. Optical flow does not correspond to the 2-D velocity field unless very special conditions are satisfied, such as no illumination changes [65]. In this section, “optical flow” is used to refer to apparent motion, which is calculated using low-level image descriptors such as intensity.

since this is all the information available from the image sequences without any prior assumptions, such as depth and physical size of the object.

The classification criterion used in [67] is based on the type of mathematics used in the algorithms. Following [67], in this section, classification of the existing 3-D motion estimation algorithms is made based on the formulation of the problem and the mathematical tools applied. We show that most of the existing 3-D motion estimation algorithms, or at least the algorithms reviewed in this section, are formulated in a way that finally come to the form of an optimization problem, either linear or nonlinear. Throughout this section, the problem in concern is to estimate 3-D motion of a moving object with a single stationary calibrated² camera under the perspective projection, unless otherwise stated. For the cases when the camera is moving while the object is stationary, or both the camera and the object are moving, the problem formulation is briefly discussed at the end of this section (section 4.8).

4.1 General Notation and Motion Models

Referring to the perspective projection in fig. 3.1 and equation (3.6), in this section, we adopt a loose distinction between (x_i, y_i) and (u_i, v_i) , where the subscript i denotes the i^{th} feature point, since their relationship is represented by the linear transform via A_{intr} , which is not the key issue in motion estimation, where we are mainly concerned with the nonlinearity introduced by the perspective projection during the normalization process.

For 3-D motion estimation, early works focused on 3-D rigid motion. Recently, researchers have begun to discuss nonrigid motions, either by representing it as a combination of rigid motion and some deformations [68], or by assuming some simple models, such as

²Camera calibration includes the calibration of a camera's intrinsic parameters and its lens distortion compensation.

affine and Riccati motions, where Riccati motion is represented by [22]:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}}_{\mathbf{b}} + \underbrace{\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}}_A \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + F \begin{bmatrix} X^2 \\ XY \\ XZ \\ Y^2 \\ YZ \\ Z^2 \end{bmatrix}, \quad (4.1)$$

with

$$F = \begin{bmatrix} f_1 & f_2 & f_3 & 0 & 0 & 0 \\ 0 & f_1 & 0 & f_2 & f_3 & 0 \\ 0 & 0 & f_1 & 0 & f_2 & f_3 \end{bmatrix}.$$

When $F = 0$, equation (4.1) represents an affine motion. Furthermore, if $F = 0$ and the matrix A in (4.1) is skew-symmetric, that is, when

$$A = \Omega = \begin{bmatrix} 0 & w_1 & w_2 \\ -w_1 & 0 & w_3 \\ -w_2 & -w_3 & 0 \end{bmatrix}, \quad (4.2)$$

the Riccati motion becomes a rigid motion.

The problem of 3-D motion estimation is to estimate the position and orientation of the object via observations on the image plane. Existing algorithms have been carried out in two subcategories, where one category performs direct calculation of (R, \mathbf{t}) (as in sections 4.2.1 ~ 4.2.5), while the other assumes certain motion dynamics, such as the general Riccati motion (4.1), and identifies the corresponding motion parameters (as in section 4.2.6 and 4.3).

4.2 3-D Motion Estimation Techniques

A direct approach to the 3-D motion estimation problem is to formulate it as a nonlinear optimization problem, where classical optimization algorithms, such as the Gauss-Newton and the Levenberg-Marquardt optimization methods, can be used to search for the optimal solution. Though very accurate, this classical nonlinear method is computationally expensive, which prevents it from being applied to real-time applications. Algorithms

that are based on linearization become prevalent. Generally, they are formulated as a minimization problem that can be solved either by SVD (Singular Value Decomposition) or recursively [16, 17, 18, 19]. Recently, exciting results have been published on the perspective dynamic system theory, where the system's observability/identifiability property is discussed in a more theoretical way and most of the known results in the computer vision literature are revealed [20, 21]. Besides the rigid motion, nonrigid motions such as the affine and Riccati motions are discussed in this general framework [22].

In this section, we assume that a set of suitable features on the image plane, point features (N feature points) or active contour [17], have been extracted and the correspondences of these features between consecutive frames are established. Further, the features are always assumed to be within the field of view (FOV) of the camera.

4.2.1 Nonlinear Optimization Formulation

Assume that an object is undergoing a rotational and translational motion described by (R, \mathbf{t}) . At time instant t , for the i -th point,

$$\lambda [u_i, v_i, 1]^T = A_{\text{intr}} q_i. \quad (4.3)$$

At time instant $t + 1$,

$$\lambda' \begin{bmatrix} u'_i \\ v'_i \\ 1 \end{bmatrix} = A_{\text{intr}} q'_i = A_{\text{intr}} (R q_i + \mathbf{t}) = A_{\text{intr}} \left(\lambda R A_{\text{intr}}^{-1} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} + \mathbf{t} \right). \quad (4.4)$$

Consider a set of 2-D displacement vectors

$$\begin{bmatrix} \Delta u_i \\ \Delta v_i \end{bmatrix} = \begin{bmatrix} u'_i - u_i \\ v'_i - v_i \end{bmatrix}, \quad \text{for } i = 1, \dots, N. \quad (4.5)$$

To calculate the displacement vectors in (4.5), the camera's intrinsic matrix A_{intr} in (4.3) and (4.4) can be simplified as

$$A_{\text{intr}} = \begin{bmatrix} \alpha & \gamma & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (4.6)$$

since we are interested in the displacement on the image plane. To illustrate the basic idea of the algorithm, consider a simple case when $\gamma = 0$. Using the intrinsic matrix A_{intr} in (4.6), equation (4.4) becomes

$$\begin{aligned} \lambda' \begin{bmatrix} u'_i \\ v'_i \\ 1 \end{bmatrix} &= A_{\text{intr}} \left(\lambda R A_{\text{intr}}^{-1} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} + \mathbf{t} \right), \\ &= \begin{bmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \left(\lambda R \begin{bmatrix} 1/\alpha & 0 & 0 \\ 0 & 1/\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \right), \\ &= \lambda \begin{bmatrix} u_i r_{11} + \alpha/\beta v_i r_{12} + \alpha r_{13} + \alpha t_1/\lambda \\ \beta/\alpha u_i r_{21} + v_i r_{22} + \beta r_{23} + \beta t_2/\lambda \\ u_i r_{31}/\alpha + v_i r_{32}/\beta + r_{33} + t_3/\lambda \end{bmatrix}. \end{aligned}$$

Thus, we have

$$\begin{bmatrix} \Delta u_i \\ \Delta v_i \end{bmatrix} = \begin{bmatrix} \frac{u_i r_{11} + \alpha/\beta v_i r_{12} + \alpha r_{13} + \alpha t_1/\lambda}{u_i r_{31}/\alpha + v_i r_{32}/\beta + r_{33} + t_3/\lambda} - u_i \\ \frac{\alpha/\beta u_i r_{21} + v_i r_{22} + \beta r_{23} + \beta t_2/\lambda}{u_i r_{31}/\alpha + v_i r_{32}/\beta + r_{33} + t_3/\lambda} - v_i \end{bmatrix}. \quad (4.7)$$

Let $\mu = [t_1, t_2, t_3, \omega_1, \omega_2, \omega_3]$. By eliminating λ from (4.7), we can obtain a nonlinear equation in the following form for the i -th point [69]:

$$f_i(\mu) = f_i(t_1, t_2, t_3, \omega_1, \omega_2, \omega_3) = 0, \quad (4.8)$$

whose solution is in the form of

$$(\lambda t_1, \lambda t_2, \lambda t_3, \omega_1, \omega_2, \omega_3), \quad (4.9)$$

where the scalar λ indicates the ambiguity in the translational components. A detailed equation of (4.8) can be found in [69] with $\gamma = 0$. When $\gamma \neq 0$, a similar nonlinear equation can be derived but with more complexity.

Generally speaking, the nonlinear minimization approach suffers from the initial value selection problem. In order to solve the nonlinear equations robustly, an initial guess algorithm is proposed in [69], where the optimal solution is guaranteed due to the initial

values determined using a weak-perspective projection model.³ and an image coordinate normalization technique. The initial guess algorithm proposed in [69] decouples the translational parameters from the rotational ones, so that robust initial values for the solution of the original nonlinear problem can be obtained easily. This proposed nonlinear solver is shown to have a better performance compared with the Total Least Squares (TLS) and Least Squares (LS) methods in [70, 71] that are presented in section 4.2.2.

4.2.2 Linear LS/TLS Algorithms

The classical LS problem considers the presence of noise only in the observation vector \mathbf{b} (in the form of $M\mathbf{x} = \mathbf{b}$) and it minimizes the error $\|M\mathbf{x} - \mathbf{b}\|_2$. The LS problem can be recast in the following form

$$\min_{\mathbf{b} + \mathbf{e} \in \text{range}\{M\}} \|\mathbf{e}\|_2, \quad \mathbf{e} \in \mathbb{R}^m.$$

In many cases, it is more appropriate to assume that the error is also present in the M matrix (in the form of $[M + E]\mathbf{x} = \mathbf{b} + \mathbf{e}$). This problem is known as the TLS problem, which can be recast as

$$\min_{\mathbf{b} + \mathbf{e} \in \text{range}\{M + E\}} \|[E \mid \mathbf{e}]\|_F, \quad \mathbf{e} \in \mathbb{R}^m, E \in \mathbb{R}^{m \times n}.$$

Let us consider the case of a 3-D moving object and a stationary camera tracking the scene. Suppose a 3-D point undergoes the transformation

$$[X_i^{c'}, Y_i^{c'}, Z_i^{c'}]^T = R[X_i^c, Y_i^c, Z_i^c]^T + \mathbf{t}.$$

Let

$$\begin{aligned} \begin{bmatrix} x_i \\ y_i \end{bmatrix} &= \begin{bmatrix} X_i^c / Z_i^c \\ Y_i^c / Z_i^c \end{bmatrix}, \quad \begin{bmatrix} x'_i \\ y'_i \end{bmatrix} = \begin{bmatrix} X_i^{c'} / Z_i^{c'} \\ Y_i^{c'} / Z_i^{c'} \end{bmatrix}, \\ \begin{bmatrix} \Delta x_i \\ \Delta y_i \end{bmatrix} &= \begin{bmatrix} x'_i - x_i \\ y'_i - y_i \end{bmatrix}. \end{aligned} \tag{4.10}$$

³Weak-perspective projection is an approximation to the perspective camera model. Weak-perspective projection is valid when the object is close to the optical axis of the camera and the depths of all feature locations are roughly equal to the principle depth, conventionally chosen to be the depth of the origin of the object [63]. Under the weak-perspective projection, the projection on the image plane is modelled as $x_i = sX, y_i = sY$, where s is a scale factor.

If we assume that 1) the translation component t_3 along the Z direction is much smaller than the Z_i 's, 2) the rotation components $(\omega_1, \omega_2, \omega_3)$ are small (less than 5°), and 3) the field of view of the camera is small, we can write [18, 72]:

$$\begin{bmatrix} \Delta x_i \\ \Delta y_i \end{bmatrix} = \begin{bmatrix} \frac{t_1 - x_i t_3}{Z_i^c} + \omega_1 x_i y_i - \omega_2(1 + x_i^2) + \omega_3 y_i \\ \frac{t_2 - y_i t_3}{Z_i^c} - \omega_2 x_i y_i + \omega_1(1 + y_i^2) - \omega_3 x_i \end{bmatrix}. \quad (4.11)$$

Eliminating Z_i^c from the above equation leads to the following linear equation

$$M \theta = 0, \quad (4.12)$$

where

$$\theta = [\theta_1, \dots, \theta_9]^T = [t_1, t_2, t_3, t_1 \omega_1, t_2 \omega_2, t_3 \omega_3, t_1 \omega_2 + t_2 \omega_1, t_1 \omega_3 + t_3 \omega_1, t_2 \omega_3 + t_3 \omega_2]^T, \quad (4.13)$$

and

$$M^T = \begin{bmatrix} \Delta y_1 & \cdots & \Delta y_N \\ -\Delta x_1 & \cdots & -\Delta x_N \\ \Delta x_1 y_1 - \Delta y_1 x_1 & \cdots & \Delta x_N y_N - \Delta y_N x_N \\ -(1 + y_1^2) & \cdots & -(1 + y_N^2) \\ -(1 + x_1^2) & \cdots & -(1 + x_N^2) \\ -(x_1^2 + y_1^2) & \cdots & -(x_N^2 + y_N^2) \\ x_1 y_1 & \cdots & x_N y_N \\ x_1 & \cdots & x_N \\ y_1 & \cdots & y_N \end{bmatrix}. \quad (4.14)$$

Moreover, the elements of θ are subject to the following constraints

$$\begin{cases} \theta_7 = \frac{\theta_1 \theta_5}{\theta_2} + \frac{\theta_2 \theta_4}{\theta_1} \\ \theta_8 = \frac{\theta_1 \theta_6}{\theta_3} + \frac{\theta_3 \theta_4}{\theta_1} \\ \theta_9 = \frac{\theta_2 \theta_6}{\theta_3} + \frac{\theta_3 \theta_5}{\theta_2} \end{cases}. \quad (4.15)$$

Equation (4.12) can be formulated as a LS problem since the right hand side is not exactly zero. It can also be solved as a TLS problem when the error in the M matrix in (4.14) is considered. After formulated as a LS/TLS problem, the solution of parameter θ is based on the SVD of a combined data matrix as described in [18, 70, 71].

The shortcoming of the above SVD-based method lies in its sensitivity to noise [71]. In real data, there are multiple sources of noise. In general, three kinds of errors should be considered [18]:

- 1) The measurement noise: noise in the extraction of feature points.
- 2) The approximation error: in the formulation of $M\theta = 0$, the right hand side is not exactly zero.
- 3) The quantization noise.

In [18], two schemes are proposed to tackle the measurement noise problem. First, a recursive robust scheme is introduced to reject outliers from the data set based on the residuals information. The basic idea is the introduction of a weight for every measurement, where the weight reflects the confidence of the system to this particular measurement. Recursively, using the residual information of the system, the weights are stabilized and only the good measurements are held to complete the estimation process. The above weighting scheme can be further improved by generating a number of smaller sub-matrices M_j that are formed by randomly selecting rows from the M matrix. Using more sub-matrices, the probability that two of them contain sufficiently low percentage of outliers is increased so that the algorithm will yield estimate $\hat{\theta}$ close to θ . Therefore, if two estimates from two different sub-matrices lie close to each other, it is more likely that they are close to the true value θ .

4.2.3 Using Epipolar Constraints

“Vision in the loop” raises new and interesting problems of a system theoretic flavor, such as analysis and control of new classes of dynamic systems. Crucial issues in the use of vision as a sensor in the control systems are, for example, nonlinear observability and identifiability in a projective geometric framework as well as estimation and control on peculiar topological manifolds [16].

Following the perspective dynamic system theory in [20, 73], in which the feasibility of motion and structure estimation is assessed, instead of studying motion estimation for a

known structure as in the above-cited references (where a planar surface structure is usually assumed), motion estimation for unknown structure is studied in [16, 74, 75], where the visual motion estimation problem is formulated in terms of identification of a nonlinear implicit system with parameters on a topological manifold, i.e., the essential manifold. The formulation using the essential manifold has the advantage that the estimation of motion is decoupled from the estimation of structure of the object being viewed, and therefore it is possible to handle occlusions in a principled way.

The formulation of motion estimation problems on the essential manifold is described briefly in the following. First, the *essential matrix* E_{ss} is defined as

$$E_{ss} \doteq \{SR \mid R \in SO(3), S = (\mathbf{t} \wedge) \in so(3)\} \subset \mathbb{R}^{3 \times 3}, \quad (4.16)$$

where \wedge is the wedge product and $so(3)$ is a group of skew-symmetric matrices.⁴

For any given rigid motion $(R, \mathbf{t}) \in SE(3)$, there exists an essential matrix Q defined by:

$$Q \doteq (\mathbf{t} \wedge)R. \quad (4.17)$$

Given the essential matrix Q in (4.17), the inverse problem of finding (R, \mathbf{t}) can be achieved by the following map defined between E_{ss} and \mathbb{R}^6 [16]:

$$\begin{aligned} \Phi: E_{ss} &\rightarrow \mathbb{R}^3 \times SO(3) \rightarrow \mathbb{R}^3 \times \mathbb{R}^3, \\ Q &\mapsto \begin{bmatrix} \pm \|Q\| U_{.3} \\ U R_z(\pm \frac{\pi}{2}) V^T \end{bmatrix} = \begin{bmatrix} \mathbf{t} \\ R \end{bmatrix}, \end{aligned} \quad (4.18)$$

where U, V are defined by the SVD of $Q = U\Sigma V^T$; $U_{.3}$ is the third column of U , and $R_z(\pi/2)$ is a rotation of $\pi/2$ about the axis $[0, 0, 1]^T$. Note that the map Φ defines \mathbf{t} with a sign ambiguity. However, this ambiguity can be resolved in the context of the visual motion estimation by imposing the positive depth constraint, which means that each visible point

⁴Definitions of $SO(3)$, $SE(3)$, and $so(3)$: $SO(3) = \{R \in \mathbb{R}^{3 \times 3} : RR^T = I, \det R = \pm 1\}$, $SE(3) = \{(p, R) : p \in \mathbb{R}^3, R \in SO(3)\}$, and $so(3) = \{S \in \mathbb{R}^{3 \times 3} : S^T = -S\}$. The wedge product of a 3×1 vector is defined by $\mathbf{t} \wedge = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$.

lies in front of the viewer. Thus, a unique mapping is established between (R, \mathbf{t}) and Q . The problem of estimating (R, \mathbf{t}) can be achieved indirectly by estimating Q first.

The well-known *essential constraint* or the *epipolar constraint* is [16]

$$m_i'^T (\mathbf{t} \wedge (R m_i)) = 0, \quad \forall i = 1, \dots, N, \quad (4.19)$$

where $m_i' = [x_i', y_i']^T$ and $m_i = [x_i, y_i]^T$ are the i -th normalized feature point in the camera frame at time instant $t+1$ and t , respectively. Using the definition of Q in (4.17), equation (4.19) becomes

$$m_i'^T Q m_i = 0, \quad \forall i = 1, \dots, N. \quad (4.20)$$

Now, estimating the motion amounts to identifying the model

$$\begin{aligned} (Q m_i)^T m_i' &= 0, \quad Q \in E_{\text{ss}}, \\ y_i &= m_i + v_i, \quad \forall i = 1, \dots, N, \quad v_i \in N(0, R_v), \end{aligned} \quad (4.21)$$

where $N(0, R_v)$ denotes Gaussian noise with zero mean and covariance R_v . From (4.20), we have

$$\begin{aligned} &[x'x, x'y, x'y', y'x, y'y, y'y', x, y, 1] \cdot \\ &[Q_{11}, Q_{12}, Q_{13}, Q_{21}, Q_{22}, Q_{23}, Q_{31}, Q_{32}, Q_{33}]^T = 0. \end{aligned} \quad (4.22)$$

When having N feature points, we can stack them to form a $N \times 9$ matrix \mathcal{X} , thus giving

$$\mathcal{X}_{m'(t), m(t)} Q(t) = 0. \quad (4.23)$$

Finally, the visual motion estimation problem is characterized as

$$\begin{aligned} Q(t+1) &\doteq Q(t) + \omega(t), \quad Q \in E_{\text{ss}}, \quad \omega(t) \in N(0, R_\omega), \\ 0 &= \mathcal{X}_{m'(t), m(t)} Q(t), \\ y_i &= m_i + v_i, \quad \forall i = 1, \dots, N, \quad v_i \in N(0, R_v). \end{aligned} \quad (4.24)$$

In this way, the 3-D structure of the scene is removed from the model, ending up with a nonlinear implicit dynamic model for the (measured) projective coordinates of the visible features with motion as unknown parameters constrained on E_{ss} .

A modification of the scheme in [16] is applied to the motion estimation of video coding in [76] and better performance is achieved in the sense of reliability to noisy measurements.

4.2.4 Lie Group Formalism

A 3-D rigid motion tracking algorithm for objects with known 3-D structures using *active contour*, the *snake*,⁵ is proposed in [17], where a Lie group formalism is used to cast the motion computation problem into simple geometric terms so that tracking becomes a simple optimization problem solved by means of iterative re-weighted least squares. This approach also takes advantage of the *aperture problem* (that the component of motion of an edge, tangent to itself, is not observable locally). It yields a significant benefit since the search for intensity discontinuities in the image sequence can be limited to a one dimensional path that lies along the edge normal \hat{n} and thus has a linear complexity in the search space. Computing the motion is done by minimizing the squared error between the transformed edge position and the actual edge position (in pixels) in the least squares sense.

Rigid motion of the camera relative to the object between consecutive frames can be represented by a Euclidean transformation of the form [17]

$$M = \begin{bmatrix} R & \mathbf{t} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (4.25)$$

The M matrix forms a 4×4 matrix representation of the group $SE(3)$ of rigid body motions in the 3-D space, which is a 6-D Lie Group. The generators of the Lie group are typically taken as translations in the x , y , and z directions and rotations about the x , y , and z axes, represented by the following six matrices:

$$G_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad G_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

⁵Snake is a deformable contour that moves under a variety of image constraints (which tend to be local) and object-model constraints. The snake algorithm seeks optimal contours by incorporating the local strengths of the edges and their spatial distribution into its energy functional. The name of snake comes from the resemblance between the active contour and snake as it moves.

$$\begin{aligned}
G_2 &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, & G_5 &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \\
G_3 &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, & G_6 &= \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.
\end{aligned}$$

Using the above six Lie group operators, M can be obtained via the exponential map [17]

$$M = e^{\alpha_i G_i}, \quad (4.26)$$

which can be approximated by:

$$M = I + \alpha_i G_i. \quad (4.27)$$

The problem now becomes how to derive α_i , which is done using the standard least-squares algorithm.

Let f_i^ξ describe the magnitude of the edge normal motion that would be observed in the image at the ξ -th sample point for the i -th group operator, and d the distance to the actual edge position, the process to derive α_i is performed by minimizing the squared error between the transformed edge position and the actual edge position (in pixels). That is, the objective function is chosen to be

$$J = \sum_{\xi} (d^\xi - \alpha_i f_i^\xi)^2, \quad (4.28)$$

where

$$\alpha_i = \left(\sum_{\xi} f_i^\xi f_j^\xi \right)^{-1} \sum_{\xi} d^\xi f_i^\xi \quad (4.29)$$

is the LS solution [17].

4.2.5 Neural Network-Based Estimation

Since 1990's, Neural Networks (NN) have been proposed to solve the motion estimation problem [68, 77, 78, 79]. The NN methods presented in [68, 77, 78] use 3-D point correspondences. They can estimate 3-D rigid motions [77, 78], as well as nonrigid [68]. The inputs to the NN are the components of the feature points. If the feature points are also the corners of certain patterns, such as triangles utilized in [68], additional constraints (i.e., the equal distance constraint between two points for rigid motion) can be applied. The weights of the NN can be updated with the delta updating rule [80] using the errors between the actual outputs of the NN and the extracted feature locations in the next time instant. For the nonrigid motion in [68], motion is decomposed into a global rigid motion and a set of local nonrigid deformations, which are coupled with the global motion at every time instant. While the global motion can be described using standard rigid motion parameters, the local deformation motion of interest is a combination of rotations along arbitrary axes with arbitrary angles, distortions in arbitrary directions, and expansions or contractions in arbitrary directions.

A NN model is introduced in [79] that estimates the 3-D rigid motion parameters of an object using 2-D motion vectors. Equal distance constraints are also applied to vertices of each triangle. Compared with [78], the NN scheme in [79] produces better 3-D motion estimation.

4.2.6 Extended Kalman Filter Approach

The 3-D motion estimation algorithms reviewed so far estimate (R, \mathbf{t}) directly. In this section, the object is assumed to follow a rigid motion dynamics, i.e., the general Riccati motion (4.1) with a skew-symmetric A matrix and $F = 0$, such that nonlinear observers, such as Extended Kalman Filter (EKF), can be used to estimate the states of a nonlinear system, where the states can be chosen to include the motion parameters and positions.

A brief description of the Kalman filter is given first. Given a discrete linear system

$$\begin{aligned} x_{k+1} &= F_k x_k + \omega_k, & \omega_k &\sim N(0, R_\omega), \\ y_k &= H_k x_k + v_k, & v_k &\sim N(0, R_v), \end{aligned} \tag{4.30}$$

where x_k is the state, ω_k represents the process noise, and v_k models the measurement noise, the Kalman filter optimally estimates x_k , represented by \hat{x}_k , by minimizing the following weighted mean-squared error

$$E [(x_k - \hat{x}_k)^T L (x_k - \hat{x}_k)], \quad (4.31)$$

where L is any symmetric nonnegative definite weighting matrix. This is the so-called Linear Quadratic Gaussian (LQG) problem, since the dynamic system is linear, the performance cost function is quadratic, and the random processes are Gaussian [81]. As stated in [81], the principal uses of linear filtering theory are for solving nonlinear problems. Now, for a discrete nonlinear system

$$x_{k+1} = f(x_k, k) + \omega_k, \quad \omega_k \sim N(0, R_\omega), \quad (4.32)$$

$$y_k = h(x_k, k) + v_k, \quad v_k \sim N(0, R_v), \quad (4.33)$$

where $h(\cdot)$ represents the perspective projection, the problem is to find an estimate \hat{x}_k for x_k , based on y_k . Since both $f(x_k, k)$ and $h(x_k, k)$ are nonlinear, the EKF is used and linearization of $f(x_k, k)$ and $h(x_k, k)$ is performed about an estimated trajectory, defined as [81]:

$$\begin{aligned} F^{[1]} &= \frac{\partial f(x, k)}{\partial x} \Big|_{x=\hat{x}_k}, \\ H^{[1]} &= \frac{\partial h(x, k)}{\partial x} \Big|_{x=\hat{x}_k}. \end{aligned} \quad (4.34)$$

For the 3-D rigid motion estimation and assuming N feature points, the following set of states is chosen in [19]:

$$s(t) = [\frac{X}{Z}, \frac{Y}{Z}, \frac{\dot{X}}{Z}, \frac{\dot{Y}}{Z}, \frac{\dot{Z}}{Z}, q_1, q_2, q_3, q_4, w_1, w_2, w_3, \frac{X_{1 \sim N}}{Z}, \frac{Y_{1 \sim N}}{Z}, \frac{Z_{1 \sim N}}{Z}], \quad (4.35)$$

where constant translational speed $(\dot{X}, \dot{Y}, \dot{Z})$ and constant rotational speed (w_1, w_2, w_3) are assumed. $[X, Y, Z]^T$ is the origin of the object-centered coordinate system and $\mathbf{q} = [q_1, q_2, q_3, q_4]^T$ is the quaternion representation of rotation.⁶ As can be seen from the states

⁶Quaternion is another representation of rotation matrix. Unlike Euler angles, quaternion gives a global representation of $SO(3)$, the group of rotation matrix, at the cost of using four numbers instead of three to represent a rotation. A quaternion is defined as: $(q_0, q_1, q_2, q_3)^T = (\cos(\theta/2), (n_1, n_2, n_3) \sin(\theta/2))^T$, where $\mathbf{n} = (n_1, n_2, n_3)^T$ represents the unit axis of rotation and θ represents the angle of rotation. One advantage of using quaternion is that the time-propagation is much simpler than the analogous system for propagating Euler angles.

in (4.35), all the positions and translational components are homogeneous in $1/Z$, which means that the depth information is not recovered. For the states chosen in (4.35), the time derivative of $s(t)$ is also a function of $s(t)$ [19], denoted by $f(s(t))$ here. Using $G_t\omega_t$ to represent either random or un-modelled deterministic deviations from the given plant model, we have

$$\dot{s}(t) = f(s(t)) + G_t\omega_t, \quad (4.36)$$

whose discrete model fits in the form of (4.32). The output measurement can also be described as a function of the states plus some measurement noise. Combined with (4.36), this problem formulation is appropriate for solution with an iterative EKF (IEKF) [19].

The value of the recursive approaches is that they usually require less computation time for each new set of data (each new image, for example). State estimates are continuously computed indexed to the current time, based on all past data, and can readily extrapolate the state estimates ahead in time to aid in preprocessing the next set of data. A modification of the IEKF in [19] is proposed in [82], where in addition to the image plane coordinates, the image plane velocity is also available. Thus, the output measurement function will be doubled in size and improvements can be expected in the estimation of velocity states, in the reduction of large initial estimation errors, and in the tracking of abrupt object maneuvers. A complete and detailed implementation of the Kalman-filter based algorithm is described in [83, 84], which runs on a personal computer.

The EKF is based on the linearization about an estimated trajectory. However, for the vision-based motion estimation problem, geometric structures of a perspective system will be lost if studied via linearization. Moreover, the EKF is much complicated since *a priori* knowledge of the noise distribution is required. Due to the above mentioned problems, efforts have been made towards other nonlinear observers for the perspective dynamic system, which are presented in the next section.

4.3 Perspective Dynamic System Approach

Besides the commonly assumed rigid motion discussed in the above sections, nonrigid motions, such as the affine and Riccati motions, have been discussed in the perspective

dynamic system (PDS) framework, which refers to a linear system with homogeneous observation function. The PDS has been found to be a good control theoretic framework for the motion estimation problems where CCD cameras are used as sensors. The motion estimation problem becomes an identification problem of a PDS. Within the PDS framework and assuming a planar surface structure, theoretical analysis shows that, for the general Riccati motion, motion parameters can only be identified up to a group of action, where the resultant homogeneous output remains the same. The perspective dynamic system theory (PDS theory) not only reveals some existing knowledge in the computer vision, but also provides a theoretical analysis about which motion parameter can be identified and to what extent.

Using the PDS formulation, nonlinear observers applicable to the PDS [20, 23, 85, 86, 87] have been used to estimate the states of a PDS, uniquely or to the extent possible, for simple cases when the object is undergoing some special motions, such as the rigid and pure rotational motions as described in section 4.3.2. For the general affine or Riccati motions, recent results on canonical forms of the PDS show that the parameters in the canonical forms can be identified uniquely using an EKF [88, 89, 90]. In order to estimate all the motion parameters uniquely, research has been carried out in the directions of using multiple cameras, integrating vision with range data, and using active vision system, which are discussed in section 4.8.

A PDS is described by [64]:

$$\dot{x} = Ax + Bu, \quad y = [Cx], \quad (4.37)$$

where the projective observation function is defined as

$$\begin{aligned} Y : \mathbb{R}^n - \mathcal{B} &\longrightarrow \mathbb{RP}^{m-1}, \\ x &\longmapsto [Cx], \end{aligned} \quad (4.38)$$

where $[Cx]$ is the homogeneous line spanned by the nonzero vector $Cx \in \mathbb{R}^m$. The set \mathcal{B} is defined as

$$\mathcal{B} = \{x : Cx = 0\}. \quad (4.39)$$

As seen from (4.37) and (4.38), the PDS is a linear dynamic system with a homogeneous observation function.

An example of the PDS when the object is moving according to an affine motion can be described as:

$$\begin{bmatrix} \dot{X}(t) \\ \dot{Y}(t) \\ \dot{Z}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}}_A \begin{bmatrix} X(t) \\ Y(t) \\ Z(t) \end{bmatrix} + \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}}_b u(t), \quad (4.40)$$

with $u(t) = 1$, $C = I_{3 \times 3}$, and the outputs $y(t)$ defined to be

$$y(t) = [y_1, y_2]^T = [X(t)/Z(t), Y(t)/Z(t)]^T, \quad (4.41)$$

where $[X(t), Y(t), Z(t)]^T$ denotes the 3-D position of the moving object in the camera centered 3-D space and $(y_1(t), y_2(t))$ is its projection in the camera frame that can be derived from the corresponding observations on the image plane.

Basic issues of the PDS are the observability, identifiability, and controllability problems that are briefly listed in the following [20, 21, 22, 64]:

- 1) **Observability:** Assuming that A is known, estimate the initial state

$$X_0 = [X(0), Y(0), Z(0)]^T$$

up to a homogeneous line from the output homogeneous observation function $y(t)$.

- 2) **Identifiability:** Assuming that A is unknown and the output $y(t)$ for $t > 0$ is given, identify A to the extent possible with X_0 up to a homogeneous line.

- 3) **Controllability:** Transmit from one state to another in the apparent motion on the image plane.

Since the main focus of this section is on the estimation/identification of motion parameters, the controllability problem is only stated without further discussion.

4.3.1 Background Theory

Using the PDS concept, theoretical analysis has been facilitated to the question about which motion parameters are identifiable and to what extent. Moreover, the 3-D motion estimation problem can be formulated into a parameter identification problem of a PDS and the formulation is independent of the data measured. The data measured from the image plane can be either brightness pattern (image intensity) or features such as points, lines, and curves [21, 91]. In this way, the optical flow and feature-based methods for the 2-D motion estimation is unified as one step in the 3-D motion estimation. In the following, we will review the analysis on the identifiable parameters, but leave the discussion on identification to section 4.3.2.

Consider the following planar surface undergoing a Riccati motion in (4.1):

$$\bar{p}X + \bar{q}Y + \bar{s}Z + 1 = 0. \quad (4.42)$$

We can have

$$\underbrace{\begin{bmatrix} \dot{\bar{p}} \\ \dot{\bar{q}} \\ \dot{\bar{s}} \end{bmatrix}}_{\dot{\bar{P}}} = \underbrace{\begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}}_{-f} - A^T \underbrace{\begin{bmatrix} \bar{p} \\ \bar{q} \\ \bar{s} \end{bmatrix}}_P + B \underbrace{\begin{bmatrix} \bar{p}^2 \\ \bar{p}\bar{q} \\ \bar{p}\bar{s} \\ \bar{q}^2 \\ \bar{q}\bar{s} \\ \bar{s}^2 \end{bmatrix}}_{\bar{P}}, \quad (4.43)$$

where

$$B = \begin{bmatrix} b_1 & b_2 & b_3 & 0 & 0 & 0 \\ 0 & b_1 & 0 & b_2 & b_3 & 0 \\ 0 & 0 & b_1 & 0 & b_2 & b_3 \end{bmatrix}$$

and the detailed derivation is shown in Appendix B.2. Equation (4.43) is called shape dynamics in [21], which is a dynamic system describing the motion and shape parameters.

Further, let

$$X = X_1/W_1, Y = Y_1/W_1, Z = Z_1/W_1, \quad (4.44)$$

and

$$\bar{p} = p/w, \bar{q} = q/w, \bar{s} = s/w. \quad (4.45)$$

We can have

$$\underbrace{\begin{bmatrix} \dot{X}_1 \\ \dot{Y}_1 \\ \dot{Z}_1 \\ \dot{W}_1 \end{bmatrix}}_{\dot{\mathcal{X}}} = \underbrace{\begin{pmatrix} A & \mathbf{b} \\ \mathbf{f}^T & 0 \end{pmatrix}}_{\mathcal{A}} \underbrace{\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ W_1 \end{bmatrix}}_{\mathcal{X}}, \quad (4.46)$$

and [22]

$$\underbrace{\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{s} \\ \dot{w} \end{bmatrix}}_{\dot{\mathcal{P}}} = \underbrace{\begin{pmatrix} -A^T & -\mathbf{f} \\ -\mathbf{b}^T & 0 \end{pmatrix}}_{-\mathcal{A}^T} \underbrace{\begin{bmatrix} p \\ q \\ s \\ w \end{bmatrix}}_{\mathcal{P}}. \quad (4.47)$$

Derivations of the above two equations are similar and are presented in Appendix B.2. Both of equations (4.46) and (4.47) can be taken as the state equation for a PDS, where the outputs are the homogeneous observations of either the state (X, Y, Z) or the shape parameters (p, q, s, w) . Notice that equations (4.46) and (4.47) share a similar structure.

Analysis on the identifiable parameters are based on the following homogeneous observation function

$$\underbrace{\begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}}_{\mathcal{Z}} = \underbrace{\begin{bmatrix} \lambda & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & \lambda & 0 \end{bmatrix}}_{\mathcal{C}} \underbrace{\begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ W_1 \end{bmatrix}}_{\mathcal{X}}, \quad (4.48)$$

which forms a PDS with the motion dynamics in (4.46), expressed in a compact form as

$$\dot{\mathcal{X}}(t) = \mathcal{A}\mathcal{X}(t), \quad \mathcal{Z}(t) = \mathcal{C}\mathcal{X}(t). \quad (4.49)$$

Due to the homogenization in the output observation function, $(\mathcal{A}, \mathcal{C}, \mathcal{X}(0))$ that produces the same output \mathcal{Z} is not unique. The non-uniqueness in $(\mathcal{A}, \mathcal{C}, \mathcal{X}(0))$ is given by the following group of action \mathcal{G} [21]:

$$(P, \mathcal{A}, \mathcal{C}, \mathcal{X}_0) \mapsto (PAP^{-1}, CP^{-1}, P\mathcal{X}_0), \quad (4.50)$$

$$(\lambda, \mathcal{A}, \mathcal{C}, \mathcal{X}_0) \mapsto (\mathcal{A}, \lambda\mathcal{C}, \mathcal{X}_0), \quad (4.51)$$

$$(\theta, \mathcal{A}, \mathcal{C}, \mathcal{X}_0) \mapsto (\theta I + \mathcal{A}, \mathcal{C}, \mathcal{X}_0). \quad (4.52)$$

This is because, using the actions in (4.50)~(4.52), the resultant outputs are:

$$(4.50) : \bar{\mathcal{Z}}(t) = (CP^{-1}) e^{PAP^{-1}t} (P\mathcal{X}_0) = (CP^{-1}) P e^{\mathcal{A}t} P^{-1} (P\mathcal{X}_0) = (\mathcal{C} e^{\mathcal{A}t} \mathcal{X}_0),$$

$$(4.51) : \bar{\mathcal{Z}}(t) = \lambda (\mathcal{C} e^{\mathcal{A}t} \mathcal{X}_0),$$

$$(4.52) : \bar{\mathcal{Z}}(t) = \mathcal{C} e^{\theta I + \mathcal{A}t} \mathcal{X}_0 = \mathcal{C} e^{\theta I} e^{\mathcal{A}t} \mathcal{X}_0 = e^{\theta I} (\mathcal{C} e^{\mathcal{A}t} \mathcal{X}_0),$$

which remain invariant compared with the output of the original PDS in (4.49).

Based on \mathcal{G} , the following theorem has been proposed to analyze the dimensions of the orbit of motion and shape parameters [64].

Theorem: Consider the homogeneous dynamic system (4.49). For a generic choice of the matrix \mathcal{A} and the state vector $\mathcal{X}(0)$, the set of all triplets that produce the same output as given by (4.48) is described by:

$$(PAP^{-1}, \lambda_1 CP^{-1}, P\mathcal{X}(0)), \quad (4.53)$$

where λ_1 is a nonzero real number and P is a nonsingular matrix of the form

$$P^{-1} = \begin{bmatrix} p_{11} & 0 & 0 & 0 \\ 0 & p_{11} & 0 & 0 \\ 0 & 0 & p_{11} & 0 \\ p_{41} & p_{42} & p_{43} & p_{44} \end{bmatrix}. \quad (4.54)$$

Using the above theorem and denoting $\nu = (p_{41}, p_{42}, p_{43})$, the scaling on the matrix \mathcal{A} in (4.49) is given by:

$$\begin{aligned} \mathcal{A} &\mapsto \mathcal{A} + \frac{\mathbf{b}\nu + \nu\mathbf{b}I}{p_{11}}, \\ \mathbf{b} &\mapsto \frac{p_{44}}{p_{11}}\mathbf{b}, \\ \mathbf{f}^T &\mapsto \frac{p_{11}}{p_{44}}\mathbf{f}^T - \frac{\nu\mathcal{A}}{p_{44}} - \frac{\nu\mathbf{b}\nu}{p_{11}p_{44}}, \\ \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} &\mapsto \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \frac{p_{44}/p_{11}}{1 - \frac{p_{41}}{p_{11}}X - \frac{p_{42}}{p_{11}}Y - \frac{p_{43}}{p_{11}}Z}. \end{aligned} \quad (4.55)$$

Note that the above equation denotes a 4-parameter orbit parameterized by

$$(p_{41}/p_{11}, p_{42}/p_{11}, p_{43}/p_{11}, p_{44}/p_{11}).$$

Equation (4.55) is a general description of the parameter orbit for the Riccati motion.

In the following, two special cases are listed:

- 1) When A is skew-symmetric as in (4.2), (4.55) reduces to a 1-parameter orbit:

$$\begin{aligned} A &\longmapsto A, \\ \mathbf{b} &\longmapsto \frac{p_{44}}{p_{11}}\mathbf{b}, \\ \mathbf{f}^T &\longmapsto \frac{p_{11}}{p_{44}}\mathbf{f}^T, \\ [X, Y, Z] &\longmapsto [X, Y, Z] \frac{p_{44}}{p_{11}}. \end{aligned} \tag{4.56}$$

- 2) When A is skew-symmetric as in (4.2) and $\mathbf{b} = -\mathbf{f}$, (4.55) reduces to a sign ambiguity:

$$\begin{aligned} (w_1, w_2, w_3) &\longmapsto (w_1, w_2, w_3), \\ (b_1, b_2, b_3) &\longmapsto \pm(b_1, b_2, b_3), \\ (X, Y, Z) &\longmapsto \pm(X, Y, Z). \end{aligned} \tag{4.57}$$

The above discussion provides theoretical analysis about the identifiable parameters, from which it can be concluded that, using vision information from a single stationary camera, motion parameters can only be identified up to certain orbit, which is a 4-parameter orbit for the Riccati motion and a 1-parameter orbit for the rigid motion. Based on the group action \mathcal{G} in (4.50)~(4.52), canonical forms for the PDS are introduced in [88, 89, 90], where it is shown that parameters in the canonical form can be identified uniquely using an EKF.

In the next section, several nonlinear observers are reviewed for the identification task for the rigid and pure rotational motions.

4.3.2 Applicable Nonlinear Observers for PDS

The PDS theory is a mathematical formulation that reveals known results in computer vision and allows exploitation of much more difficult situations, such as the nonrigid motion estimation problem. After revealing the non-uniqueness nature in identifying the motion parameters, this section reviews several nonlinear observers, applicable to the PDS, that actually carry out the identification task.

Pure Rotational Motion:

For a class of PDS in the form

$$[\dot{X}, \dot{Y}, \dot{Z}]^T = A[X, Y, Z]^T, \quad y = [C x], \quad \text{with } m = n, \quad (4.58)$$

where n are m are the dimensions of the states and outputs as defined in (4.38), the problem of identifying parameters of the above PDS is equal to the problem of identifying parameters of the following Riccati equation: for the PDS in (4.58), we can have

$$\dot{y} = A(y) \theta, \quad (4.59)$$

with

$$A(y) = \begin{bmatrix} \mathbf{v} & & & \mathbf{z}_1 \\ & \ddots & & \vdots \\ & & \ddots & \vdots \\ & & & \mathbf{v} \quad \mathbf{z}_{n-1} \end{bmatrix},$$

$$\mathbf{v} = [y_1, \dots, y_{n-1}, 1], \quad \mathbf{z}_j = -[y_j y_1, \dots, y_j y_{n-1}].$$

For example, when $n = m = 3$, by letting $y_1 = X/Z$, $y_2 = Y/Z$, we have the following 2-D Riccati equation

$$\begin{cases} \dot{y}_1 = a_{12}y_2 - y_1(a_{31}y_1 + a_{32}y_2) + y_1(a_{11} - a_{33}) + a_{13} \\ \dot{y}_2 = a_{21}y_1 - y_2(a_{31}y_1 + a_{32}y_2) + y_2(a_{22} - a_{33}) + a_{23} \end{cases}, \quad (4.60)$$

which can be written in the form of (4.59) with

$$A(y) = \begin{bmatrix} y_1 & y_2 & 1 & 0 & 0 & 0 & -y_1^2 & -y_1 y_2 \\ 0 & 0 & 0 & y_1 & y_2 & 1 & -y_1 y_2 & -y_2^2 \end{bmatrix}, \quad (4.61)$$

$$\theta = [a_{11} - a_{33}, a_{12}, a_{13}, a_{21}, a_{22} - a_{33}, a_{23}, a_{31}, a_{32}]^T.$$

It is observed from the θ in the above equation that the parameters a_{11} , a_{22} , and a_{33} can only be identified to the extent $(a_{11} - a_{33})$ and $(a_{22} - a_{33})$.

For the Riccati equation (4.59), when the motion parameters are constant, the following full-state observer can be used to estimate the states of the PDS [20, 85, 92]:

$$\begin{aligned}\dot{\hat{y}} &= H(\hat{y} - y) + A(y)\hat{\theta}, \\ \dot{\hat{\theta}} &= -A^T P(\hat{y} - y),\end{aligned}\tag{4.62}$$

where H is any Hurwitz matrix whose eigenvalues are in the open left half of the complex plane, P is the positive definite solution of $H^T P + PH = -Q$, and Q is a positive definite symmetric matrix. When the parameters are not constant, but varying according to a law that is a linear function of the parameters and a possible nonlinear function of the states of the system, a full-state observer constructed in [20] can be used to estimate θ .

Persistent Excitation (PE), a concept in the theory of identification and adaptive control, turns out to be the sufficient and necessary condition for the convergence of the identification error of the observer in (4.62) to zero. For the case with time varying motion parameters, the applied observer in [20] is also convergent under a suitable condition that is given by a generalization of the PE condition [20, 85].

Rigid Motion:

Consider the planar surface (4.42) undergoing a Riccati motion in (4.1) under the perspective projection, where the outputs are taken as

$$y_1 = X/Z, \quad y_2 = Y/Z.\tag{4.63}$$

The optical flow dynamics on the image plane takes the form

$$\begin{cases} \dot{y}_1 = d_1 + d_3 y_1 + d_4 y_2 + d_7 y_1^2 + d_8 y_1 y_2 \\ \dot{y}_2 = d_2 + d_5 y_1 + d_6 y_2 + d_8 y_2^2 + d_7 x_1 y_2 \end{cases},\tag{4.64}$$

where

$$\begin{cases} d_1 = a_{13} - b_1 \bar{s} \\ d_2 = a_{23} - b_2 \bar{s} \\ d_3 = (a_{11} - a_{33}) - b_1 \bar{p} + b_3 \bar{s} \\ d_4 = a_{12} - b_1 \bar{q} \\ d_5 = a_{21} - b_2 \bar{p} \\ d_6 = (a_{22} - a_{33}) - b_2 \bar{q} + b_3 \bar{s} \\ d_7 = b_3 \bar{p} - a_{31} \\ d_8 = b_3 \bar{q} - a_{32} \end{cases},\tag{4.65}$$

and the parameters

$$\mathbf{d} = (d_1, d_2, \dots, d_8)^T \quad (4.66)$$

are called the essential parameters of the optical flow dynamics [21]. Detailed derivation of the optical flow dynamics in (4.64) is shown in Appendix B.2.

The identification task for the rigid motion can be performed via the above essential parameters. Let us rewrite the optical flow dynamics (4.64) as

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & y_1 & y_2 & 0 & 0 & \frac{y_1^2}{f} & \frac{y_1 y_2}{f} \\ 0 & 1 & 0 & 0 & y_1 & y_2 & \frac{y_1 y_2}{f} & \frac{y_2^2}{f} \end{bmatrix}}_{w(y)} \mathbf{d}. \quad (4.67)$$

Combined with $\dot{\mathbf{d}} = \phi(\mathbf{d})$, where $\phi(\cdot)$ is a function of \mathbf{d} , we have [86, 93]

$$\dot{y} = w(y) \mathbf{d}, \quad \dot{\mathbf{d}} = \phi(\mathbf{d}). \quad (4.68)$$

The above system fits into the form of the following nonlinear system

$$\begin{aligned} \dot{x}_1 &= w^T(x_1, u)x_2 + \phi(x_1, u), \\ \dot{x}_2 &= g(x_1, x_2, u), \\ y &= x_1, \end{aligned} \quad (4.69)$$

where the matrix $w^T(x_1, u)$ and the vector $g(x_1, x_2, u)$ are nonlinear functions of their arguments. For the class of nonlinear systems in (4.69), a discontinuous nonlinear state estimator, called the Identifier Based Observer (IBO), can be applied for the state estimation [23, 93]. Applying the IBO to (4.68), \hat{y} , $\hat{\mathbf{d}}$ can be estimated.⁷ However, the motion parameters $a_{i,j}$ for $i, j = 1, 2, 3$ and b_i for $i = 1, 2, 3$ can not be extracted from $\hat{\mathbf{d}}$ exactly for a general motion. In the special case of a rigid motion, the essential parameters of the optical flow dynamics (4.65) become a set of eight nonlinear equations with eight parameters, commonly known as the *recovery equation*. The recovery equation has two solutions and the use of shape dynamics results in the recovery of the correct alternative [21, 94].

⁷Though the correspondence problem between a set of feature points in the image sequence has been assumed to be established, the estimation of \hat{y} in (4.67), together with the states in (4.35) estimated by the IEKF approach, helps to track features over time to reduce the search region.

Rigid Motion Estimation with Active Vision System:

In the last few years, the observability of perspective linear systems has been systematically studied in the literature and it is noted that without input, it is never possible to recover the norm of the state. However, with a proper input, the state estimation can be made possible [87, 95]. An optimal state observer is proposed in [87] for systems with multiple homogeneous outputs of the following form

$$\begin{aligned}\dot{x} &= A(u)x + b(u) + G(u)d, \\ \alpha_j y_j &= C_j(u)x + d_j(u) + n_j, \quad j = \{1, 2, \dots, m\},\end{aligned}\tag{4.70}$$

where d denotes the disturbance and n_j denotes the measurement noise for the j -th observation function. By assuming that A, G, C_j are constant matrices (denoted by A, G, C_j) and letting

$$b(u) \triangleq Bu, \quad d_j(u) \triangleq D_j u,$$

equation (4.70) reduces to

$$\begin{aligned}\dot{x} &= Ax + Bu + Gd, \\ \alpha_j y_j &= C_j x + D_j u + n_j, \quad j = \{1, 2, \dots, m\},\end{aligned}\tag{4.71}$$

which is the PDS in (4.37) with disturbance and measurement noise. The state estimation problem of the system in (4.70) is formulated in a deterministic setting by searching for the value of the state that is most compatible with the dynamics, in the sense that it requires the least amount of noise to explain the measured output [87]. Under an appropriate observability assumption that depends only on the motion of the camera, the state observer proposed in [87] is globally convergent to the correct position and orientation in the absence of noise. When there is noise, the magnitude of the estimation error is essentially proportional to the magnitude of the noise [87, 96].

The work in [87, 96] is taken one step further in [97] to incorporate a set of quadratic constraints for the states. The main reason to consider state constraints is to take into account that some elements of the state are known to lie in a given manifold. For example, in the rigid motion case, part of the state of the perspective system is a rotation matrix that lies in $SO(3)$. The estimator proposed in [87, 97] requires the camera's linear and

angular velocities, which can be reasonably assumed known in applications such as robot navigation, where the motion of the camera is determined by the applied control signals.

Using the observer proposed in [87, 97], rigid motion estimation is obtained in a setup where a camera attached to the body frame seeks several fixed feature points in the inertial frame, which serves as a scheme for robot localization. The states estimated are related to the nine rotational and three translational components.

4.3.3 Range Identification with Known Motion Parameters

Besides the identification problem to estimate the unknown 3-D motion parameters, research is also oriented to the range identification problem with known motion parameters. For $y(t) = [y_1(t), y_2(t), y_3(t)]^T = [X(t)/Z(t), Y(t)/Z(t), 1/Z(t)]^T$, the derivative of $y(t)$ under a general Riccati motion is:

$$\begin{cases} \dot{y}_1(t) = a_{13} + (a_{11} - a_{33})y_1 + a_{12}y_2 - a_{31}y_1^2 - a_{32}y_1y_2 + (b_1 - b_3y_1)y_3 \\ \dot{y}_2(t) = a_{23} + a_{21}y_1 + (a_{22} - a_{33})y_2 - a_{31}y_1y_2 - a_{32}y_2^2 + (b_2 - b_3y_2)y_3 \\ \dot{y}_3(t) = -(a_{31}y_1 + a_{32}y_2 + a_{33})y_3 - b_3y_3^2 - (f_1y_1 + f_2y_2 + f_3) \end{cases} \quad (4.72)$$

The range identification (depth estimation) problem is to estimate $Z(t)$, or its inverse $y_3(t)$, assuming $y_1(t)$ and $y_2(t)$ are available and the motion parameters $a_{i,j}(t)$, for $i, j = 1, 2, 3$ and $b_i(t)$, for $i = 1, 2, 3$ are known.

For the depth estimation problem, some nonlinear observers have been proposed in the literature [23, 24, 25]. Preliminary comparisons of some of the nonlinear observers will be presented in section 4.4. The challenge in the 3-D motion estimation lies in the identification of unknown motion parameters. If the motion parameters can be identified exactly, depth information can be recovered using the nonlinear observers reviewed in this section.

4.4 Comparative Study of Existing Perspective Observers

In the 3-D motion estimation reviewed in section 4.3, there are basically two sub-categories of identification problems. One category is to estimate the parameters of the

motion dynamics of the moving object. The other is to recover the depth information assuming that the motion parameters are already known. The solutions to the first class of problems can be resolved via algorithms such as nonlinear optimization formulations [69], linear least squares/total least squares approximations [18], nonlinear observers [20, 23, 83], and the application of epipolar constraints [16]. Using image sequences from a single camera, it is well-known that for rigid motion, the rotation parameters can be estimated uniquely, but the translational parameters can only be estimated up to a depth ambiguity when the physical size of the target is not available. Though multi-view cameras, such as stereo cameras and multi-cameras operating asynchronously, can be used to recover the depth information, in this work, we stick to the single camera setup and focus on nonlinear observer techniques that have been applied to this particular nonlinear system, the perspective dynamic system (PDS). In addition to the affine motion in (4.40), the more general Riccati motion (4.1) has been discussed in [24, 25], which introduces no more difficulty in the observer design.

4.4.1 Perspective Nonlinear Observers

The nonlinear observers that have been applied to the PDS for the range estimation problem include:

- 1) The identifier-based observer (IBO) proposed in [23] that is motivated from the adaptive control theory and is suitable for the class of nonlinear systems in (4.69).
- 2) The state observer (SMO) in [24], which is a combination of the sliding mode control method (SMC), the adaptive method, and the discontinuous observer techniques.
- 3) The Range Identification Observer (RIO) in [25], which facilitates Lyapunov-based analysis and is motivated from the recent disturbance observer results [98].

The above three observers can all be applied to the Riccati motion and extended to n -dimensional cases. The IBO and SMO are similar in that they try to compensate the

bounded items $(b_i - b_3 y_i) e_3$ for $i = 1, 2$ to assure that the errors $|e_1|$ and $|e_2|$, which are defined to be

$$e_1 = y_1 - \hat{y}_1, \quad e_2 = y_2 - \hat{y}_2, \quad (4.73)$$

are very small. The RIO is less complex compared with the IBO and SMO by not investigating the structure of $(b_i - b_3 y_i) e_3$. For the general Riccati motion, the derivative of $y(t)$ is in (4.72). It is based on the above equation that the three observers are formulated.

4.4.1.1 IBO

Obviously, (4.72) is in the form of (4.69). After applying the observer in [23], the IBO for the PDS takes specifically the following form:

$$\begin{cases} \begin{bmatrix} \dot{\hat{y}}_1 \\ \dot{\hat{y}}_2 \end{bmatrix} = GH \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} + \begin{bmatrix} b_1 - b_3 y_1 \\ b_2 - b_3 y_2 \end{bmatrix} \hat{y}_3 + \begin{bmatrix} a_{13} + (a_{11} - a_{33})y_1 + a_{12}y_2 \\ a_{23} + a_{21}y_1 + (a_{22} - a_{33})y_2 \end{bmatrix} - \begin{bmatrix} a_{31}y_1^2 + a_{32}y_1y_2 \\ a_{31}y_1y_2 + a_{32}y_2^2 \end{bmatrix} \\ \dot{\hat{y}}_3 = -G^2[b_1 - b_3 y_1, b_2 - b_3 y_2]P \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} - (a_{31}y_1 + a_{32}y_2 + a_{33})\hat{y}_3 - b_3\hat{y}_3^2 - (\bar{f}_1 y_1 + \bar{f}_2 y_2 + \bar{f}_3) , \\ \hat{y}_3(t_i^+) = M \frac{\hat{y}_3(t_i^-)}{\|\hat{y}_3(t_i^-)\|} \end{cases} \quad (4.74)$$

where the sequences of t_i are defined as

$$t_i = \min \{t : t > t_{i-1} \text{ and } \|\hat{y}_3(t)\| \geq \gamma M\}. \quad (4.75)$$

and the matrix P is a positive definite solution of the Lyapunov equation $H^T P + P H = -Q$. In (4.75), scalar M is an assumed upper bound for the state estimate $\|\hat{y}_3(t)\|$ and γ is a fixed constant scalar with $\gamma > 1$. The G in (4.74) is a constant scalar gain. Notation wise, all the observer parameters correspond to their specific observers, i.e., G or γ do not have a global meaning.

The observability assumption of the above IBO is detailed as [23]:

Observability Condition of IBO: The regressor matrix $w^T(x_1, u)$ is piecewise smooth and uniformly bounded together with its first time derivative and there exist positive constants r, β such that

$$\int_t^{t+\rho} w(x_1(\tau), u(\tau)) w^T(x_1(\tau), u(\tau)) d\tau > \rho\beta I \quad (4.76)$$

for all $0 < \rho < r$ with I being the identity matrix.

The IBO is recursive and guaranteed to converge in an arbitrarily large but bounded set of initial conditions. Further, the convergence is exponential, though the rate of convergence depends on the choice of initial conditions.

4.4.1.2 SMO

For the perspective dynamic system (4.40), (4.41) and defining $e_3 = y_3 - \hat{y}_3$, the SMO is formulated as [24]:

$$\begin{cases} \begin{bmatrix} \dot{\hat{y}}_1 \\ \dot{\hat{y}}_2 \end{bmatrix} = \begin{bmatrix} \hat{\lambda}_1(t) e_1 / (|e_1| + \delta_1) \\ \hat{\lambda}_2(t) e_2 / (|e_2| + \delta_2) \end{bmatrix} + \begin{bmatrix} b_1 - b_3 y_1 \\ b_2 - b_3 y_2 \end{bmatrix} \hat{y}_3 + \begin{bmatrix} a_{13} + (a_{11} - a_{33})y_1 + a_{12}y_2 \\ a_{23} + a_{21}y_1 + (a_{22} - a_{33})y_2 \end{bmatrix} - \begin{bmatrix} a_{31}y_1^2 + a_{32}y_1y_2 \\ a_{31}y_1y_2 + a_{32}y_2^2 \end{bmatrix} \\ \dot{\hat{y}}_3 = \alpha [b_1 - b_3 y_1, b_2 - b_3 y_2] \begin{bmatrix} \hat{\lambda}_1(t) e_1 / (|e_1| + \delta_1) \\ \hat{\lambda}_2(t) e_2 / (|e_2| + \delta_2) \end{bmatrix} - (a_{31}y_1 + a_{32}y_2 + a_{33})\hat{y}_3 - b_3\hat{y}_3^2 - (\bar{f}_1 y_1 + \bar{f}_2 y_2 + \bar{f}_3) \\ \hat{y}_3(t_i^+) = M \frac{\hat{y}_3(t_i^-)}{\|\hat{y}_3(t_i^-)\|} \end{cases} \quad (4.77)$$

where $\hat{\lambda}_1(t)$ and $\hat{\lambda}_2(t)$ are adaptively updated by:

$$\begin{aligned} \dot{\hat{\lambda}}_1(t) &= \begin{cases} 2\alpha_1 |e_1|, & \text{if } |e_1| > 2\delta_1 \\ 0, & \text{otherwise} \end{cases}, \\ \dot{\hat{\lambda}}_2(t) &= \begin{cases} 2\alpha_2 |e_2|, & \text{if } |e_2| > 2\delta_2 \\ 0, & \text{otherwise} \end{cases}. \end{aligned} \quad (4.78)$$

The design parameters α , α_1 , α_2 , δ_1 , and δ_2 in (4.77) and (4.78) are all positive scalar constants.

The SMO is constructed based on the following three assumptions: 1) the parameters $a_{ij}(t)$, $i, j = 1, 2, 3$ and $b_i(t)$, $i = 1, 2, 3$ are known bounded functions of time t ; $b_i(t)$ are piecewise differentiable and have bounded derivatives, 2) $x_3(t) > 0$, and 3) the output $y_1(t)$ and $y_2(t)$ are bounded. The second and the third assumptions are reasonable by referring to the practical system.

Observability Condition of SMO: Suppose that there exist positive constants β, ρ such that

$$\int_t^{t+\rho} ((b_1 - b_3 y_1(\tau))^2 + (b_2 - b_3 y_2(\tau))^2) d\tau \geq \beta \quad (4.79)$$

for all $t \geq 0$. Then, for given $\delta_1 > 0$ and $\delta_2 > 0$, there exist $T_0 \geq 0$ and a function $\epsilon(u, v) > 0$ such that $|e_3(t)| < \epsilon(\delta_1, \delta_2)$ for all $t \geq T_0$.

The key idea of $e_{1,2}(t)$ decreasing to a small regions around 0 is based on the “equivalent control method.” That is, for a system $\dot{z}(t) = v(t) - N \text{sign}(z(t))$ with $|v(t)| < N$, $z(t) \rightarrow 0$ as $t \rightarrow \infty$. Further, for the SMO, $e_3^2(t)$ is proved to decrease exponentially if it is not very small.

4.4.1.3 RIO

The RIO in [25] is relatively simple in the observer design and it does not utilize the specific structure of $(b_i - b_3 y_i) e_3$ for $i = 1, 2$. Let

$$f_1 \doteq (b_1 - b_3 y_1) y_3, \quad f_2 \doteq (b_2 - b_3 y_2) y_3. \quad (4.80)$$

Then $\dot{e}_1 = \dot{y}_1 - \dot{\hat{y}}_1 = f_1 - \dot{\hat{f}}_1$, $\dot{e}_2 = \dot{y}_2 - \dot{\hat{y}}_2 = f_2 - \dot{\hat{f}}_2$. The estimates \hat{f}_1 and \hat{f}_2 are designed to be:

$$\begin{cases} \dot{\hat{f}}_1 = -(k_{s1} + \alpha_1) \hat{f}_1 + \gamma_1 \text{sign}(e_1) + \alpha_1 k_{s1} e_1 \\ \dot{\hat{f}}_2 = -(k_{s2} + \alpha_2) \hat{f}_2 + \gamma_2 \text{sign}(e_2) + \alpha_2 k_{s2} e_2 \end{cases}, \quad (4.81)$$

where k_{si} , α_i , and γ_i for $i = 1, 2$ are all positive scalars. Using the estimates \hat{f}_1 and \hat{f}_2 , \hat{y}_3 can be calculated by:

$$\hat{y}_3^2 = \frac{\hat{f}_1^2 + \hat{f}_2^2}{(b_1 - b_3 y_1)^2 + (b_2 - b_3 y_2)^2}. \quad (4.82)$$

Based on the structure of (4.82), the following observability condition must be satisfied.

Observability Condition of RIO:

$$(b_1 - b_3 y_1)^2 + (b_2 - b_3 y_2)^2 > 0 \quad \text{for all } t. \quad (4.83)$$

The RIO facilitates a Lyapunov-based analysis that is less complex than the sliding mode based analysis, such as the SMO. Using the RIO, the 3-D task space coordinates of the feature point can be identified asymptotically.

Remark 4.4.1 To avoid chattering caused by the discontinuous terms $\text{sign}(e_i)$ for $i = 1, 2$ in (4.81), $\text{sign}(e_i)$ are modified as $e_i/(|e_i| + \delta_i)$ in our simulation to compare with the other two observers.

Table 4.1: Design Parameters of Existing Perspective Observers

	Parameter	Description
IBO	M	Upper bound of the state estimate \hat{y}_3
	G	Determine the converging speed
	(H, P)	Matrices that satisfy Lyapunov equation
SMO	M	Upper bound of the state estimate \hat{y}_3
	α_i	Determine the speed to update $\hat{\lambda}_i$
	δ_i	Determine the estimation precision
	α	Determine the converging speed
RIO	k_{si}, α_i	Related to the transient response
	γ_i	Determine the converging speed
	δ_i	Related to the smoothness and precision

4.4.1.4 Observer Parameters

The functions of the design parameters of the three observers are summarized in table 4.1 with $i = 1, 2$.

4.4.2 Simulation Results and Discussions

In this section, our simulation results for the three nonlinear observers (IBO, SMO, RIO) are presented. In the original papers of IBO and SMO [23, 24], simulation results are presented for the affine motion ($\mathbf{f} = 0$ in (4.72)). The simulations performed here are for both the affine and the Riccati motions.

Two specific examples are performed in [24] using Matlab for the SMO observer, where the two examples are:

1) Example 1: Time invariant motion dynamics [24]

$$\begin{bmatrix} \dot{X}(t) \\ \dot{Y}(t) \\ \dot{Z}(t) \end{bmatrix} = \begin{bmatrix} -0.2 & 0.4 & -0.6 \\ 0.1 & -0.2 & 0.3 \\ 0.3 & -0.4 & 0.4 \end{bmatrix} \begin{bmatrix} X(t) \\ Y(t) \\ Z(t) \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0.25 \\ 0.3 \end{bmatrix}, \quad (4.84)$$

$$[X(0) \ Y(0) \ Z(0)]^T = [1 \ 1.5 \ 2.5]^T.$$

2) **Example 2:** Time varying motion dynamics [24]

$$\begin{bmatrix} \dot{X}(t) \\ \dot{Y}(t) \\ \dot{Z}(t) \end{bmatrix} = \begin{bmatrix} 0 & -2\pi & 0 \\ 2\pi & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} X(t) \\ Y(t) \\ Z(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 2\pi \cos(2\pi t) \end{bmatrix}, \quad (4.85)$$

$$[X(0) \ Y(0) \ Z(0)]^T = [1 \ 1 \ 2]^T.$$

In our simulation, the above two examples are used along with

$$\mathbf{f} = [0.001, 0.001, -0.001]^T \quad (4.86)$$

for an extension to the estimation of the Riccati motion. The reason for choosing this small \mathbf{f} is the lack of guidance for selecting \mathbf{f} such that the outputs are bounded. For all the observer simulations, the initial conditions of $[\hat{y}_1(0), \hat{y}_2(0), \hat{y}_3(0)]^T$ are chosen to be $[y_1(0), y_2(0), 1]^T$. Further, the observer design parameters listed in table 4.1 are tuned such that all the observers are compared based on a similar converging speed in the absence of noise along with acceptable performance in the presence of noise.

The parameters chosen for the three observers are:

- IBO: $M = 10$, $G = 10$, $\gamma = 1$, $H = I_{2 \times 2}$, $P = -I_{2 \times 2}/2$.
- SMO: $M = 10$, $\alpha = 5$, $\alpha_1 = \alpha_2 = 10$, $\delta_1 = \delta_2 = 0.2$, $\hat{\lambda}_1(0) = \hat{\lambda}_2(0) = 1$.
- RIO: $\gamma_1 = \gamma_2 = 30$, $k_{s1} = k_{s2} = 5$, $\alpha_1 = \alpha_2 = 5$.

Figures 4.1 and 4.2 present the simulations results of the three observers for the above two examples with $\mathbf{f} = 0$ and the \mathbf{f} in (4.86). From fig. 4.1, it is observed that with $\mathbf{f} = 0$ and $\mathbf{f} \neq 0$, the estimation errors e_3 for the affine and Riccati motions decrease with a similar pattern. This phenomenon is also true for the second example, as can be seen in fig. 4.2. The reason for this similarity may due to the small values of \mathbf{f} . Furthermore, a nonzero \mathbf{f} in the Riccati motion only introduces one more term $-(f_1 y_1 + f_2 y_2 + f_3)$ in (4.72). Since the outputs y_1 and y_2 are available signals, theoretically, there should not be big differences as long as the states are bounded.

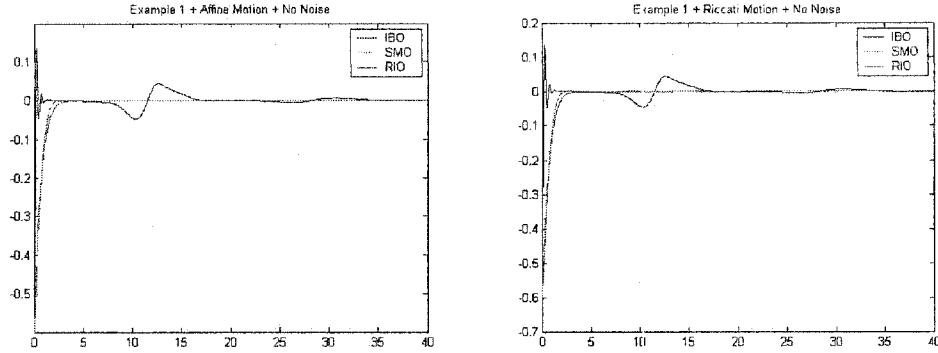


Fig. 4.1: Simulation results of e_3 for the three observers for Example₁.

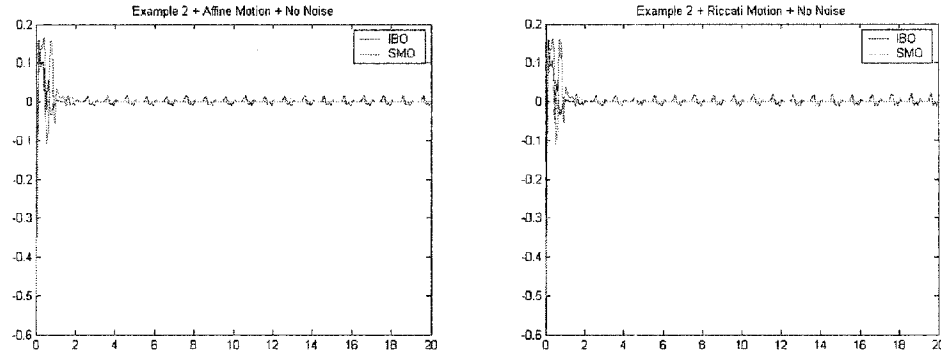


Fig. 4.2: Simulation results of e_3 for the three observers for Example₂ (RIO can not be applied to this example, as can be seen from fig. 4.3).

In fig. 4.2, the simulation results of RIO for the Example₂ are not plotted because the error e_3 explodes at the time instances when $b_3(t) = 0$ periodically, as can be seen from fig. 4.3. This is explained by the observability condition of RIO, which does not allow $(b_1 - b_3y_1)^2 + (b_2 - b_3y_2)^2$ to be zero at any time.

Remark 4.4.2 *The simulation results of IBO and SMO are not exactly the same as those presented in [24] when exactly the same motion dynamics, the same initial conditions, and the same observer parameters are applied. In [24], the ideal plant is violated with 1% noise. However, in our simulation, it is either ideal or with certain amount of uniform noise.*

Figures 4.4 and 4.5 show the observer performances in the presence of uniform noise to the order of 10^{-3} and 10^{-2} , respectively. It is observed that when the noise level reaches

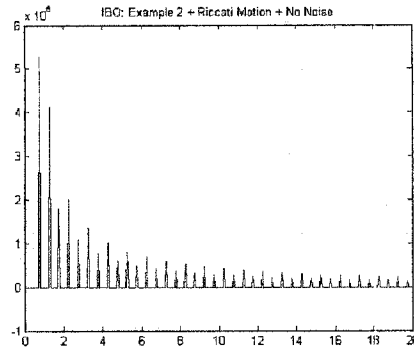


Fig. 4.3: Simulation results of e_3 for Example₂ using RIO.

10^{-2} , all the three observers could not give satisfactory estimates, where the RIO appears to be even worse.

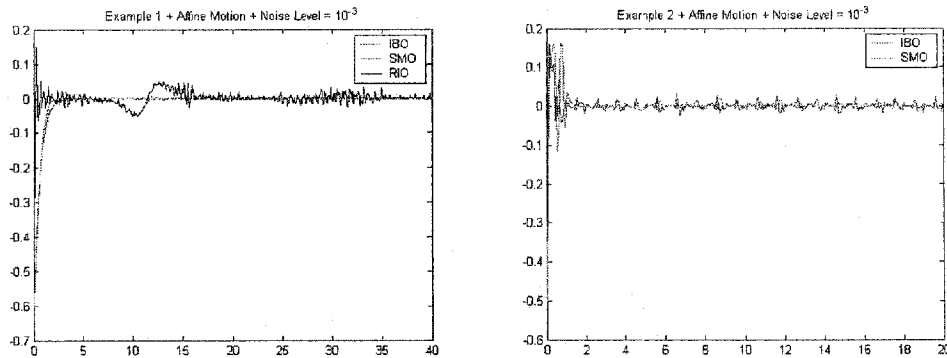


Fig. 4.4: Examples_{1,2} with noise level = 10^{-3} .

In our study, we found that the RIO, which is simpler in the structure, does not perform as well as the other two observers studied in this dissertation. First, the RIO can not be applied to the Example₂. Second, a careful examination of fig. 4.1 for the Example₁ shows that e_3 of the RIO fluctuate around the 12-th second, which is due to the fast changing of f_1 and f_2 and the slow catching of \hat{f}_1 and \hat{f}_2 designed in (4.81), as can be seen from fig. 4.6. Finally, in the presence of noise, RIO experiences more obvious performance degradation compared with the other two observers.

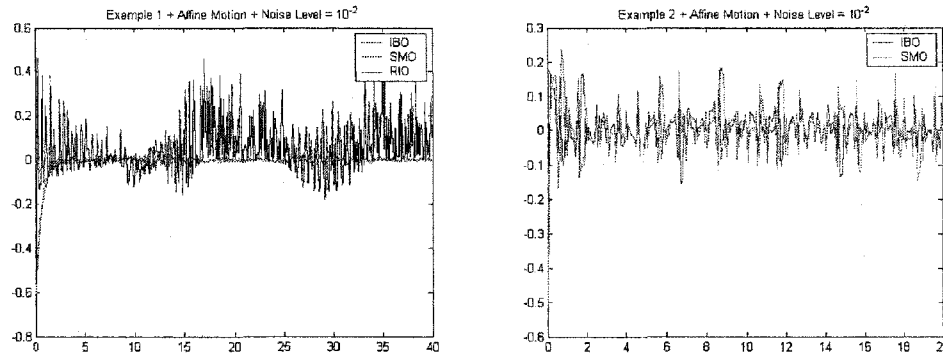


Fig. 4.5: Examples_{1,2} with noise level = 10^{-2} .

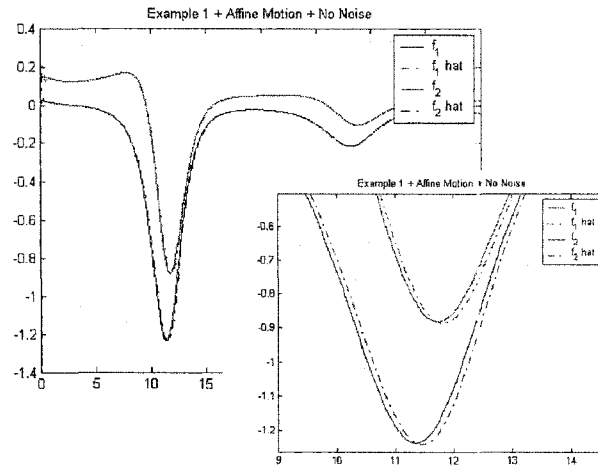


Fig. 4.6: $f_{1,2}$ and $\hat{f}_{1,2}$ of IBO for Example₁.

4.5 Range Identification for Perspective Dynamic Systems Using Linear Approximation

For a general nonlinear system, observer design methods fall into the following categories:

- 1) Assuming bounded or local/global Lipschitz conditions.
- 2) Lyapunov-based design.
- 3) Linear approximation-based technique.

4) Via transformations [99].

The IBO and SMO observers discussed in section 4.4 belong to the first category, while the RIO belongs to the second. In this section, perspective nonlinear observer design is pursued in the linear approximation category, where the nonlinear system is approximated by a sequence of Linear Time-Varying (LTV) subsystems. Observer design of the original nonlinear system reduces to the observer design of this sequence of LTV subsystems, allowing the use of well-known linear techniques [100, 101]. The approach presented in this section is an original contribution of this dissertation.

The section is organized as follows. Section 4.5.1 introduces the linear approximation technique and applies this technique to the range identification problem. For each approximation subsystem of the original PDS, observer design is carried out using the LTV observer in [102], whose detailed procedures are presented in section 4.5.2. Section 4.6.2 presents simulation results of the linear approximation-based observer with comparisons among several other perspective nonlinear observers for the range identification problem. Finally, section 4.7.5 concludes the section.

4.5.1 Linear Approximation Technique

A recently developed method for replacing a nonlinear system by a sequence of LTV approximations is briefly described, where the sequence of LTV subsystems converges on any compact time interval, uniformly in time, to the solution of the original nonlinear system under the condition of local Lipschitz.

4.5.1.1 Introduction to Linear Approximation Technique

Consider the following nonlinear system

$$\begin{aligned}\dot{x}(t) &= A(x)x(t) + B(x)u(t), \\ y(t) &= C(x)x(t),\end{aligned}\tag{4.87}$$

with $x(0) = x_0 \in \mathbb{R}^n$. The sequence of LTV approximations is introduced as [100, 101]:

$$\begin{cases} \dot{x}^{[0]}(t) = A(x_0)x^{[0]}(t) + B(x_0)u^{[0]}(t), \\ \dot{x}^{[i]}(t) = A(x^{[i-1]}(t))x^{[i]}(t) + B(x^{[i-1]}(t))u^{[i]}(t), \\ x^{[0]}(t) = x_0, \text{ for } i = 0, \quad x^{[i]}(t) = x_0, \text{ for } i \geq 1. \end{cases}\tag{4.88}$$

Using the above linear approximations, global convergence is guaranteed in the sense that if the solution of the original nonlinear system exists and is bounded in the interval $[0, \tau] \subseteq \mathbb{R}$, then the sequence of LTV approximations converges uniformly on $[0, \tau]$ to the solution of the original nonlinear system as $i \rightarrow \infty$ [101].

4.5.1.2 Linearization of PDS

From now on, we use the vector variable x to denote the state vector in (4.72) to be consistent with the general nonlinear system (4.87). After changing the variables, the dynamics (4.72) can be rewritten in the form of (4.87) as:

$$\begin{aligned} \dot{x}(t) &= \underbrace{\begin{bmatrix} a_{11} - Q_1(t) & a_{12} & b_1 - b_3 x_1 \\ a_{21} & a_{22} - Q_1(t) & b_2 - b_3 x_2 \\ 0 & 0 & -Q_1(t) - b_3 x_3 \end{bmatrix}}_{A(x)} x(t) + \underbrace{\begin{bmatrix} a_{13} & a_{23} & 0 \end{bmatrix}^T}_{B^T(x)}, \\ y(t) &= \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}}_{C(x)} x(t), \quad x(0) = x_0, \end{aligned} \quad (4.89)$$

with

$$Q_1(t) = a_{33} + a_{31}x_1 + a_{32}x_2. \quad (4.90)$$

Applying the linear approximation idea described in section 4.5.1.1, the nonlinear system (4.89) can be replaced by a sequence of linear approximations specifically of the form:

$$\begin{aligned} \dot{x}^{[i]}(t) &= \underbrace{\begin{bmatrix} a_{11} - Q_1^{[i-1]} & a_{12} & b_1 - b_3 x_1^{[i-1]} \\ a_{21} & a_{22} - Q_1^{[i-1]} & b_2 - b_3 x_2^{[i-1]} \\ 0 & 0 & -Q_1^{[i-1]} - b_3 x_3^{[i-1]} \end{bmatrix}}_{A(x^{[i-1]}(t))} x^{[i]}(t) + \underbrace{\begin{bmatrix} a_{13} & a_{23} & 0 \end{bmatrix}^T}_{B^T(t)}, \\ y^{[i]}(t) &= \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}}_{C(t)} x^{[i]}(t), \quad x^{[i]}(0) = x_0, \end{aligned} \quad (4.91)$$

where $x^{[-1]}(t) = x_0$ for $i = 0$, $Q_1^{[i]}(t) = a_{33} + a_{31}x_1^{[i]} + a_{32}x_2^{[i]}$, and $B(t)$ and $C(t)$ are constant matrices.

4.5.2 Observer Design for LTV Systems

Using the sequence of LTV approximations in (4.88), the observer design of the original nonlinear system can be converted to the observer design of this sequence of LTV subsystems. In [100], the state estimation of each LTV subsystem is performed using the algorithm in [102] due to its simplicity in implementation. For the range identification problem in concern, the LTV observer in [102] is temporarily adopted to illustrate the idea in this section. Other LTV observer techniques can also be applied.

4.5.3 LTV Observer for Each Approximation Subsystem

Consider (4.91) and assume temporarily that $x^{[i-1]}(t)$ and $y^{[i]}(t)$ are known signals for the i -th approximation subsystem. Each approximation subsystem can be regarded as a stand alone LTV system in the form

$$\begin{aligned}\dot{x}(t) &= A(t)x(t) + B(t)u(t), \\ y(t) &= C(t)x(t),\end{aligned}\tag{4.92}$$

where $A(t) = A(x^{[i-1]}(t))$ is available since $x^{[i-1]}(t)$ is assumed known. $B(t)$ and $C(t)$ are constant matrices and $u(t) = 1$. Based on these assumptions, the following sequence of LTV observers can be constructed as in [100] using the observer design algorithm of [102] for the sequence of LTV subsystems:

$$\begin{cases} \dot{\hat{x}}^{[0]}(t) = \bar{F}\hat{x}^{[0]}(t) + \bar{G}(x_0)y^{[0]}(t) + \bar{B}(x_0)u^{[0]}(t), \\ \dot{\hat{x}}^{[i]}(t) = \bar{F}\hat{x}^{[i]}(t) + \bar{G}(x^{[i-1]})y^{[i]}(t) + \bar{B}(x^{[i-1]})u^{[i]}(t), \\ x^{[0]}(0) = x_0, \text{ for } i = 0, \quad x^{[i]}(0) = x_0, \text{ for } i \geq 1, \end{cases}\tag{4.93}$$

with

$$\hat{x}^{[i]}(t) = P(t)\hat{\hat{x}}^{[i]}(t), \quad \text{for } i \geq 0,\tag{4.94}$$

where $P(t)$ is a Lyapunov transformation to be designed. The matrices \bar{B} , \bar{G} , and \bar{F} can be calibrated from $P(t)$ as to be shown in equations (4.100), (4.103), and (4.105).

In the following, detailed LTV observer design procedures in [102] are given for each LTV subsystem of a PDS. The readers are strongly recommended to refer [102] for notations and algorithms.

Using the same variables as in [102], for the i -th subsystem, the observability matrix is

$$N(t) = [N_1(t), N_2(t)]^T, \quad (4.95)$$

where

$$\begin{aligned} N_1(t) &= C(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \\ N_2(t) &= C(t)A(t) = \begin{bmatrix} a_{11} - Q_1^{[i-1]} & a_{12} & b_1 - b_3 x_1^{[i-1]} \\ a_{21} & a_{22} - Q_1^{[i-1]} & b_2 - b_3 x_2^{[i-1]} \end{bmatrix}. \end{aligned} \quad (4.96)$$

If $N(t)$ is of full rank and if one row of $N_2(t)$, say the first row, is always linearly independent with the two rows of $N_1(t)$ for all $t \in [0, \infty]$, we can form a matrix $\bar{N}(t)$

$$\bar{N}(t) = \begin{bmatrix} 1 & 0 & 0 \\ a_{11} - Q_1^{[i-1]}(t) & a_{12} & b_1 - b_3 x_1^{[i-1]} \\ 0 & 1 & 0 \end{bmatrix}, \quad (4.97)$$

whose inverse exists with the following form:

$$\bar{N}^{-1}(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ \frac{Q_1^{[i-1]}(t) - a_{11}}{b_1 - b_3 x_1^{[i-1]}} & \frac{1}{b_1 - b_3 x_1^{[i-1]}} & \frac{-a_{12}}{b_1 - b_3 x_1^{[i-1]}} \end{bmatrix}. \quad (4.98)$$

It has been noticed that the observability indices are $n_1 = 2$ and $n_2 = 1$. A Lyapunov transformation $P(t)$ can be constructed as

$$P(t) = [A(t) \alpha_1 - \dot{\alpha}_1, \quad \alpha_1, \quad \alpha_2], \quad (4.99)$$

where α_1 and α_2 are the 2-nd and the 3-rd column of $\bar{N}^{-1}(t)$, respectively.

After getting $P(t)$ and following the algorithm procedures in [102], we have the following steps:

1) Obtain $\bar{A}(t), \bar{B}(t), \bar{C}(t)$:

$$\begin{aligned} \bar{A}(t) &= P^{-1}(t) (A(t)P(t) - \dot{P}(t)), \\ \bar{B}(t) &= P^{-1}(t) B(t) = P^{-1}(t)[a_{13}, a_{23}, 0]^T, \\ \bar{C}(t) &= C(t) P(t). \end{aligned} \quad (4.100)$$

From (4.100), the calculation of $\bar{A}(t)$ requires the calculation of $\dot{P}(t)$. If $\bar{N}(t)$ is a constant matrix, α_1 and α_2 are constant vectors. Then, $P(t)$ becomes $[A(t)\alpha_1, \alpha_1, \alpha_2]$, which contains at most $x^{[i-1]}(t)$. In this case, the calculation of $\dot{P}(t)$ only requires $\dot{x}^{[i-1]}(t)$ besides $x^{[i-1]}(t)$. However, if α_1 is not constant, but having terms of $x^{[i-1]}(t)$, we are in a more complex situation to calculate $\dot{x}^{[i-1]}(t)$. Notice that in the linear approximation framework, the input signal $x^{[i-1]}(t)$ to the i -th subsystem has no analytical formula, which is different from the situation when $A(t), B(t), C(t)$ in (4.92) has analytical formulae of t , where the matrix $P(t)$ is also in an analytical form and its derivative $\dot{P}(t)$ can be calculated straightforward.

To calculate $\ddot{x}^{[i-1]}(t)$, more information is needed besides $\dot{x}^{[i-1]}(t)$ and $x^{[i-1]}(t)$. This is because:

$$\begin{aligned}\ddot{x}^{[i-1]}(t) &= \frac{d}{dt}[\dot{x}^{[i-1]}(t)], \\ &= \frac{d}{dt} \left[A(x^{[i-2]})x^{[i-1]}(t) + \begin{bmatrix} a_{13} \\ a_{23} \\ 0 \end{bmatrix} \right], \\ &= A(x^{[i-2]})\dot{x}^{[i-1]}(t) + \dot{A}(x^{[i-2]})x^{[i-1]}(t).\end{aligned}\tag{4.101}$$

In the above equation, the calculation of $\dot{A}(x^{[i-2]})$ requires $\dot{x}^{[i-2]}(t)$. Thus, the input to each LTV subsystem for the range identification problem include $x^{[i-2]}(t)$ and $\dot{x}^{[i-2]}(t)$, as well as $x^{[i-1]}(t)$ and $\dot{x}^{[i-1]}(t)$.

2) Choose the desired eigenvalues $(\lambda_1, \lambda_2, \lambda_3)$. The resulting coefficients of the characteristic polynomial are $(\beta_0, \beta_1, \beta_2)$ in the form of

$$\prod_{i=1}^n (\lambda - \lambda_i) = \lambda^n + \beta_{n-1}\lambda^{n-1} + \dots + \beta_1\lambda + \beta_0.\tag{4.102}$$

3) Obtain $\bar{G}(t)$:

$$\bar{G}(t) = (A_2 - A_3 + A_4)C_1^{-1},\tag{4.103}$$

where

$$\begin{aligned}A_2 &= [\bar{A}(:, 1), \bar{A}(:, 3)], \quad C_1 = [\bar{C}(:, 1), \bar{C}(:, 3)], \\ A_3 &= \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad A_4 = \begin{bmatrix} \beta_0 & 0 \\ \beta_1 & 0 \\ \beta_2 & 0 \end{bmatrix},\end{aligned}\tag{4.104}$$

and where $\bar{A}(:, i)$ and $\bar{C}(:, j)$ denote the i -th and the j -th column of $\bar{A}(t)$ and $\bar{C}(t)$, respectively.

4) $\bar{F}(t)$: Using the above design procedures, $\bar{F}(t)$ becomes

$$\bar{F}(t) = \begin{bmatrix} -\beta_0 & 1 & 0 \\ -\beta_1 & 0 & 1 \\ -\beta_2 & 0 & 0 \end{bmatrix}. \quad (4.105)$$

4.5.3.1 Comments on Availability of Signals

In the LTV observer (4.93), it is assumed that $x_0, x^{[i-1]}(t)$ and $y^{[i]}(t)$ are known signals for the i -th approximation subsystem. Unfortunately, the signals $x_0, x^{[i-1]}(t)$ and $y^{[i]}(t)$ are unknown from the original nonlinear system (4.89). If $x_0, A(x), B(x)$ are all assumed known, all the information of the original nonlinear system is available and there is no need to construct an observer. In other words, the fundamental nature of an observer is corrupted using the “observer” in (4.93).

Then, how can the linear approximation technique help for the observer design of an original nonlinear system? For the PDS in (4.89), $C(x)$ is a constant matrix. Outputs of the sequence of LTV subsystems will converge to the actual output $y(t)$ of the original nonlinear system. That is:

$$\lim_{t \rightarrow \infty} y^{[i]}(t) = y(t), \quad \text{as } i \rightarrow \infty. \quad (4.106)$$

By replacing $y^{[i]}(t)$ with $y(t)$ and $x^{[i-1]}(t)$ with $\hat{x}^{[i-1]}(t)$, we construct the following sequence of LTV observers, which we call LAO for an abbreviation of Linear Approximation Observer:

$$\text{LAO} \begin{cases} \dot{\hat{x}}^{[0]}(t) = \bar{F} \hat{x}^{[0]}(t) + \bar{G}(x'_0) y(t) + \bar{B} u(t), \\ \dot{\hat{x}}^{[i]}(t) = \bar{F} \hat{x}^{[i]}(t) + \bar{G}(\hat{x}^{[i-1]}) y(t) + \bar{B} u(t), \\ x^{[0]}(0) = x'_0, \text{ for } i = 0, \quad \hat{x}^{[i]}(0) = x'_0, \text{ for } i \geq 1, \end{cases} \quad (4.107)$$

with

$$\hat{x}^{[i]}(t) = P(t) \hat{\hat{x}}^{[i]}(t), \quad \text{for } i \geq 0,$$

where $u(t)$ and $y(t)$ are the input and output of the original nonlinear system (4.89). $\hat{x}^{[i-1]}(t)$ is the state estimation from the $(i-1)$ -th subsystem.

In (4.107), the state estimate is denoted by $\hat{x}^{[i]}(t)$ as an abuse of the variable. It is worth emphasizing that $\hat{x}^{[i]}(t)$ is not the estimate of $x^{[i]}(t)$, but the estimate of state $x(t)$ of the original nonlinear system. Besides, the initial conditions of each LTV subsystem are chosen to be x'_0 , which is different from x_0 . As noticed, based on the linear approximation techniques, a sequence of observers needs to be constructed. Fortunately, this drawback is not that severe due to the modularity feature of each observer.

Remark 4.5.1 *The application of the LTV observer in [102] requires the LTV system to be uniformly observable ($N(t)$ is invertible) and have a lexicographic basis (one row in $N_2(t)$ in (4.96) must always be able to form a 3×3 nonsingular matrix with $N_1(t)$). Obviously, the above observability condition of the LTV observer imposes restrictions on the overall LAO observer. LTV observers that have less restricted observability conditions, possibly with only mild Lipschitz requirement, thus need to be investigated and applied. Due to the above mentioned restriction, the LTV observer in [102] is only temporarily applied to illustrate the main point. That is, after applying the linear approximation technique, the observer design of a nonlinear system can be reduced to the conventional observer design of a sequence of LTV systems.*

The linear approximation-based observer can be applicable to general nonlinear systems that satisfy the mild local Lipschitz condition for linear approximation, not just a special class of nonlinear systems, as for which the IBO, SMO, and RIO are designed. Further, for a PDS with possible more general nonlinear imaging surface, this LAO can still be applied.

4.5.4 Discussions

- 1) Must $B(x)$ and $C(x)$ in (4.87) be constant matrices?

When the original nonlinear system is rewritten in the form

$$\begin{aligned}\dot{\hat{x}}^{[i]} &= A(\hat{x}^{[i-1]})\hat{x}^{[i]} + B(\hat{x}^{[i-1]})u, \\ y^{[i]} &= C(\hat{x}^{[i-1]})\hat{x}^{[i]},\end{aligned}\tag{4.108}$$

the designed LAO in (4.107) works when the matrix $B(x)$ is not constant, as will be shown in one of the examples in section 4.6.2. In this case, the observer design follows the same procedures in [102]. However, when the matrix $C(x)$ is not constant, the designed LAO fails since a simple substitute of $y^{[i]}(t)$ with $y(t)$ is not correct. Thus, linear approximation based nonlinear observers for more general nonlinear systems with a non-constant $C(x)$ matrix needs to be investigated further.

2) Why $x^{[i]}(t)$ and $\hat{x}^{[i]}(t)$ are not the same?

From the two sets of approximations in (4.88) and (4.107), it is obvious that $x^{[i]}(t)$ and $\hat{x}^{[i]}(t)$ are not the same. However, the appealing converging pattern of the LAO observer as illustrated in fig. 4.11 needs more study.

3) Must the motion parameters in the original nonlinear system be constant scalars for the application of LAO?

For example, for the PDS,

$$\dot{x} = Ax + b, \quad y = [x_1/x_3, x_2/x_3]^T, \quad (4.109)$$

must $a_{i,j}$ and b_j for $i, j = 1, 2, 3$ are all constant scalars? The answer is No. If the motion parameters are not constant scalars, but varying according to a law, the whole design procedures are the same, since the time-varying feature in the motion parameters can be handled by the LTV observer. One example is going to be shown in section 4.6.2.

4) Relationship with linear approximation:

The LAO observer in (4.107) can also be regarded as a sequence of LTV approximations of the original nonlinear system, which is similar to the approximations described in equation (4.88) at this point. The difference between these two sets of approximations is shown in fig. 4.7. The approximations in (4.88) require the initial state x_0 of the nonlinear system. In other words, it approximates a completely known

nonlinear system, since the structure and the system parameters are already assumed known. The proposed LAO observer, though motivated directly by the approximation idea, does not require a knowledge of x_0 . The LAO observer approximates a partially known nonlinear system, whose structure and the parameters are known but with an unknown initial state, by applying a sequence of LTV observers.

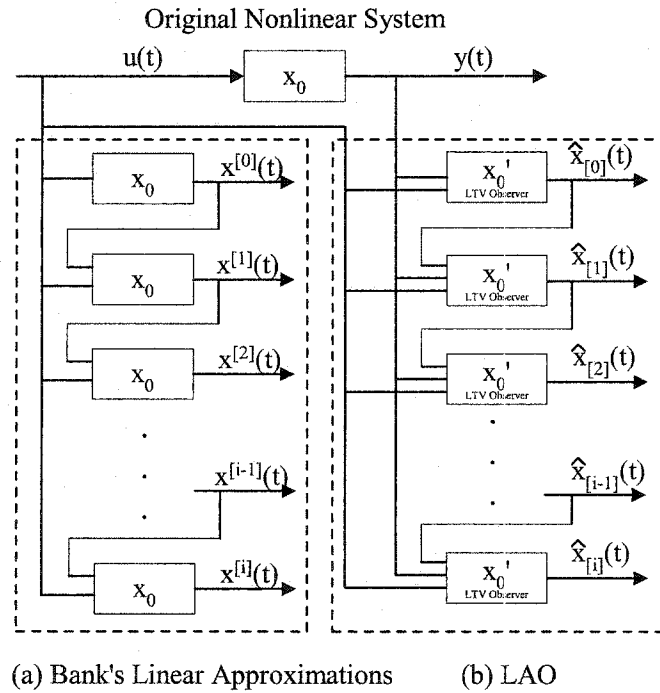


Fig. 4.7: Linear approximations of a nonlinear system.

4.5.5 Simulation Results

Simulation results of the LAO observer are presented along with three other perspective nonlinear observers, the IBO [23], SMO [24], and RIO [25], for a PDS system when the target is moving according to the following affine motions:⁸

⁸Throughout this dissertation, the motion dynamics that are used to test the perspective nonlinear observers are from the two examples in [24], except the two motions in the following, which is due to the reason described in Remark 4.5.1

1) **Motion₁**:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} -0.2301 & 0.4043 & -0.5769 \\ 0.1276 & -0.3003 & 0.2839 \\ 0.2450 & -0.4247 & 0.4319 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0.25 \\ 0.3 \end{bmatrix}, \quad (4.110)$$

$$[X_0, Y_0, Z_0]^T = [0.4, 0.6, 1.0]^T.$$

2) **Motion₂**:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} -1.7073 & -0.6368 & -1.0540 \\ 0.2279 & -1.0026 & -0.0715 \\ 0.6856 & -0.1856 & 0.2792 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} 0.5 \\ 0.25 \\ 0.3 \end{bmatrix}, \quad (4.111)$$

$$[X_0, Y_0, Z_0]^T = [0.4, 0.6, 1.0]^T.$$

In this section, the linear approximation technique is first implemented to verify/show, via example, how the $x^{[i]}(t)$ signals in (4.88) converge to the state $x(t)$ of the nonlinear system. Secondly, detailed experimental results are presented for the LAO observer's convergence using the two motions in (4.110) and (4.111), where we again emphasize that the state estimates $\hat{x}^{[i]}(t)$ in (4.107) converge to the actual state $x(t)$ of the nonlinear system, not the $x^{[i]}(t)$ in (4.88). Finally, performance comparisons of our LAO with the other three perspective nonlinear observers are presented.

4.5.5.1 Simulation Validation of Linear Approximation Technique

Given a nonlinear system (4.87), if all the information about the system is known, i.e., its initial condition, its structure, its parameters, and its input, then a sequence of LTV subsystems can be formed, as illustrated in fig. 4.8. In each subsystem of the sequence, it receives $x^{[i-1]}(t)$ signal from its previous block such that the matrices $A(x^{[i-1]}(t))$, $B(x^{[i-1]}(t))$, $C(x^{[i-1]}(t))$ can be determined. In this way, each subsystem becomes a LTV system.

Using **Motion₁**, the states $x^{[i]}(t)$ for $i = 1, 2, 3$ are shown in fig. 4.9. It is observed that the convergence pattern of $x^{[i]}(t)$ to $x(t)$ is up and down around $x(t)$, and finally converges to $x(t)$.

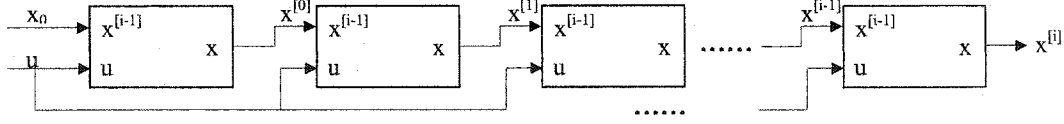
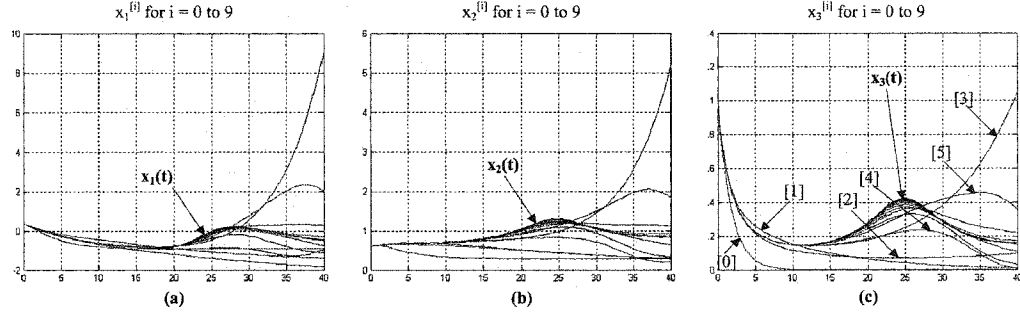


Fig. 4.8: A sequence of LTV subsystems of a nonlinear system.

Fig. 4.9: $x_{1,2,3}^{[i]}(t)$ of (4.91) under **Motion₁** without noise.

4.5.5.2 Performance Study of LAO Observer

A block diagram of the LAO observer is shown in fig. 4.10, where the inputs to each observer module include: 1) the actual output of the original nonlinear system, 2) the control input, and 3) $P(t)$, $\bar{F}(t)$, $\bar{B}(t)$, $\bar{G}(t)$ calculated using $\hat{x}^{[i-1]}(t)$, $\dot{\hat{x}}^{[i-1]}(t)$ and possibly $\hat{x}^{[i-2]}(t)$, $\dot{\hat{x}}^{[i-2]}(t)$ signals. The “LTVObs” function, implemented by a S-function in Matlab Simulink due to its demanding matrix manipulations, calculates the $P(t)$, $\bar{F}(t)$, $\bar{B}(t)$, $\bar{G}(t)$ matrices in equations (4.99), (4.100), (4.103), and (4.105). With the above calculated matrices, the other part of the LTV observer module outputs the state estimates according to (4.94).

Under **Motion₁** and **Motion₂**, the LAO observer is tested in the cases of no noise and with uniform noise bounded by $\pm 10^{-2}$ with two different initial conditions $x'_0 = [0, 0, 0]^T$ and $x'_0 = [-1, 2, 1]^T$. In this section, only simulation results with $x'_0 = [0, 0, 0]^T$ are presented to save space and maintain clarity. The design parameters for the LAO observer are $\lambda_1 = -1$, $\lambda_2 = -2$, $\lambda_3 = -3$. That is, $\beta_0 = 6$, $\beta_1 = 11$, $\beta_2 = 6$.

In fig. 4.11, $\hat{x}_{1,2,3}^{[i]}(t)$ for $i = 0, \dots, 9$ are plotted together with their true state trajectories for the ideal case of no noise. To extensively test the LAO observer, the simulation

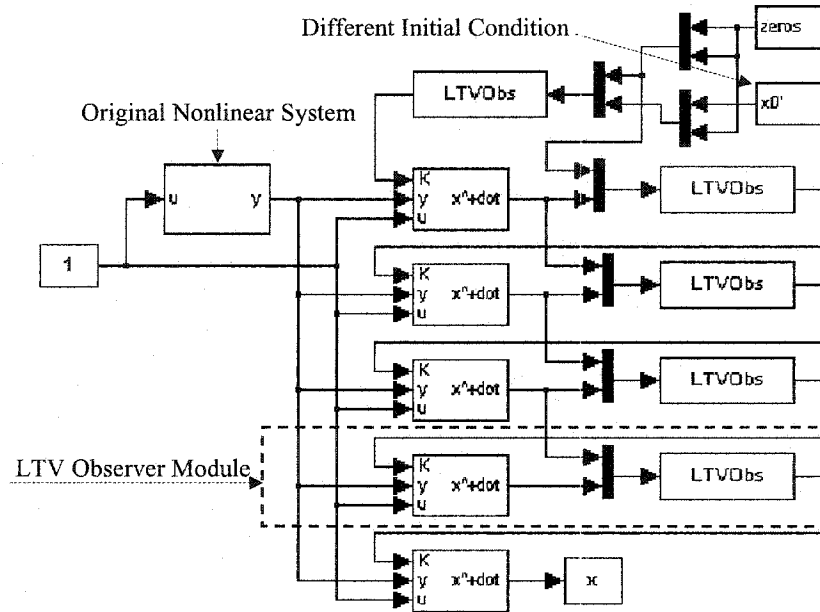


Fig. 4.10: Matlab simulation block diagram of the LAO observer.

time is set to be 80 seconds. It is observed that as i increases, $\hat{x}_{1,2,3}^{[i]}(t)$ reach $x_{1,2,3}(t)$ closer and closer. The LAO is also tested when the output is corrupted with uniform noise bounded by $\pm 10^{-2}$, as shown in fig. 4.12. Compared with fig. 4.11, $\hat{x}_{1,2,3}^{[i]}(t)$ in fig. 4.12 are more noisy, but still show good convergence to their true trajectories.

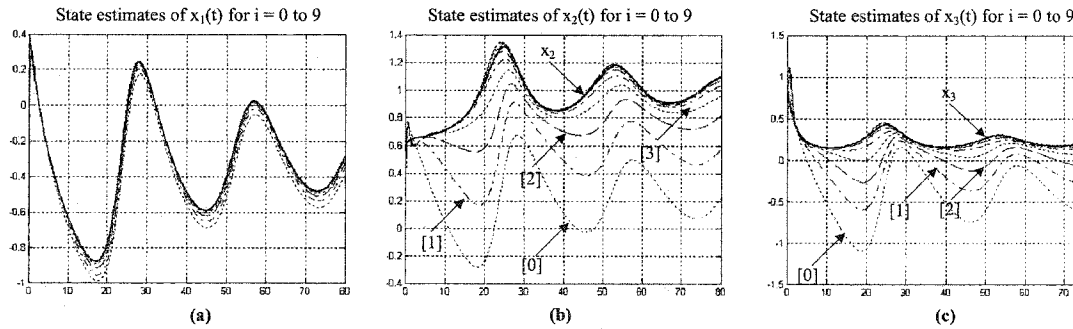


Fig. 4.11: $\hat{x}_{1,2,3}^{[i]}(t)$ under **Motion₁** without noise.

The above simulations are also performed using **Motion₂**, where the corresponding results are shown in figs. 4.13 and 4.14, respectively. The simulation time is set to be 20 seconds due to the simple varying pattern of the system's outputs.

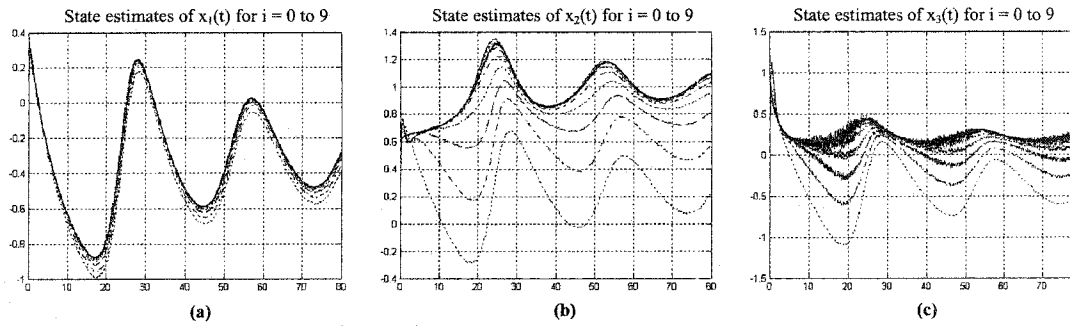


Fig. 4.12: $\hat{x}_{1,2,3}^{[i]}(t)$ under **Motion₁** with uniform noise bounded by $\pm 10^{-2}$.

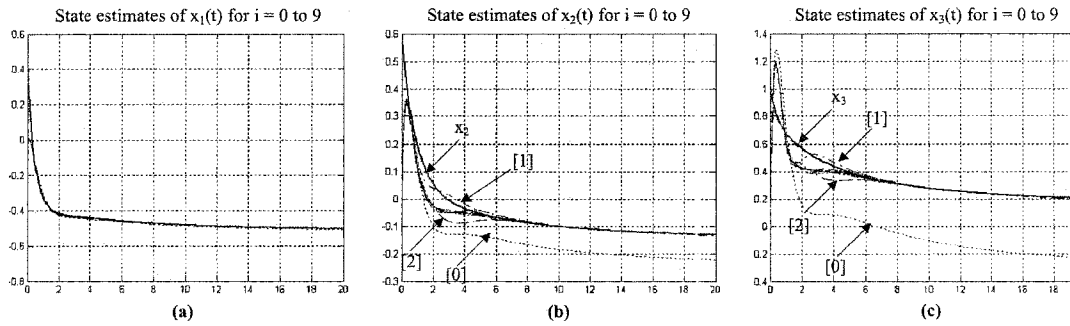


Fig. 4.13: $\hat{x}_{1,2,3}^{[i]}(t)$ under **Motion₂** without noise.

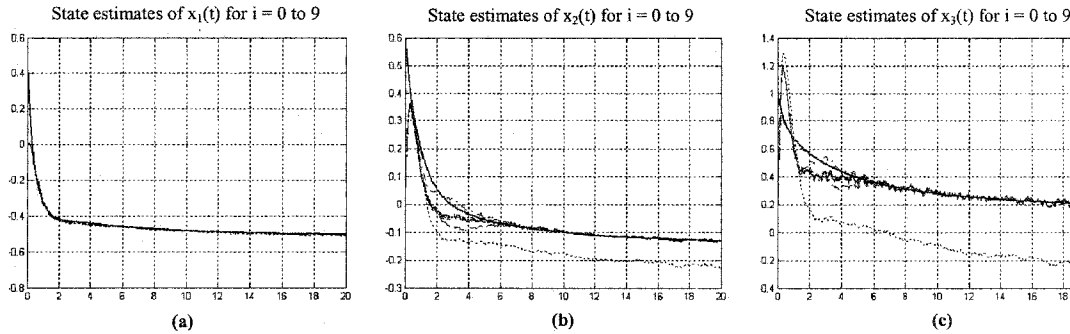


Fig. 4.14: $\hat{x}_{1,2,3}^{[i]}(t)$ under **Motion₂** with uniform noise bounded by $\pm 10^{-2}$.

4.5.5.3 Comparison Among Perspective Nonlinear Observers

At the last part of our simulations, a comparison between the LAO and the other three perspective nonlinear observers is performed using the two motions in (4.110) and (4.111), where the simulation results are shown in figs. 4.15 and 4.16, respectively. In these

comparisons, we use $x'_0 = [0, 0, 0]^T$. The observer parameters for the other three observers are the same as those in section 4.4.2.

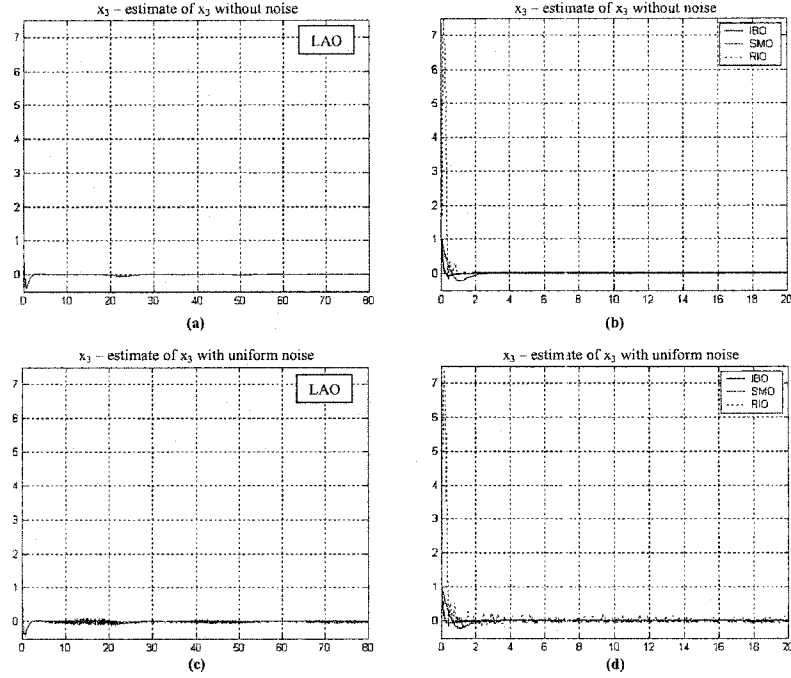


Fig. 4.15: Comparison of LAO with the other three perspective nonlinear observers under **Motion₁** with $x_0 = [0.4, 0.6, 1.0]^T$ and $x'_0 = [0, 0, 0]^T$: (a, b) without noise; (c, d) with uniform noise bounded by $\pm 10^{-2}$.

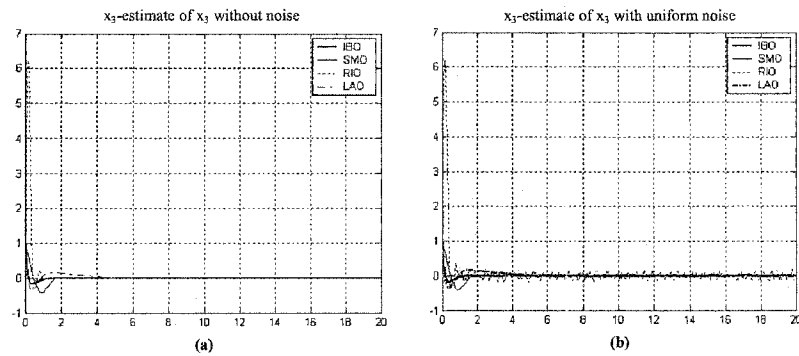


Fig. 4.16: Comparison of LAO with the other three perspective nonlinear observers under **Motion₂** with $x_0 = [0.4, 0.6, 1.0]^T$ and $x'_0 = [0, 0, 0]^T$: (a, b) without noise; (c, d) with uniform noise bounded by $\pm 10^{-2}$.

Figures 4.15 and 4.16 show the state estimation error, $x_3(t) - \hat{x}_3(t)$, for all the four observers. Note that in figs. 4.15 and 4.16, $\hat{x}_3^{[9]}(t)$ is taken as $\hat{x}(t)$ for the LAO observer. In fig. 4.15, the plots in (a) and (b) are the comparisons among the four observers in the ideal case of no noise, while (c) and (d) show the comparison in the presence of uniform noise bounded by $\pm 10^{-2}$. Similarly, in fig. 4.16, (a) shows the results without noise and (b) with uniform noise. Notice that a different simulation time (80 seconds) is used in fig. 4.15 for the LAO for an extensive testing of its performance. From figs. 4.15 and 4.16, it can be observed that the LAO observer has a slower converging speed than the other three observers designed specifically for the PDS. Though different observer design parameters might result in different converging speed, it is not surprising that the LAO, which is based on the linear approximation technique applicable to more general nonlinear systems, would not perform as well as the perspective nonlinear observers designed specifically for a PDS, as the other three observers in comparison.

4.5.6 Concluding Remarks

In this section, a recently developed linear approximation technique, which replaces a general nonlinear system (satisfying local Lipschitz condition) by a sequence of linear time-varying (LTV) approximations, is used for the design of a state observer for the range identification of a perspective dynamic system (PDS). Using the linear approximation idea, state estimation of the original nonlinear system is reduced to a state estimation of a sequence of LTV subsystems, where standard linear methods can be applied. Since the linear approximation-based observer is applicable to a broad class of nonlinear systems, when applied to the range identification problem, it has a slower converging speed compared to several other nonlinear observers designed specifically for a PDS. Moreover, the observability conditions of the LTV observer imposes a restriction on the resultant overall observer. LTV observers with less restricted condition thus need to be investigated and applied in this linear approximation framework. However, the LAO observer proposed in this section has a straightforward application to PDS with more general setup, e.g., with a general imaging surface as in certain omnidirectional vision systems.

4.6 Range Identification for Perspective Dynamic System with Single Homogeneous Observation

In this section, we consider the range identification problem with a single homogeneous observation. That is, when either $y_1(t)$ or $y_2(t)$ is known, instead both of them. Consider a special situation, as shown in fig. 4.17, when an object is moving on a plane P_1OP_2 , whose projection on the image plane is a line p_1p_2 that has either a constant $y_1(t)$ or a constant $y_2(t)$. If $y_2(t)$ is a constant, $\dot{y}_2 = 0$. The range identification problem becomes to identify $y_2(t)$ and $y_3(t)$ using $y_1(t)$.⁹ The above discussion serves as a motivation to investigate the range identification problem for a PDS with single homogeneous observation. However, in the following sections, $y_2(t)$ will be treated as unavailable, not necessarily be restricted as a constant. This discussion is an original contribution of this dissertation.

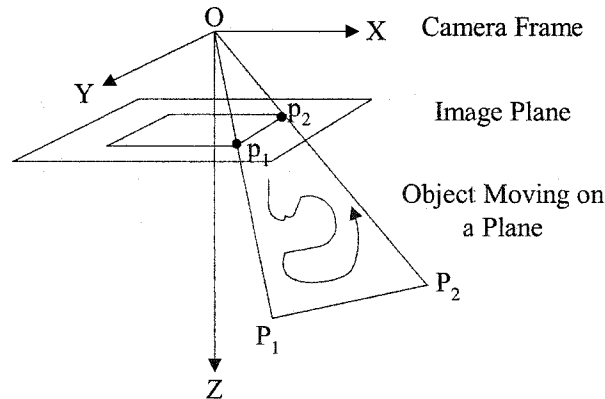


Fig. 4.17: Illustration of PDS with single observation function.

4.6.1 Nonlinear Observers for PDS with Single Homogeneous Observation

Range identification for a PDS using $(y_1(t)$ and $y_2(t))$ has been discussed in the previous sections. In the case of single homogeneous observation using only $y_1(t)$, range identification can be solved by a direct application of IBO since the IBO observer is designed for a class of nonlinear systems in the form of (4.69). Further, based on a resemblance in

⁹The case to estimate $y_1(t)$ and $y_3(t)$ using $y_2(t)$ is similar.

the constructions of IBO and SMO for the case of full homogeneous observations (when both $y_1(t)$ and $y_2(t)$ are available), a modified SMO can be used for the single observation case. However, extension/modification of the RIO observer for the single observation case, though possible, is not straightforward and will not be pursued in this section.

4.6.1.1 Direct Application of IBO

Range identification with single homogeneous observation can be solved by a direct application of the IBO observer, which has been applied to estimate $y_3(t)$ in [23] when both $y_1(t)$ and $y_2(t)$ are available. The dynamic system (4.72) can be rewritten in the form of (4.69) as

$$\begin{cases} \begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} b_1 - b_3 y_1 \\ b_2 - b_3 y_2 \end{bmatrix}}_{w^T([y_1, y_2]^T)} y_3 + \underbrace{\begin{bmatrix} a_{13} + (a_{11} - a_{33})y_1 + a_{12}y_2 - a_{31}y_1^2 - a_{32}y_1y_2 \\ a_{23} + a_{21}y_1 + (a_{22} - a_{33})y_2 - a_{31}y_1y_2 - a_{32}y_2^2 \end{bmatrix}}_{\phi([y_1, y_2]^T)}, \\ \dot{y}_3 = -\underbrace{(a_{31}y_1 + a_{32}y_2 + a_{33})y_3 - b_3y_3^2}_{g([y_1, y_2]^T, y_3)}, \end{cases} \quad (4.112)$$

and

$$\begin{cases} \dot{y}_1 = \underbrace{[a_{12} - a_{32}y_1, b_1 - b_3y_1]}_{w^T(y_1)} \begin{bmatrix} y_2 \\ y_3 \end{bmatrix} + \underbrace{[a_{13} + (a_{11} - a_{33})y_1 - a_{31}y_1^2]}_{\phi(y_1)}, \\ \begin{bmatrix} \dot{y}_2 \\ \dot{y}_3 \end{bmatrix} = \underbrace{\begin{bmatrix} \text{Same as } \dot{y}_2 \text{ in (4.72)} \\ \text{Same as } \dot{y}_3 \text{ in (4.72)} \end{bmatrix}}_{g(y_1, [y_2, y_3]^T)}, \end{cases} \quad (4.113)$$

respectively. Define

$$e_1 = y_1 - \hat{y}_1, \quad e_2 = y_2 - \hat{y}_2, \quad e_3 = y_3 - \hat{y}_3. \quad (4.114)$$

The constructed IBO observer for the case when $(y_1(t), y_2(t))$ are available is shown in (4.74) and in the following when only $y_1(t)$ is available:

$$\text{IBO}_{y_1} : \begin{cases} \dot{\hat{y}}_1 = GH e_1 + [a_{12} - a_{32}y_1, b_1 - b_3y_1] \begin{bmatrix} \hat{y}_2 \\ \hat{y}_3 \end{bmatrix} + [a_{13} + (a_{11} - a_{33})y_1 - a_{31}y_1^2], \\ \begin{bmatrix} \dot{\hat{y}}_2 \\ \dot{\hat{y}}_3 \end{bmatrix} = -G^2 \begin{bmatrix} a_{12} - a_{32}y_1 \\ b_1 - b_3y_1 \end{bmatrix} P e_1 + \begin{bmatrix} a_{23} + a_{21}y_1 + (a_{22} - a_{33})\hat{y}_2 \\ -(a_{31}y_1 + a_{32}\hat{y}_2 + a_{33})\hat{y}_3 \end{bmatrix} \\ \quad + \begin{bmatrix} -a_{31}y_1\hat{y}_2 - a_{32}\hat{y}_2^2 + (b_2 - b_3\hat{y}_2)\hat{y}_3 \\ -b_3\hat{y}_3^2 \end{bmatrix}, \end{cases} \quad (4.115)$$

under the corresponding observability conditions

$$\lambda_{\min}\{w([y_1(t), y_2(t)]^T) w^T([y_1(t), y_2(t)]^T)\} > \varepsilon > 0, \quad (4.116)$$

and

$$\lambda_{\min}\{w(y_1(t)) w^T(y_1(t))\} > \varepsilon > 0, \quad (4.117)$$

where λ_{\min} denotes the smallest eigenvalue of a matrix. Notation wise, all the observer parameters correspond to their specific observers, i.e., G or γ do not share a global meaning.

The two observability conditions in (4.116) and (4.117) are of the same level of complexity. Substituting

$$\begin{aligned} w^T([y_1, y_2]^T) &= [b_1 - b_3 y_1, b_2 - b_3 y_2]^T, \\ w^T(y_1) &= [a_{12} - a_{32} y_1, b_1 - b_3 y_1], \end{aligned}$$

into (4.116) and (4.117), we have

$$\lambda_{\min}\{(b_1 - b_3 y_1)^2 + (b_2 - b_3 y_2)^2\} > 0, \quad (4.118)$$

and

$$\lambda_{\min}\left\{\begin{bmatrix} \tilde{a}^2 & \tilde{a}\tilde{b} \\ \tilde{a}\tilde{b} & \tilde{b}^2 \end{bmatrix}\right\} > 0, \quad (4.119)$$

with $\tilde{a} = a_{12} - a_{32} y_1$ and $\tilde{b} = b_1 - b_3 y_1$. The above two conditions are equivalent to

$$\begin{aligned} (b_1 - b_3 y_1)^2 + (b_2 - b_3 y_2)^2 &> 0, \\ (b_1 - b_3 y_1)^2 + (a_{12} - a_{32} y_1)^2 &> 0, \end{aligned} \quad (4.120)$$

which are obviously of the same complexity.

Proof of IBO, the observer in (4.115) for the single homogeneous observation case, is omitted from here since it is a direct application of the IBO, with its detailed proof provided in [23].

4.6.1.2 Direct Modification of SMO

The SMO observer proposed in [24] has been applied to the state estimation of (4.72) when both $y_1(t)$ and $y_2(t)$ are available. When only $y_1(t)$ is available, the following observer,

which is based on a modification of the SMO and a resemblance between SMO and IBO, can also be used for the state estimation of $y_2(t)$ and $y_3(t)$:

$$\text{SMO}_{y_1} : \begin{cases} \dot{\hat{y}}_1 = \frac{\hat{\lambda}_1(t)e_1}{|e_1| + \delta_1} + [a_{12} - a_{32}y_1, b_1 - b_3y_1] \begin{bmatrix} \hat{y}_2 \\ \hat{y}_3 \end{bmatrix} + [a_{13} + (a_{11} - a_{33})y_1 - a_{31}y_1^2], \\ \begin{bmatrix} \dot{\hat{y}}_2 \\ \dot{\hat{y}}_3 \end{bmatrix} = \alpha \begin{bmatrix} a_{12} - a_{32}y_1 \\ b_1 - b_3y_1 \end{bmatrix} \frac{\hat{\lambda}_1(t)e_1}{|e_1| + \delta_1} + \begin{bmatrix} a_{23} + a_{21}y_1 + (a_{22} - a_{33})\hat{y}_2 \\ -(a_{31}y_1 + a_{32}\hat{y}_2 + a_{33})\hat{y}_3 \end{bmatrix} \\ \quad + \begin{bmatrix} -a_{31}y_1\hat{y}_2 - a_{32}\hat{y}_2^2 + (b_2 - b_3\hat{y}_2)\hat{y}_3 \\ -b_3\hat{y}_3^2 \end{bmatrix}, \end{cases} \quad (4.121)$$

where $\hat{\lambda}_1(t)$ is adaptively updated by:

$$\dot{\hat{\lambda}}_1(t) = \begin{cases} 2\alpha_1|e_1|, & \text{if } |e_1| > 2\delta_1, \\ 0, & \text{otherwise,} \end{cases} \quad (4.122)$$

with observability condition similar to that in (4.117).

The modified SMO observer SMO_{y_1} in (4.121) achieves extremely similar performance to IBO using properly chosen observer parameters. However, its proof of convergence is not as straightforward.

4.6.2 Simulation Results

The range identification observers IBO_{y_1} and SMO_{y_1} in (4.115) and (4.121) for single homogeneous observation are tested via Matlab simulations using the first example in [24], where the target is moving according to the affine motion in (4.84). In our simulations, the observer parameters are chosen to be

$$\begin{aligned} G = 10, H = 1, P = -1/2, M = 10, \gamma = 1, \\ \alpha = 5, \hat{\lambda}_1(0) = 1, \alpha_1 = 10, \delta_1 = 0.2, M = 10, \gamma = 1, \end{aligned} \quad (4.123)$$

for the IBO and SMO, respectively, with initial conditions

$$\begin{aligned} \hat{y}_1(0) = y_1(0) = X(0)/Z(0) = 0.4, \\ [\hat{y}_2(0), \hat{y}_3(0)] \in \{-1, -1\}, [0, 0], [1, 1\}. \end{aligned} \quad (4.124)$$

Since the SMO_{y_1} has an extremely close performance to that of IBO_{y_1} , simulation results are only presented for the IBO.

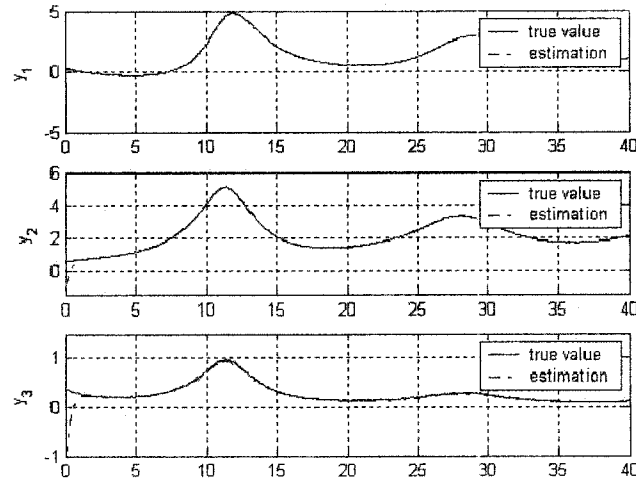


Fig. 4.18: State estimation using IBO with $[\hat{y}_2(0), \hat{y}_3(0)] = [-1, -1]$.

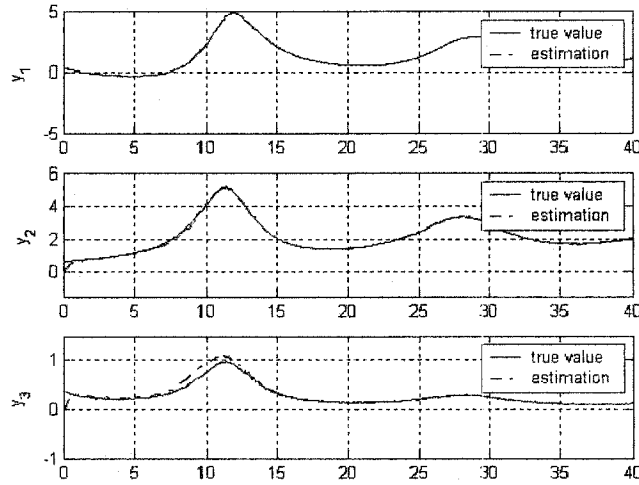


Fig. 4.19: State estimation using IBO with $[\hat{y}_2(0), \hat{y}_3(0)] = [0, 0]$.

First, simulation results are presented in figs. 4.18, 4.19, and 4.20 for IBO with the observer parameters (4.123) and initial conditions (4.124) in the ideal case with no noise. After 15 seconds, the estimations of $y_2(t)$ and $y_3(t)$ converge to their true values.

Next, comparison between IBO and IBO, the observers in (4.74) and (4.115), in the case of full and single observations is shown in fig. 4.21 for the ideal case of no noise with initial conditions $[\hat{y}_1(0), \hat{y}_2(0), \hat{y}_3(0)] = [0, 0, 0]$ for full observations and $[\hat{y}_1(0), \hat{y}_2(0), \hat{y}_3(0)] =$

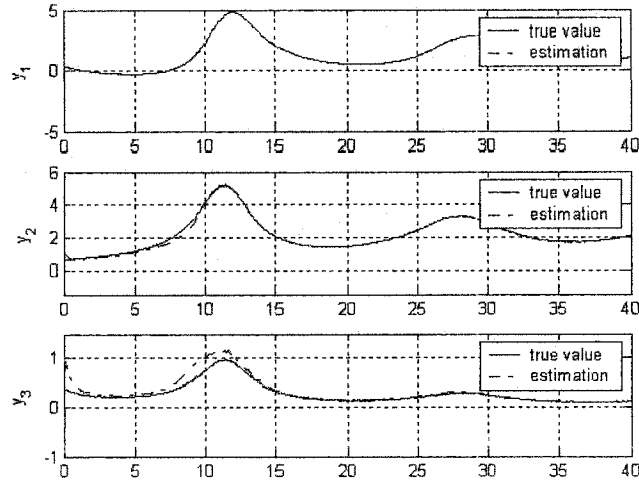


Fig. 4.20: State estimation using IBO with $[\hat{y}_2(0), \hat{y}_3(0)] = [1, 1]$.

$[X(0)/Z(0), 0, 0]$ for single. It is observed from fig. 4.21 that IBO for the single case has a less converging speed than that of IBO.

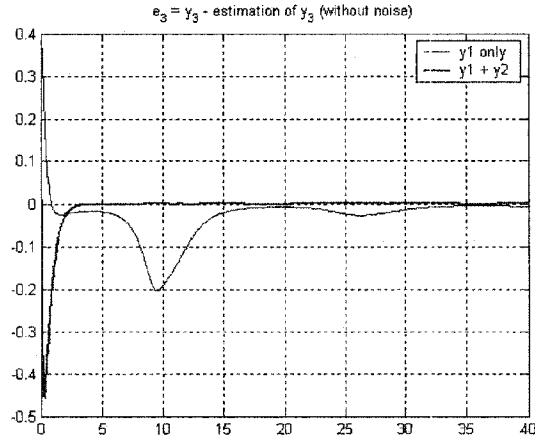


Fig. 4.21: Estimation error comparison between IBO and IBO in the ideal case of no noise.

Finally, the observers in IBO and IBO are compared for their robustness to noise, where observer performance is shown in fig. 4.22 with uniform noise bounded by $\pm 10^{-2}$. With a lagging converging performance similar to that in fig. 4.21, IBO achieves estimation accuracy comparable to that of IBO after 15 seconds. For fair comparison, in the comparisons shown in figs. 4.21 and 4.22, same/similar observer parameters are used: M, G, γ are

the same; (H, P) are chosen to be $(1, -1/2)$ for IBO_{y_1} and $(I_{2 \times 2}, -I_{2 \times 2}/2)$ for $\text{IBO}_{y_1+y_2}$, where $I_{2 \times 2}$ denotes the identity matrix of dimension two.

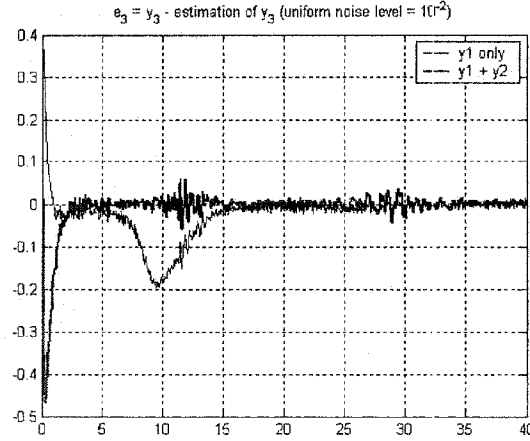


Fig. 4.22: Estimation error comparison between IBO_{y_1} and $\text{IBO}_{y_1+y_2}$ in the presence of uniform noise bounded by $\pm 10^{-2}$.

From the above three-step simulation studies, it can be concluded that IBO_{y_1} , as well as SMO_{y_1} , can achieve satisfactory estimation performance with single observation function. It is unsurprising that their convergence speed is not as fast as those with full observations. However, in the presence of noise, the SMO_{y_1} using single homogeneous observation manifests similar robustness performance compared to $\text{SMO}_{y_1+y_2}$ using full observations.

4.6.3 Concluding Remarks

For a perspective dynamic system (PDS) with single homogeneous observation function, the range identification problem was discussed using nonlinear observers previously used for the full observation case. Our simulation results show that convergence speed of the observer for the single observation case is slower than those with full observations. However, both observers have similar robust performance.

4.7 Perspective Systems with More General Projection Surfaces

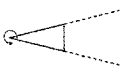



Conventional video cameras have limited fields of view that make them restrictive in a variety of vision applications, including autonomous navigation, remote surveillance, video

conferencing, and scene recovery. Existing devices typically use a photographic camera or a video camera, in conjunction with an off-the-shelf lens. This configuration allows the device to view the world through a relatively small solid angle subtended from the center of projection of the lens. To enhance the field of view (FOV), omnidirectional/panoramic imaging sensors that have big, close to hemi-spherical FOV are under developments [103]. However, it is desirable that the entire imaging system still possesses a single effective viewpoint to enable the generation of pure perspective images from a sensed scene [104]. More specifically, images that adhere to perspective projection are consistent with the way we observe scenes. Further, available works in computational vision that assume linear perspective projection can be applied for processing [105].

Current technologies to achieve wide FOV are illustrated in table 4.2, where the disadvantages of each technology are briefly listed in the followings:

- 1) Rotating camera: It requires moving parts and precise positioning. A serious drawback lies in the total time to obtain an image with enhanced FOV.
- 2) Cluster of cameras: Centers of projections reside inside each individual camera. Consequently, the entire imaging system does not have a unique effective viewpoint.
- 3) Fish-eye lens: It is difficult to design a fish-eye lens that ensures that all the incoming rays intersect at a single point to yield a fixed viewpoint. Using two fish-eye lens that each provides a hemi-spherical FOV requires perfect seaming.
- 4) Catadioptric camera: Incorporate reflecting surfaces (mirrors) into conventional imaging system.

Table 4.2: Technologies to Achieve Wide Fields of View

Rotating Camera	Cluster of Cameras	Fish-eye Lens	Catadioptric Lens
			

Though different difficulties exist in the above categorized technologies to construct powerful omnidirectional imaging systems, it is clear that imaging systems that have already been applied, or to be used in the future, are not restricted to the camera-type vision system, where the imaging surface is a 2-D plane perpendicular to the optical axis. While efforts are being carried on towards the construction of different candidates of omnidirectional vision systems, in this section, we disregard the physical construction issue, but focus on the 3-D motion and range identification problems based on the constructed omnidirectional systems, whose imaging surfaces are not necessarily a plane perpendicular to the optical axis, but can be a 3-D sphere, an ellipsoid, a paraboloid, and etc.

Currently, motion and range identification problems discussed in the perspective dynamic system (PDS) framework are regarding 3-D information observed via a conventional camera-type imaging system. In this section, the imaging surface is extended to a general plane, a 3-D sphere and ellipsoid, and a paraboloid surface. We assume that the discussed omnidirectional systems have a single center of projection such that the images observed preserve linear perspective geometry. Further, throughout this section, parameters of the imaging surface are assumed known.

The section is organized as follows. Sections 4.7.1 and 4.7.2 discuss the range identification and motion estimation problems with planar and ball-shape imaging surfaces. Section 4.7.3 addresses these two problems via observations on a paraboloid, where the resulting PDS does not preserve an affine form such that most existing perspective observers that are applicable for range identification problem via observations from a traditional camera-type vision system cannot be applied directly. Section 4.7.4 presents our simulations studies for the range identification problem when using a general planar surface, a sphere, and an ellipsoid as discussed in sections 4.7.1 and 4.7.2. Finally, section 4.7.5 concludes the section. The ideas in this section is our original contribution of this dissertation.

4.7.1 PDS with General Planar Imaging Surface

In a PDS system, the projection of a 3-D point $[X, Y, Z]^T$ can only be observed up to a homogeneous line. That is

$$X_p = Z_p X/Z, \quad Y_p = Z_p Y/Z, \quad (4.125)$$

where the subscript $_p$ denotes the projection on the imaging surfaces. An arbitrary planar surface, as shown in fig. 4.23, can be described by its normal vector $\vec{n} = [n_1, n_2, n_3]^T$ and a point on the plane. To simplify derivations, the point on the planar imaging surface is chosen to be $[0, 0, 1]$ without loss of generality. Thus, for any point $[X_p, Y_p, Z_p]^T$ on the planar imaging surface, we have

$$n_1 X_p + n_2 Y_p + n_3 (Z_p - 1) = 0, \quad (4.126)$$

where we further assume $n_3 \neq 0$ to enforce the imaging surface facing toward the Z axis. From equations (4.125) and (4.126), we can easily get

$$X_p = n_3 X/L_{pl}, \quad Y_p = n_3 Y/L_{pl}, \quad Z_p = n_3 Z/L_{pl}, \quad (4.127)$$

where $L_{pl} \triangleq n_1 X + n_2 Y + n_3 Z$ and the subscript $_{pl}$ denotes the planar surface.

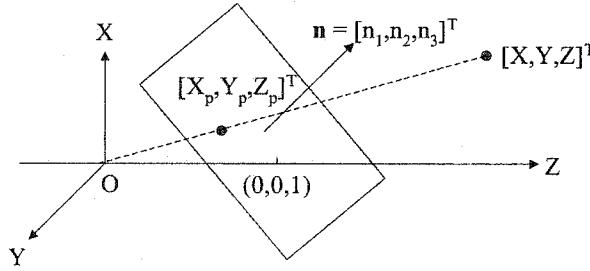


Fig. 4.23: Simulation results of e_3 for the three observers for Example₁.

4.7.1.1 Range Identification with Known Motion Parameters

Since n_3 is known (parameters of the imaging surface are assumed known in this section), we can choose $y(t)$, the observations of the PDS, to be $y(t) = [y_1, y_2, y_3]^T$ with

$$y_1 = X/L_{pl}, \quad y_2 = Y/L_{pl}, \quad y_3 = 1/L_{pl}, \quad (4.128)$$

where (y_1, y_2) are measurable from the imaging surface and y_3 contains the range information to estimate. From the above equation, $[X, Y, Z]^T$ can be calculated as

$$X = \frac{y_1}{y_3}, Y = \frac{y_2}{y_3}, Z = \frac{1 - n_1 y_1 - n_2 y_2}{n_3 y_3}. \quad (4.129)$$

Under the choice of $y(t) = [y_1, y_2, y_3]^T$ as in (4.128) and the assumption that the object is moving according to the affine motion in (4.40), the derivative of $y(t)$ is

$$\begin{cases} \begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \end{bmatrix} = \begin{bmatrix} \frac{a_{13}}{n_3} + (a_{11} - a_{13} \frac{n_1}{n_3} - \rho_3)y_1 + (a_{12} - a_{13} \frac{n_2}{n_3})y_2 \\ \frac{a_{23}}{n_3} + (a_{21} - a_{23} \frac{n_1}{n_3})y_1 + (a_{22} - a_{23} \frac{n_2}{n_3} - \rho_3)y_2 \end{bmatrix} \\ \quad + \begin{bmatrix} -\rho_1 y_1^2 - \rho_2 y_1 y_2 \\ -\rho_1 y_1 y_2 - \rho_2 y_2^2 \end{bmatrix} + \begin{bmatrix} b_1 - \rho_0 y_1 \\ b_2 - \rho_0 y_2 \end{bmatrix} y_3, \\ \dot{y}_3 = -(\rho_1 y_1 + \rho_2 y_2 + \rho_3)y_3 - \rho_0 y_3^2, \end{cases} \quad (4.130)$$

where

$$\begin{aligned} \rho_3 &= \frac{1}{n_3} \sum_{i=1}^3 n_i a_{i3}, & \rho_0 &= \sum_{i=1}^3 n_i b_i, \\ \rho_1 &= \sum_{i=1}^3 n_i a_{i1} - n_1 \rho_3, & \rho_2 &= \sum_{i=1}^3 n_i a_{i2} - n_2 \rho_3. \end{aligned} \quad (4.131)$$

When $Z_p = 1$ as for the traditional camera-type vision system, $n_1 = n_2 = 0$ and $n_3 = 1$. $(\rho_0, \rho_1, \rho_2, \rho_3)$ reduces to $(b_3, a_{31}, a_{32}, a_{33})$ and $\dot{y}(t)$ becomes (4.72), which is the PDS already presented in the above sections [23, 24, 106].

The dynamic system (4.130) still fits into the form of (4.69). In this way, state estimation of the perspective dynamic system (4.130) can be carried out using the IBO.

4.7.1.2 Motion Estimation via Optical Flow

Assuming that we have a planar textured surface described by

$$Z = pX + qY + r, \quad (4.132)$$

that always faces the vision system without any occlusion and every point on the surface moves according to the affine motion in (4.40), the optical flow dynamics¹⁰ on the imaging

¹⁰The optical flow dynamics is a time varying dynamic system described via the coordinates on the imaging surface.

surface is of the form (4.64), with

$$\begin{aligned}
d_1 &= (a_{13} + c_1)/n_3, & d_2 &= (a_{23} + c_2)/n_3, \\
d_3 &= (a_{11} - a_{13}\frac{n_1}{n_3} - \rho_3) - (\tilde{p}c_1 + \frac{\rho_0}{rn_3}), \\
d_4 &= a_{12} - a_{13}\frac{n_2}{n_3} - \tilde{q}c_1, \\
d_5 &= a_{21} - a_{23}\frac{n_1}{n_3} - \tilde{p}c_2, \\
d_6 &= (a_{22} - a_{23}\frac{n_2}{n_3} - \rho_3) - (\tilde{q}c_2 + \frac{\rho_0}{rn_3}), \\
d_7 &= \tilde{p}\frac{\rho_0}{r} - \rho_1, & d_8 &= \tilde{q}\frac{\rho_0}{r} - \rho_2,
\end{aligned} \tag{4.133}$$

where

$$\tilde{p} = p + \frac{n_1}{n_3}, \quad \tilde{q} = q + \frac{n_2}{n_3}, \quad c_i = \frac{b_i}{r}, \quad \text{for } i = 1, 2, 3. \tag{4.134}$$

Again, for the case $Z_p = 1$, $d_1 \sim d_8$ reduces to those in (4.65).

Motion parameter identification can be carried out in a two-step mode, where the first step is to estimate $\mathbf{d}_{\text{pl}} = [d_1, d_2, \dots, d_8]^T$, called the essential parameters in [21], and the second is to calculate motion parameters from \mathbf{d}_{pl} . To identify \mathbf{d}_{pl} , we can either stack a 8×8 matrix using four feature points or formulate into the following nonlinear system using one feature point:

$$\dot{\mathbf{y}} = w(\mathbf{y}) \mathbf{d}_{\text{pl}}, \quad \dot{\mathbf{d}}_{\text{pl}} = \phi(\mathbf{d}_{\text{pl}}), \tag{4.135}$$

such that the IBO observer in [23] can again be applied.

For a rigid motion, the equations in (4.133) are eight equations with eight parameters $(w_1, w_2, w_3, c_1, c_2, c_3, p, q)$, commonly known as the recovery equations. It has been reported that (w_1, w_2, w_3) can be identified uniquely, while (b_1, b_2, b_3) can only be estimated up to a scalar containing the depth ambiguity [21].

Remark 4.7.1 *Using the general planar imaging surface (4.126), the output observation function $y(t)$ can also be chosen as*

$$[X/L_{\text{pl}}, Y/L_{\text{pl}}, Z/L_{\text{pl}}, 1/L_{\text{pl}}]^T. \tag{4.136}$$

With the above choice of $y(t)$, for the range identification formulation, the resulting $\dot{\mathbf{y}}(t)$ shares the same structure as those in (4.141), with $f_s(\cdot)$ replaced by $\sum_{k=1}^3 \left[\sum_{i=1}^3 n_i a_{ik} \right] y_k$

and $g_s(\cdot)$ by $\sum_{i=1}^3 n_i b_i$. The resulting PDS system also fits into the form of (4.69) such that the IBO can be applied for the state estimation. For the motion parameter identification problem, essential parameters of the optical flow dynamics will have several more elements satisfying certain constraints. The reason that we choose $y(t)$ in the form of (4.130) for a planar imaging surface is to emphasize the extension of existing works in [21].

4.7.2 PDS with Ball-Shape Imaging Surfaces

Consider a spherical imaging surface as shown in fig. 4.24, where the center of the sphere and the radius R are assumed to be $[0, 0, 0]^T$ and $R = 1$ without loss of generality. Any point $[X_p, Y_p, Z_p]^T$ on the imaging surface satisfies

$$X_p^2 + Y_p^2 + Z_p^2 = 1. \quad (4.137)$$

From equations (4.125) and (4.137), we have

$$X_p = X/L_s, \quad Y_p = Y/L_s, \quad Z_p = Z/L_s, \quad (4.138)$$

where $L_s \triangleq \sqrt{X^2 + Y^2 + Z^2}$ and the subscript s denotes the spherical imaging surface.

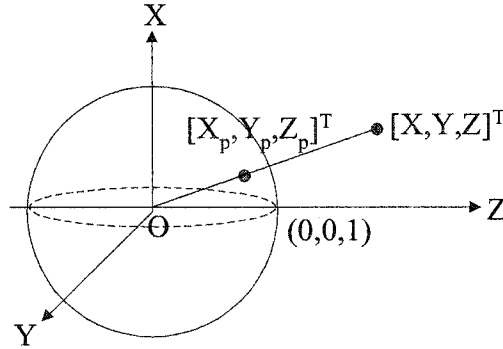


Fig. 4.24: A spherical imaging surface centered at origin with radius 1.

4.7.2.1 Range Identification with Known Motion Parameters

By letting $y(t) = [y_1, y_2, y_3, y_4]^T$ to be

$$y_1 = X/L_s, \quad y_2 = Y/L_s, \quad y_3 = Z/L_s, \quad y_4 = 1/L_s, \quad (4.139)$$

where (y_1, y_2, y_3) are observable from the imaging surface and y_4 contains the range information to estimate, $[X, Y, Z]^T$ can be calculated as

$$X = y_1/y_4, \quad Y = y_2/y_4, \quad Z = y_3/y_4. \quad (4.140)$$

The derivative of $y(t)$ is:

$$\begin{cases} \dot{y}_k = \sum_{i=1}^3 a_{ki} y_i - y_k f_s(\cdot) + [b_k - y_k g_s(\cdot)] y_4, \\ \dot{y}_4 = -y_4 [f_s(\cdot) + y_4 g_s(\cdot)], \end{cases} \quad (4.141)$$

for $k = 1, 2, 3$, where

$$\begin{aligned} f_s(\cdot) &= \sum_{i=1}^3 a_{ii} y_i^2 + \frac{1}{2} \sum_{i,j=1, i \neq j}^3 (a_{ij} + a_{ji}) y_i y_j, \\ g_s(\cdot) &= \sum_{i=1}^3 b_i y_i. \end{aligned} \quad (4.142)$$

Equation (4.141) is again in the form of (4.69) and the IBO can be applied for the state estimation.

Remark 4.7.2 For the range identification problem using a spherical imaging surface, the output $y(t)$ in the choice of (4.139) facilitates the application of the IBO observer for the state estimation of the resulting PDS. Unlike the case of a planar imaging surface, where the required term Z/L_{pl} to calculate (\dot{y}_1, \dot{y}_2) can be derived via $Z/L_{pl} = (1 - n_1 y_1 - n_2 y_2)/n_3$, which is a linear function of y_1 and y_2 . When using a spherical imaging surface as in (4.137), Z/L_s can not be written as a linear function of the measurable states. The introduction of $y_3 = Z/L_s$ into $y(t)$ makes the resulting PDS remain in the form of (4.69).

4.7.2.2 Motion Estimation via Optical Flow

Derivation of the optical flow on the spherical imaging surface follows the idea described in section 4.7.1.2. Assuming a planar texture structure in (4.132), we have

$$Z/L_s = pX/L_s + qY/L_s + r/L_s \Rightarrow 1/L_s = (y_3 - py_1 - qy_2)/r. \quad (4.143)$$

By substituting $y_4 = (y_3 - py_1 - qy_2)/r$ into equation (4.141), \dot{y}_k in (4.141) for $k = 1, 2, 3$ becomes

$$\dot{y}_k = \sum_{i=1}^3 a_{ki} y_i - y_k f_s(\cdot) + [c_k - y_k \sum_{i=1}^3 c_i y_i] (y_3 - py_1 - qy_2), \quad (4.144)$$

where $c_k = b_k/r$ as defined in (4.134).

For the rigid motion, $a_{ii} = 0$ and $(a_{ij} + a_{ji})|_{i \neq j} = 0$ for $i, j = 1, 2, 3$. Thus, $f_s(\cdot) = 0$. \dot{y}_k for $k = 1, 2, 3$ reduces to equation (4.145),

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \dot{y}_3 \end{bmatrix} = \underbrace{\begin{bmatrix} y_2 & y_3 & 0 & y_3 - y_1^2 y_3 & -y_1 y_2 y_3 & -y_1 y_3^2 & -y_1 + y_1^3 & y_1^2 y_2 & \cdots & y_1 y_2 y_3 \\ -y_1 & 0 & y_3 & -y_1 y_2 y_3 & y_3 - y_2^2 y_3 & -y_2 y_3^2 & y_1^2 y_2 & -y_1 + y_1 y_2^2 & \cdots & y_1 y_2 y_3 \\ 0 & -y_1 & -y_2 & -y_1 y_3^2 & -y_2 y_3^2 & y_3 - y_3^3 & y_1^2 y_3 & y_1 y_2 y_3 & \cdots & -y_2 + y_2 y_3^2 \end{bmatrix}}_{\substack{w(y) \\ \mathbf{d}_s}} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ c_1 \\ c_2 \\ c_3 \\ pc_1 \\ pc_2 \\ pc_3 \\ qc_1 \\ qc_2 \\ qc_3 \end{bmatrix}^T. \quad (4.145)$$

where the elements of \mathbf{d}_s satisfy the following four constraints:

$$d_8 = d_7 \frac{d_5}{d_4}, \quad d_9 = d_7 \frac{d_6}{d_4}, \quad d_{11} = d_{10} \frac{d_5}{d_4}, \quad d_{12} = d_{10} \frac{d_6}{d_4}. \quad (4.146)$$

Similar to the identification of \mathbf{d}_{pl} in (4.133), \mathbf{d}_s can be estimated either by SVD-based method or the IBO. After \mathbf{d}_s is identified, we have eight motion parameters ($w_1, w_2, w_3, c_1, c_2, c_3, p, q$) appearing in twelve equations satisfying four constraints, and the motion parameters can be calculated out.

4.7.2.3 PDS with Ellipsoid Imaging Surface

A spherical imaging surface can be further extended to an ellipsoid, such as characterized by:

$$\frac{X_p^2}{r_1^2} + \frac{Y_p^2}{r_2^2} + \frac{Z_p^2}{r_3^2} = 1. \quad (4.147)$$

The projection on the ellipsoid satisfies

$$X_p = X/L_e, \quad Y_p = Y/L_e, \quad Z_p = Z/L_e, \quad (4.148)$$

where $L_e \triangleq \sqrt{\frac{X^2}{r_1^2} + \frac{Y^2}{r_2^2} + \frac{Z^2}{r_3^2}}$ and the subscript e denotes the ellipsoid. Similar to the case of a spherical imaging surface, $y(t) = [y_1, y_2, y_3, y_4]^T$ can be chosen as

$$y_1 = X/L_e, \quad y_2 = Y/L_e, \quad y_3 = Z/L_e, \quad y_4 = 1/L_e. \quad (4.149)$$

After similar mathematical manipulations as those in (4.141), we can have

$$\begin{cases} \dot{y}_k = \sum_{i=1}^3 a_{ki} y_i - y_k f_e(\cdot) + [b_k - y_k g_e(\cdot)] y_4, \\ \dot{y}_4 = -y_4 [f_e(\cdot) + y_4 g_e(\cdot)], \end{cases} \quad (4.150)$$

where $k = 1, 2, 3$ and

$$\begin{aligned} f_e(\cdot) &= \sum_{i=1}^3 \frac{a_{ii}}{r_i^2} y_i^2 + \left(\frac{a_{12}}{r_1^2} + \frac{a_{21}}{r_2^2} \right) y_1 y_2 + \left(\frac{a_{13}}{r_1^2} + \frac{a_{31}}{r_3^2} \right) y_1 y_3 + \left(\frac{a_{23}}{r_2^2} + \frac{a_{32}}{r_3^2} \right) y_2 y_3, \\ g_e(\cdot) &= \sum_{i=1}^3 \frac{b_i}{r_i^2} y_i. \end{aligned} \quad (4.151)$$

For the 3-D motion estimation, the optical flow dynamics on the imaging surface takes the form

$$\dot{y}_k = \sum_{i=1}^3 \frac{a_{ki}}{r_i^2} y_i - y_k f_e(\cdot) + \left[c_k - y_k \sum_{i=1}^3 \frac{c_i}{r_i^2} y_i \right] (y_3 - p y_1 - q y_2). \quad (4.152)$$

for $k = 1, 2, 3$. For rigid motion, the essential parameters of the optical flow remains the same as \mathbf{d}_s in (4.145). The corresponding $w(y)$ matrix will be slightly more complex than that in (4.145).

4.7.3 PDS with Paraboloid Imaging Surface

In this section, we consider a paraboloid imaging surface characterized by:

$$Z_p = X_p^2 + Y_p^2, \quad (4.153)$$

which is illustrated in fig. 4.25. Using the above function, a 3-D point is projected onto

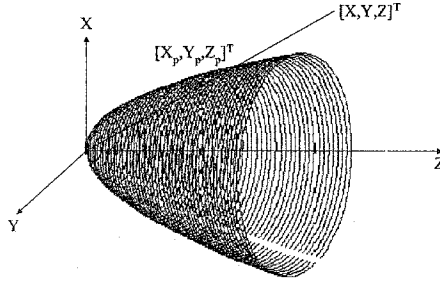


Fig. 4.25: A paraboloid imaging surface centered at $[0, 0, 0]^T$.

the imaging surface as

$$X_p = XZ/L_{pa}, \quad Y_p = YZ/L_{pa}, \quad Z_p = Z^2/L_{pa}, \quad (4.154)$$

where $L_{pa} \triangleq X^2 + Y^2$ and the subscript $_{pa}$ denotes the paraboloid.

For the range identification, the output $y(t)$ can be chosen to be $y(t) = [y_1, y_2, y_3, y_4]^T = [X_p, Y_p, Z_p, 1/L_{pa}]^T$, where (y_1, y_2, y_3) are observable and y_4 contains the range information to estimate. The procedure to derive $\dot{y}(t)$ is the same as those performed in sections 4.7.1 and 4.7.3. However, in the derivations of $\dot{y}_1, \dot{y}_2, \dot{y}_3$, we need the quantities $X/L_{pa}, Y/L_{pa}, Z/L_{pa}$, which can not be written as linear functions of y_1, y_2, y_3 , as can be seen in the following:

$$X/L_{pa} = y_1 \sqrt{y_4/y_3}, \quad Y/L_{pa} = y_2 \sqrt{y_4/y_3}, \quad Z/L_{pa} = \sqrt{y_3 y_4}. \quad (4.155)$$

Denote $y_5 = X/L_{pa}, y_6 = Y/L_{pa}, y_7 = Z/L_{pa}$, we have

$$\begin{cases} \dot{y}_1 = \sum_{i=1}^3 a_{1i} y_i + a_{31} \frac{y_1^2}{y_3} + a_{32} \frac{y_1 y_2}{y_3} + a_{33} y_1 - 2y_1 f_{pa}(\cdot) + (b_3 - 2b_1 y_1) y_5 - 2b_2 y_1 y_6 + b_1 y_7, \\ \dot{y}_2 = \sum_{i=1}^3 a_{2i} y_i + a_{31} \frac{y_1 y_2}{y_3} + a_{32} \frac{y_2^2}{y_3} + a_{33} y_2 - 2y_2 f_{pa}(\cdot) - 2b_1 y_2 y_5 + (b_3 - 2b_2 y_2) y_6 + b_2 y_7, \\ \dot{y}_3 = 2 \left[\sum_{i=1}^3 a_{3i} y_i - y_3 f_{pa}(\cdot) - b_1 y_3 y_5 - b_2 y_3 y_6 + b_3 y_7 \right], \\ \dot{y}_4 = -2y_4 [f_{pa}(\cdot) + a_{23} y_2 + b_1 y_5 + b_2 y_6], \end{cases} \quad (4.156)$$

with

$$f_{pa}(\cdot) = a_{11} \frac{y_1^2}{y_3} + (a_{21} + a_{12}) \frac{y_1 y_2}{y_3} + a_{22} \frac{y_2^2}{y_3} + a_{13} y_1. \quad (4.157)$$

In the above derivations, the following entries have been used:

$$\frac{XY}{L_{pa}} = \frac{y_1 y_2}{y_3}, \quad \frac{X^2}{L_{pa}} = \frac{y_1^2}{y_3}, \quad \frac{Y^2}{L_{pa}} = \frac{y_2^2}{y_3}. \quad (4.158)$$

It can be observed that equation (4.156) is no longer in the form of (4.69). Thus, the IBO observer that can be applied for the range identification using the 3-D planar and ball-shape imaging surfaces as discussed in sections 4.7.1 and 4.7.2 can not be applied directly in the case of a paraboloid imaging surface.

For the 3-D motion estimation and assuming that a feature point is on the plane $Z = pX + qY + r$, we have

$$Z^2/L_{pa} = (pXZ + qYZ + rZ)/L_{pa} \Rightarrow Z/L_{pa} = \frac{1}{r}(y_3 - p y_1 - q y_2). \quad (4.159)$$

In a similar manner as in (4.159), we have

$$\frac{X}{L_{pa}} = \frac{y_1(y_3 - p y_1 - q y_2)}{r y_3}, \quad \frac{Y}{L_{pa}} = \frac{y_2(y_3 - p y_1 - q y_2)}{r y_3}. \quad (4.160)$$

Substituting equations (4.159) and (4.160) into \dot{y}_1, \dot{y}_2 , and \dot{y}_3 in (4.156), we can arrive at a similar equation as (4.145) using a 12×1 essential parameter satisfying the four constraints in (4.146). The final equation is omitted from here due to its straightforward derivation.

4.7.4 Simulation Results

In this section, simulations results are presented for the range identification problem when the imaging surfaces are a traditional camera-type plane (at $Z_p = 1$), a general plane, a sphere, and an ellipsoid. Among several existing nonlinear observers applicable to the range identification problem, the IBO proposed in [23] that is suitable for a general nonlinear system in the form of (4.69) is used in our simulations due to its easy implementation and extension. The state observer proposed in [24] has a similar performance compared to the IBO observer.

The specific affine motion used in our simulation is the affine motion in (4.84) with initial values $[X(0), Y(0), Z(0)] = [0.4, 0.6, 1]$. Choosing the observer parameters to be $M = 10, G = 10, \gamma = 1$, the range identification errors for y_3 in (4.72) and (4.130), and y_4 in (4.141) and (4.150), when using a camera type imaging surface $Z_p = 1$, a general plane with $\mathbf{n} = [1, 1, 1]^T$, a spherical surface with radius 1, and an ellipsoid with $r_1^2 = 1, r_2^2 = 2, r_3^2 = 3$, are shown in fig. 4.26. In the simulations shown in fig. 4.26, we apply a relative uniform noise of level 10^{-3} to the system output according to $y(t) = y^*(t) + |y^*(t)| \cdot \text{noise_level} \cdot \text{randn}(\text{length}(y(t)))$, where $y^*(t)$ denotes the ideal system output and $y(t)$ simulates the observed one.¹¹ From fig. 4.26 it can be seen that when using different imaging surfaces, the applied IBO observer still shows consistent convergence for the state estimation.

4.7.5 Concluding Remarks

In this section, the imaging surface of a vision surface is generalized to several 3-D surfaces, i.e., a plane, a sphere, an ellipsoid, and a paraboloid. The difference in the imaging

¹¹ $\text{length}(\cdot)$ and $\text{randn}(\cdot)$ are Matlab functions giving the length of a vector and random entries of normal distribution with mean zero, variance one and standard deviation one.

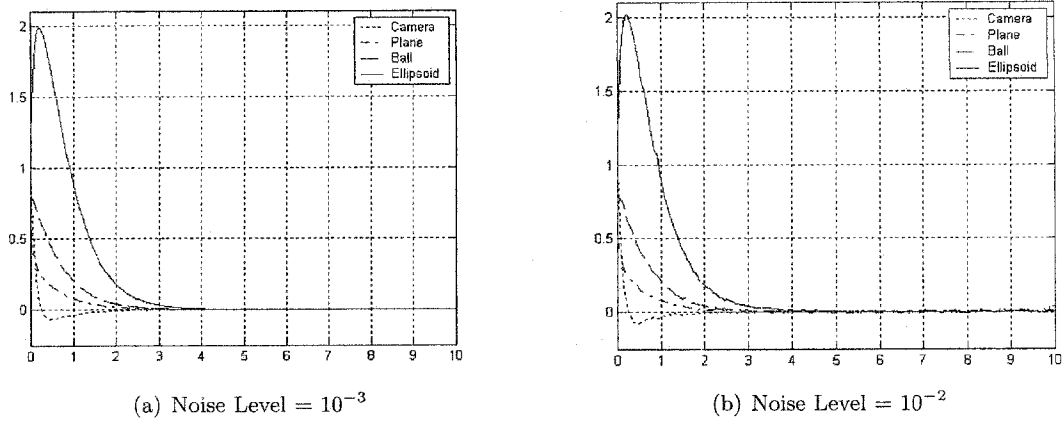


Fig. 4.26: Range identification under general imaging surfaces.

surface affects how the output is chosen, affects the design or utilization of range identification observers, and affect the optical flow dynamics on the imaging surfaces. Since we assume that the omnidirectional systems still possess a single center of projection such that the images observed preserve linear perspective geometry, the underlying nature of perspective projection is not changed. Further, the basic idea to perform range identification and 3-D motion estimation remains the same.

However, different imaging surfaces result in slightly different system structure of the PDS. When the resulting PDS system no longer preserves an affine form, range identification can not be carried out directly via most of the existing nonlinear observers.

4.8 Estimation with Unique Solution

In the setup of a stationary camera observing a moving object, it can be concluded that even for the rigid motion, depth ambiguity is always there. Of course, if either the physical size of the moving object or the depth information at one time instant is given, depth can be recovered uniquely. Further, to identify the motion and/or shape parameters to a unique orbit, ideas of using a pair of cameras that operate in parallel and possibly asynchronously, and integrating vision with range sensor, have been proposed in [22, 107] and [108, 109, 110, 111], respectively.

Using two cameras, the basic strategy is to compute the ambiguity surfaces for each camera, intersect the two surfaces and show that generally the intersection is always a point. In [22], the relative transformation between the two cameras is assumed to be known.

Integration of vision with range sensors, such as a laser, has been discussed for two cases in [108] that depends on the time the laser is applied: 1) the laser is applied after the camera's task and is only applied at one time instant and 2) the laser is applied for a time period after the camera's task. When the range data is integrated with vision, range information provides an additional observation function. A summary table presented in [108, 109] shows that, with a laser, the dimension of the unobservable space is reduced. If the laser is applied in a time interval, motion parameters can be identified uniquely. Theoretical formulation of 3-D motion estimation using vision and range information is discussed in [108, 109]. Actual implementation using EKF is presented in [110, 111], where some guidelines have been given about under what conditions the EKF is more likely to converge than the others. Among the three motions studied, which are moving sideways, moving away, and moving towards the plane, the better movements are usually moving sideways with a distance "big enough" [110].

Though the problem discussed in this section mainly focuses on 3-D motion estimation of a moving object using a single stationary camera, we will briefly discuss the cases when the camera is moving while the object is stationary, or both the camera and the object are moving. Motion estimation problem can be simplified when we assume that the motion of the camera is available, such as the camera's linear and angular velocities assumed in [87, 97], at least referring to the camera's own coordinate system.

For the cases when using an active vision system for the range identification with known motion parameters, the problem is only slightly different, as summarized in table 4.3, where the subscribe o refers to the object dynamics and c refers to the camera. When both the camera and the object are stationary, a feature point on the object is only observable up to a homogeneous line. It is thus labelled as "None" to indicate the ambiguity in depth.

When the camera is moving while the object is stationary, since the positions of the camera are assumed known, depth of a feature point can be derived simply by a triangulation, as shown in fig. 4.27 but with known (R, \mathbf{t}) . The case for a stationary camera observing a moving object has been the focus of this chapter and its depth can be estimated via the nonlinear observers described in section 4.3.2. When both the camera and the object are moving, and assuming known motion parameters of the object (the initial state is unknown) and all information of the camera, information observed on the image sequence is a homogenization of the displacement of the state of the object and the state of the camera. Relative positions between the camera and the object can be derived.

Table 4.3: Range Identification with Known Motion Parameters Using Single Camera

	Object Moving $\dot{x}_o = A_o x_o + b_o$	Object Not Moving
Camera Moving	$\dot{x}_c = A_c x_c + b_c$ $y = [x_o - x_c]$ To estimate x_{o3}	Object position achievable via triangulation, as shown in fig. 4.27
Camera Not Moving	$y = [x_o]$ To estimate x_{o3}	None

Parameter identification of a motion dynamics for the above four cases are also discussed in table 4.4. When the object is stationary, it is basically the corresponding range identification problems discussed in table 4.3. 3-D motion estimation with active vision system has shown to recover the norm of the motion parameters of a rigid motion [87, 96], which are not identifiable using a single stationary camera. For the general Riccati motion, estimating motion parameters uniquely with an active vision is the subject of further research.

In summary, methods towards the unique estimation of motion parameters in the 3-D motion estimation include:

- 1) Using multiple cameras:
 - a) Stereo vision type: Need correspondence between features between cameras to estimate the depth information.

Table 4.4: 3-D Motion Estimation for Unknown Motion Parameters Using Single Camera

	Object Moving	Object Not Moving
Camera Moving	For rich motion, be able to recover the norm of rigid motion	Depth estimation, same as the corresponding case in table 4.3
Camera Not Moving	Can not recover the norm of the states using a single camera	None

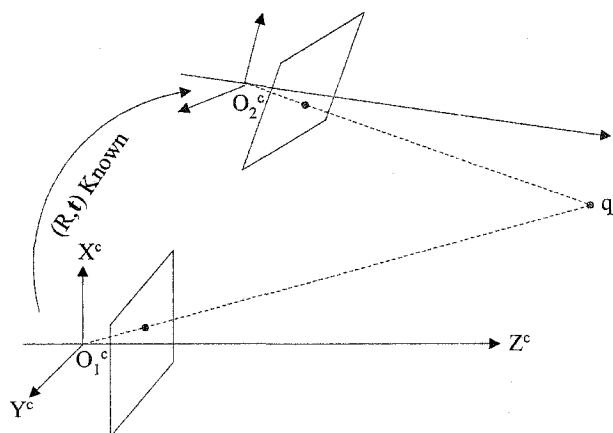


Fig. 4.27: Object location determined via triangulation for a stationary point using moving camera.

- b) Multiple cameras operating asynchronously: Estimate the parameter orbit individually and seek the intersection of the orbits.
- 2) Integration vision with range data: Laser is applied at one time instant or a period after the camera to provide additional observation function.
- 3) Active vision system: Assume motion of the camera is known, such as its linear and angular velocities.
- 4) Assuming constraints in the features observed: Use more features or features of certain types.

We end our discussion by considering several applications. For obstacle avoidance for mobile robot navigation where the robot is equipped with a vision system, both the

obstacle and the camera can be moving. This fits into the category of “Camera Moving + Object Moving.” For the case of “Camera Moving + Object Not Moving,” an example application is video surveillance of overhead power line and pipe lines, where the camera is used to detect the defects. In industrial manufacturing, the camera(s) can be fixed, while the targets to observe can be stationary or moving depending on the task.

4.9 Summary

3-D motion estimation and range identification problems using camera-type vision systems are reviewed in this chapter. Nonlinear observers that are applicable to the range/depth/state estimation of a PDS system have been compared. Based on a recently proposed linear approximation idea, a linear approximation-based perspective nonlinear observer was proposed in section 4.5, which can be applied to a slightly more general nonlinear systems than the existing perspective nonlinear observers. Besides using the full homogeneous outputs for state estimation, single homogeneous output has been shown to achieve similar, but slower, converging pattern in section 4.6. Finally, the motion estimation and range identification problems on more general imaging surfaces were briefly introduced in section 4.7.

Chapter 5

Iterative Learning Control for PDS

In chapter 4, we discussed the 3-D motion and range identification problems of a PDS system. In this chapter, iterative learning control (ILC) of a PDS will be addressed, with some preliminary simulation results given in section 5.2. Finally, Section 5.3 describes an actual feasible experimental platform.

5.1 Introduction of ILC

Iterative learning control, or ILC, is a technique for improving the transient response and tracking performance of processes, machines, equipment, or systems that execute the same trajectory, motion, or operation over and over. The approach is motivated by the observation that if the system controller is fixed and if the system's operating conditions are the same each time it executes, then any errors in the output response will be repeated during each operation. These errors can be recorded during system operation and can then be used to compute modifications to the input signal that will be applied to the system during the next operation. That is, in ILC, refinements are made to the input signal after each trial until the desired performance level is reached. It is usually assumed implicitly that the initial conditions of the system are reset at the beginning of each trial to the same value. In describing the technique of ILC, the word *iterative* is used because of the recursive nature of the system, and the word *learning* is used because of the refinement of the input signal based on past performance in executing a task [26]. Most of the current ILC algorithms use the encoder readings of the plant as the feedback information. In the perspective ILC, vision measurement serves as the actual feedback. To date, there are few contribution in the ILC literature where vision feedback is used. Further, there have been no contribution that exploits the PDS theory in an ILC problem.

5.2 ILC Control of PDS

Iterative learning control of both linear and nonlinear systems has been studied in the literature. When the global Lipschitz condition is assumed, either in the system dynamics or in the control mechanism, the conventional contraction mapping method can be applicable. The contraction mapping method allows us to completely ignore the system dynamic part and most ILC schemes proposed so far are of simple linear type. To widen the learning control framework under which ILC can handle broader classes of system nonlinearities, such as local Lipschitz continuous ones, the use of Composite Energy Function (CEF) has been proposed in [112]. Motivated from the Lyapunov function, the CEF ILC updating law includes a term similar to the Lyapunov function and a term to capture the learning process in the iteration domain.

Consider an object that is moving along a 3-D trajectory iteratively in the 3-D space whose motion is observed via a camera-type vision system. This problem is basically an ILC control problem of a PDS system. The first question in concern is whether the observations on the image plane, a 2-D projection trajectory, suffices to refine the 3-D motion trajectory. Our preliminary study shows that the homogeneous output information from the image plane can help to improve the system performance, under the precondition that the initial position of the plant (the moving target) is identical with the desired position. The above precondition is referred to as the Identical Initial Condition (IIC) in the ILC literature, which has caused a lot criticism yet remains an important condition for most ILC schemes.

The next question is if this IIC condition can be relaxed and at what expenses. From chapter 4, when the motion parameters of a 3-D moving object are known, perspective nonlinear observers can be applied to estimate the system states from its homogeneous outputs. An intuitive idea is to estimate the system's states, based on which the ILC scheme can be built. However, this requires one additional assumption: that the motion parameters of the PDS are exactly known. So far, we have not arrived at a conclusion on the observer-based ILC control of a PDS system, which will be included in the future investigations.

5.2.1 Problem Formulation

Definition 5.2.1 General Perspective ILC Formulation:

Given:

- 1) *Object moving according to a known motion.*
- 2) *Desired observed trajectory (u_d, v_d) on the image plane of a stationary camera.*
- 3) *Calibrated camera whose parameters, such as the camera's intrinsic parameters, distortion coefficients, and focal length, are known.*

Task: *Track $(u_d(t), v_d(t))$, along with the 3-D desired trajectory, on the image plane and the desired path on the object plane in the presence of uncertainties, where uncertainties could arise from calibration of the camera, measurements on the image plane, and about controlling the plant.*

Additional Condition: *With or without the identical initial condition (IIC).*

5.2.2 Preliminary Study

The problem formulation in Definition. 5.2.1 states the general problem regarding the ILC control of a PDS system. In the section, some preliminary simulation results are given in the ideal situation of no noise with or without the IIC condition. These simulation results show that, with the assumption of the identical initial condition, the homogeneous output information can be enough to make the moving object track a desired 3-D trajectory.

Consider a 2-D PDS system described as follows:

- 1) Motion Dynamics:

$$\begin{bmatrix} \dot{X}(t) \\ \dot{Y}(t) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} X(t) \\ Y(t) \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} u(t). \quad (5.1)$$

- 2) Homogeneous Output:

$$y(t) = [y_1(t)] = [X(t)/Y(t)], \quad (5.2)$$

where the output is assumed to be available from a calibrated 1-D camera.

The specific problem is to control the plant to track a desired 2-D trajectory using information $y(t) = X(t)/Y(t)$, which is the information directly available from the image plane.

Before theoretically proving the feasibility, the following simulations are conducted that give an intuitive idea of the problem. Let the motion parameters $[a_{i,j}, b_j]$ for $i, j = 1, 2$ in (5.1) be

$$\begin{aligned} A &= \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} -0.1 & 0.3 \\ 0.1 & -0.2 \end{bmatrix}, \\ B &= \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 0.15 \\ 0.2 \end{bmatrix}, \end{aligned} \quad (5.3)$$

$$(X_d(0), Y_d(0)) = (X(0), Y(0)) = (0.15, 0.2).$$

The desired 2-D trajectory is generated by a sinusoidal input $u_d(t) = \sin(t)$. Suppose for the first iteration, $u_1(t)$ is taken as the step input $u_1(t) = 1$. Apply the following ILC updating law

$$\begin{aligned} u_{k+1}(t) &= u_k(t) + k_1 e(t) + k_2 \dot{e}(t), \\ e(t) &= y_d(t) - y_k(t), \end{aligned} \quad (5.4)$$

where k_1 and k_2 are chosen to be

$$k_1 = k_2 = -0.4. \quad (5.5)$$

Notice that the learning rates in (5.5) are chosen by trial and error. Detailed procedures to get the proper learning rate need to be investigated and this will be included in the future investigations.

Figure 5.1 shows the results of the 41 iterations. The two plots in the first row of fig. 5.1 show the homogeneous output $y_1(t)$ and the 2-D desired trajectory with those of the first iteration using $u_1(t) = 1$, where the desired curves are plotted in red and the actual in blue. The second row in fig. 5.1 show the corresponding plots after 41 iterations with the ILC updating law (5.4) using learning rates (5.5). It can be observed that the ILC updating law in (5.4) helps to make the system output, along with the 2-D states, to track their desired values.

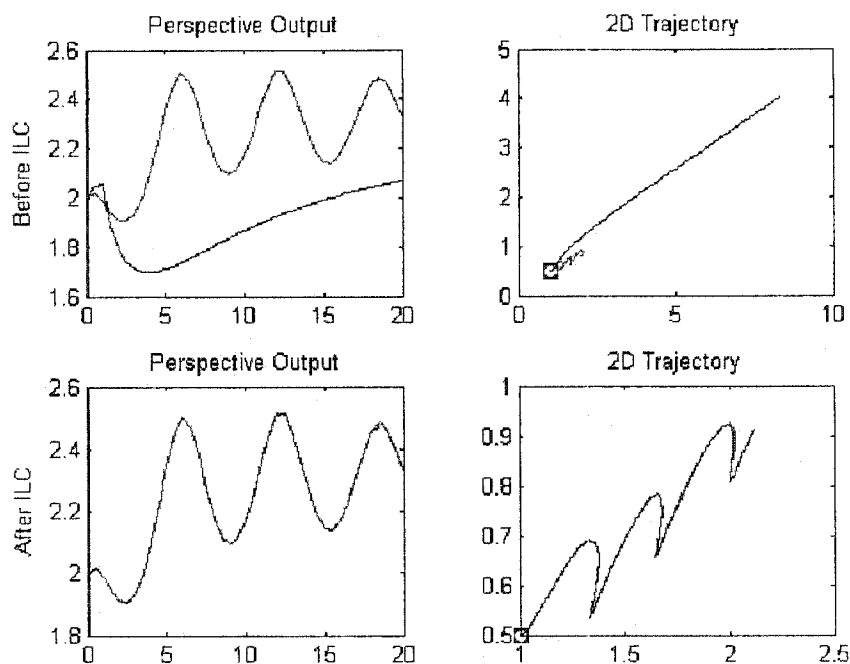


Fig. 5.1: Simulation results of the ILC control of a 2-D PDS system (under IIC).

Remember that the results shown in fig. 5.1 are under the condition of IIC. When the IIC condition is not satisfied, that is $(X_d(0), Y_d(0)) \neq (X(0), Y(0))$, will the tracking of the output $y_1(t)$ guarantee the tracking of the 2-D states? An intuitive thinking of this question gives the answer “no,” which is supported by the simulation results shown in fig. 5.2 with $(X(0), Y(0)) = (1.2, 0.9)$.

From fig. 5.2, it can be observed that, using the ILC updating law (5.4), when the IIC condition cannot be satisfied, tracking in the perspective output cannot guarantee the tracking the 2-D states. That is, in this case, only the ratio $Y(t)/X(t)$ tracks the desired one, instead of $X(t)$ and $Y(t)$ track $X_d(t)$ and $Y_d(t)$ individually, as can be seen from the “2D Trajectory” in fig. 5.2 after the ILC iterations. With or without the IIC condition, the $\|e_k\|_2 = \|y_d - y_k\|_2$ both show a monotone convergence in fig. 5.3.

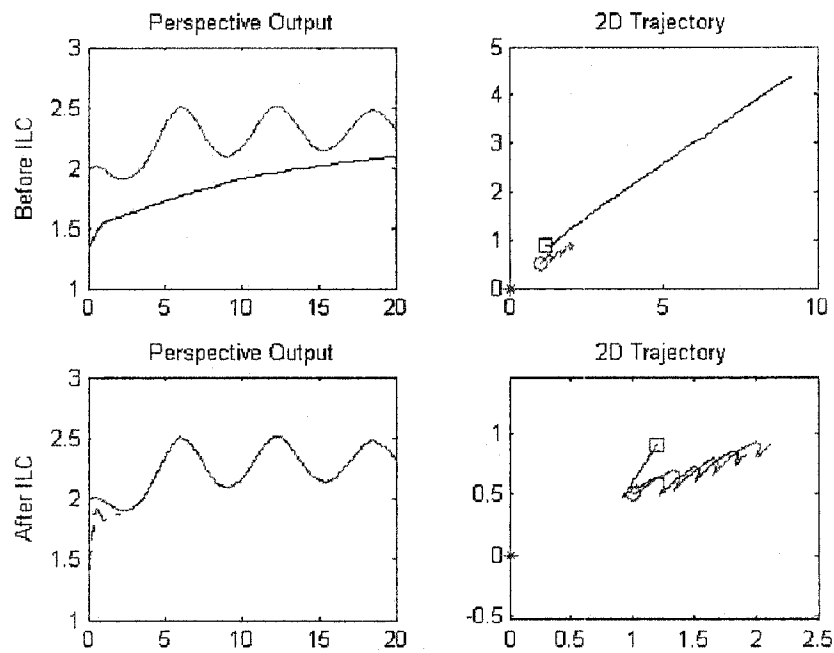


Fig. 5.2: Simulation results of the ILC control of a 2-D PDS system (without IIC).

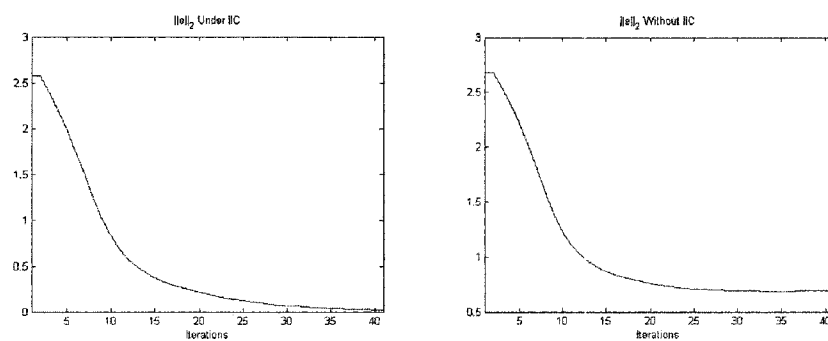


Fig. 5.3: Simulation results of $\|e_k\|_2 = \|y_d - y_k\|_2$ with or without the IIC condition.

5.2.3 Proof

Sections 5.2.1 and 5.2.2 give the simulation results of the ILC control of a 2-D PDS system. In this section, a proof of the convergence of the ILC control law (5.5) for a PDS system is presented.

From [113], we have the following Corollary:

Corollary 5.2.1 *Suppose that the plant is described by:*

$$\begin{aligned}\dot{x}(t) &= a(t, x) + B_s(t)u(t), \\ y(t) &= C_s x(t),\end{aligned}\tag{5.6}$$

where C_s is a constant matrix, $B_s(t)$ depends on the state variable x that is bounded, and $a(t, x)$ satisfies the condition

$$\|a(t, x_1) - a(t, x_2)\| \leq \|x_1 - x_2\|.\tag{5.7}$$

Define

$$e_k(t) = \dot{y}_d(t) - \dot{y}_k(t)\tag{5.8}$$

and the following learning control scheme

$$\begin{aligned}\dot{v}_k(t) &= A_c(t)v_k(t) + B_c(t)e_k(t), \\ w_k(t) &= C_c(t)v_k(t) + D_c(t)e_k(t), \\ u_{k+1}(t) &= u_k(t) + w_k(t),\end{aligned}\tag{5.9}$$

is to be applied to this plant. Then, if

$$\|I_m - C_s B_s(t) D_c(t)\| \leq 1, \quad \forall t \in [0, T],\tag{5.10}$$

holds, the error defined by (5.8) converges to zero in the sense of

$$\|e_{k+1}(t)\|_q \leq b_0 \|e_k(t)\|,\tag{5.11}$$

where I_m denotes the m -dimensional identity matrix, $0 \leq b_0 \leq 1$, and $q \geq 0$. The above equation implies

$$\|e_k(t)\|_q \rightarrow 0, \quad \text{as } k \rightarrow \infty.\tag{5.12}$$

Before using the above corollary in the proof, we first calculate the derivative of $y(t) = (y_1(t), y_2(t))$ from (5.1) and (5.2):

$$\begin{aligned}\dot{y}_1 &= a_{12} + (a_{11} - a_{22})y_1 - a_{21}y_1^2 + (b_1 - b_2y_1)y_2 u(t), \\ \dot{y}_2 &= -(a_{22} + a_{21}y_1)y_2 - b_2y_2^2 u(t),\end{aligned}\tag{5.13}$$

Substituting the symbol y with x in the above equation, we have:

$$\begin{aligned}\dot{x}_1 &= a_{12} + (a_{11} - a_{22})x_1 - a_{21}x_1^2 + (b_1 - b_2x_1)x_2 u(t), \\ \dot{x}_2 &= -(a_{22} + a_{21}y_1)x_2 - b_2x_2^2 u(t), \\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},\end{aligned}\tag{5.14}$$

which is in the form of (5.6). Thus, in order to prove that the ILC updating law (5.4) makes the system (5.1) converge in the sense of (5.11), we only need to show that the ILC updating law in (5.4) is equivalent to, or a special case of, that in (5.9) with the $e_k(t)$ defined in (5.8).

Compare (5.9) with (5.4). By letting

$$A_c(t) = 0, \quad Bc(t)Cc(t) = k_1, \quad Dc(t) = k_2,\tag{5.15}$$

the applied ILC updating law in (5.4) is a special form of (5.9). Thus, we can be sure that (5.4) will work.

5.3 Experimental Setup

A feasible experimental setup available is a 2-D gimbal system, as shown in fig. 1.3. We put a laser pointer on the gimbal system, whose projection on the wall is observed by a stationary camera (illustrated in fig. 5.4). To test the idea that, under the ILC condition, tracking in the system's states can be guaranteed by the tracking in its homogeneous observations on the image plane of the camera, we can put a desired trajectory on the wall. The goal is to control the gimbal such that the hooked laser pointer tracks the attached path on the wall.

The ILC control via vision-feedback information has been partially validated in [114]. In [114], the stationary camera is required to be placed parallel to the 2-D projection plane, which is the wall in our experimental setup. In this section, we are discussing the situation that the camera can be placed not so restrictively.

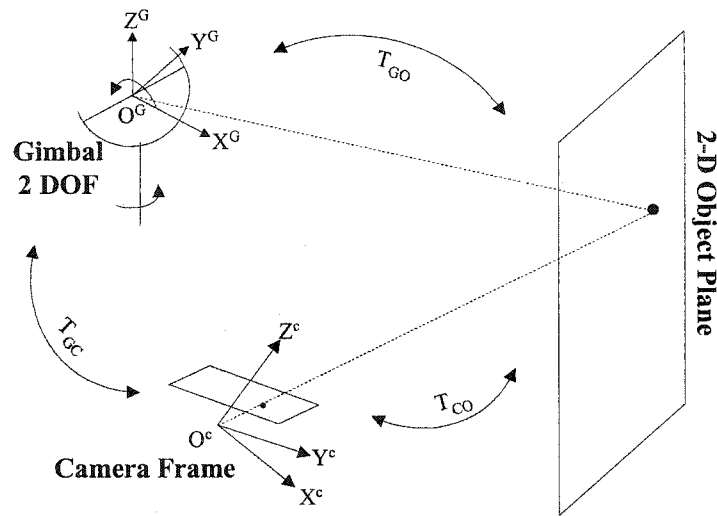


Fig. 5.4: Experimental setup for perspective ILC using gimbal system.

5.4 Summary

This chapter discusses the iterative learning control (ILC) of a perspective dynamic system (PDS). Our preliminary study shows that, under the identical initial condition (IIC), tracking in the homogeneous outputs assures tracking in the system's states. However, when the IIC condition is violated, the above statement is no longer true. PDS-based state estimation might be helpful for the task of tracking in the system's states. This remains our future research problem.

Chapter 6

Conclusions

6.1 Contributions

The specific contributions are listed in the order of appearance:¹

- 1) Wireless iterative visual servoing using an uncalibrated camera.
- 2) Camera lens distortion modeling and correction:
 - A class of rational functions for radial distortion modeling.
 - Piecewise smooth idea applied to radial distortion modeling.
 - Evaluation and validation of distortion calibration using the geometric AIC and/or geometric MDL criteria, the idea that straight lines have to be straight, and the calibrated distortion curves.
 - Simplified distortion modeling.
 - Blind removal of lens geometric distortion using higher order spectral analysis.
- 3) Perspective theory and perspective nonlinear observers for the range identification problem:
 - Literature review of existing 3-D motion estimation algorithms.
 - Comparative study of existing perspective nonlinear observers for the range identification problem.
 - Linear approximation-based nonlinear observers applicable to the range identification problem.

¹In addition to these vision-based contributions, Appendix A also contributes a specific idea for laser/sonar fusion for HIMM and describes our implementation of template-based object recognition.

- Range identification with single homogeneous observation.
- Extension of camera-type imaging surface to general planar and spherical imaging surfaces.

4) Perspective ILC:

- Vision system serves as actual feedback to iterative learning control problem.
- Tracking in the homogeneous output can guarantee the tracking in the system's states under the assumption of identical initial condition.
- The calibrated camera can be placed flexibly.

6.2 Future Directions

Some of the future directions of the research topics concerned in this dissertation have been described in the corresponding chapters. In this section, we emphasis on the Perspective Dynamic Systems (PDS) and Iterative Learning Control of the PDS system, which have not been solved completely in this dissertation, and also not in the literature.

6.2.1 Perspective Dynamic Systems

The 3-D motion estimation and range identification problems described in chapter 4 is mainly focused on the PDS framework. However, only the range identification problem is extensively studied. 3-D motion estimation has not been performed. Besides, all the results presented are simulations. The application of the described algorithms to on-line real image sequences needs to be carried out. Moreover, as already mentioned in section 4.5, detailed concerns about the LAO observer include:

- 1) Design of less restrictive LTV observers.
- 2) Application of the proposed LAO observer to single homogeneous PDS system.

6.2.2 Perspective ILC

Section 5.2 provides a preliminary study of the ILC control of a PDS system. The following issues are very important and need to be considered. However, these will only be included in our future investigations:

- 1) Regarding the IIC condition: Will the perspective observer discussed in chapter 4 help for the ILC control of the PDS when the condition of IIC is not satisfied?
- 2) Regarding the learning rates: The learning rates specified in (5.5) for the PDS system (5.3) are simply obtained by trial and error. Theoretical procedures need to be provided.
- 3) Regarding the ILC updating law: Theoretically, for a system of relative degree one, an ILC updating law in the form of

$$u_{k+1}(t) = u_k(t) + k(\dot{y}_d(t) - \dot{y}_k(t))$$

is enough. However, during our simulation, we found that an additional term of $(y_d(t) - y_k(t))$ helps to improve the tracking performance. Some analysis needs to be carried out on this issue.

- 4) Regarding uncertainty: The ILC scheme is well-known to be useful at the existence of measurement noise and model uncertainties. The preliminary simulation results in section 5.2.2 is conducted under the ideal case of no noise, which is far from an actual application situation. The ILC control of a PDS system needs to be considered in the existence of measurement noise.
- 5) Regarding camera calibration: On the so far discussed vision-based control tasks, we all assume that the camera has been calibrated perfectly beforehand. However, when the camera is not properly calibrated, what is its influence on the resulting control performance?

- 6) Regarding experimental verification: The preliminary study presented in chapter 5 is only based on simulations. Experimental verification using the gimbal system shown in fig. 1.3 needs to be carried out.

References

- [1] CSOIS, "Homepage," <http://www.csois.usu.edu>.
- [2] M. Berkemeier, M. Davidson, V. Bahl, Y. Chen, and L. Ma, "Visual servoing of an omni-directional mobile robot for alignment with parking lot lines," in *IEEE International Conference on Robotics and Automation*, 2002.
- [3] L. Ma, M. Berkemeier, Y. Chen, M. Davidson, V. Bahl, and K. L. Moore, "Wireless visual servoing for ODIS - an under car inspection mobile robot," in *IFAC World Congress*, Spain, 2002.
- [4] K. L. Moore, *Iterative Learning Control for Deterministic Systems*. London: Springer-Verlag, 1993.
- [5] 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, Oct. 1996.
- [6] E. Malis, F. Chaumette, and S. Boudet, "2-1/2-D visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 2, pp. 238–250, Apr. 1999.
- [7] E. Malis and F. Chaumette, "2 1/2 D visual servoing with respect to unknown objects through a new estimation scheme of camera displacement," *International Journal of Computer Vision*, vol. 37, no. 1, pp. 79–97, 2000.
- [8] R. K. Lenz and R. Y. Tsai, "Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 5, pp. 713–720, Sept. 1988.
- [9] F. Devernay and O. Faugeras, "Straight lines have to be straight," *Machine Vision and Applications*, vol. 13, no. 1, pp. 14–24, 2001.

- [10] 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. 3, no. 4, pp. 323–344, Aug. 1987.
- [11] C. C. Slama, Ed., *Manual of Photogrammetry*, 4th ed. Falls Church: American Society of Photogrammetry, 1980.
- [12] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientation," *IEEE International Conference on Computer Vision*, pp. 666–673, Sept. 1999.
- [13] J. Heikkilä and O. Silvén, "A four-step camera calibration procedure with implicit image correction," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1106–1112, San Juan, Puerto Rico, 1997.
- [14] J. Heikkilä and O. Silvén, "Calibration procedure for short focal length off-the-shelf CCD cameras," in *Proceedings of 13th International Conference on Pattern Recognition*, pp. 166–170, Vienna, Austria, 1996.
- [15] H. Farid and A. C. Popescu, "Blind removal of lens distortion," *Journal of the Optical Society of America A, Optics, Image Science, and Vision*, vol. 18, no. 9, pp. 2072–2078, Sept. 2001.
- [16] S. Soatto, R. Frezza, and P. Perona, "Motion estimation via dynamic vision," *IEEE Transactions on Automatic Control*, vol. 41, no. 3, pp. 393–413, Dec. 1996.
- [17] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 932–946, July 2002.
- [18] T. Papadimitriou, K. I. Diamantaras, M. G. Strintzis, and M. Roumeliotis, "Robust estimation of rigid-body 3-D motion parameters based on point correspondences," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 4, pp. 541–549, June 2000.

- [19] T. J. Broida, S. Chandrashekar, 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. 639–665, July 1990.
- [20] B. K. Ghosh, M. Jankovic, and Y. T. Wu, "Perspective problems in system theory and its application to machine vision," *Journal of Mathematical Systems, Estimation and Control*, vol. 4, no. 1, pp. 3–38, 1994.
- [21] B. K. Ghosh and E. P. Loucks, "A perspective theory for motion and shape estimation in machine vision," *SIAM Journal of Control and Optimization*, vol. 35, no. 5, pp. 1530–1559, 1995.
- [22] B. K. Ghosh, H. Inaba, and S. Takahashi, "Identification of Riccati dynamics under perspective and orthographic observations," *IEEE Transactions on Automatic Control*, vol. 45, no. 7, pp. 1267–1278, July 2000.
- [23] M. Jankovic and B. K. Ghosh, "Visually guided ranging from observations of points, lines and curves via an identifier based nonlinear observer," *Systems and Control Letters*, vol. 25, pp. 63–73, 1995.
- [24] X. Chen and H. Kano, "A new state observer for perspective systems," *IEEE Transactions on Automatic Control*, vol. 47, no. 4, pp. 658–663, Apr. 2002.
- [25] W. E. Dixon, Y. Fang, D. M. Dawson, and T. J. Flynn, "Range identification for perspective vision systems," in *Proceedings of the American Control Conference*, pp. 3448–3453, Denver, Colorado, U.S.A., June 4–6 2003.
- [26] K. L. Moore, "Iterative learning control: An expository overview," *Applied and Computational Controls, Signal Processing, and Circuits*, vol. 1, no. 1, pp. 425–488, 1998.
- [27] L. Ma, Y. Chen, and K. L. Moore, "Rational radial distortion models of camera lenses with analytical solution for distortion correction," *International Journal of Information Acquisition*, To Appear.

- [28] L. Ma and Y. Chen, "Blind detection and compensation of camera lens geometrical distortions," in *SIAM Imaging Science*, Salt Lake City, USA, <http://arXiv.org/abs/cs.CV/0405095>, May 3-5 2004.
- [29] L. Ma, Y. Chen, and K. L. Moore, "Flexible camera calibration using a new analytical radial undistortion formula with application to mobile robot localization," in *IEEE International Symposium on Intelligent Control*, Houston, Oct. 2003.
- [30] L. Ma, Y. Chen, and K. L. Moore, "Range identification for perspective dynamic systems using linear approximation," in *IEEE International Conference on Robotics and Automation*, New Orleans, April 26-May 1 2004.
- [31] L. Ma, Y. Chen, and K. L. Moore, "Range identification for perspective dynamic system with single homogeneous observation," in *IEEE International Conference on Robotics and Automation*, New Orleans, April 26-May 1 2004.
- [32] R. Zhu, Y. Zhang, and Q. Bao, "A novel intelligent strategy for improving measurement precision of FOG," *IEEE Transactions on Instrumentation and Measurement*, vol. 49, no. 6, pp. 1183-1188, Dec. 2000.
- [33] E. M. Mouaddib and B. Marhic, "Geometrical matching for mobile robot localization," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 5, pp. 542-552, Oct. 2000.
- [34] I. Shimshoni, "On mobile robot localization from landmark bearings," in *Proceedings of the 2001 IEEE Conference on Robotics and Automation*, pp. 3605-3610, Seoul, Korea, May 21-26 2001.
- [35] N. S. Flann, M. Davidson, J. Martin, and K. L. Moore, "Intelligent behavior generation strategy for autonomous vehicles using a grammar-based approach," in *Proceedings of 3rd International Conference on Field and Service Robotics*, p. CDROM:174.pdf, Helsinki University of Technology, Otaniemi, Espoo, Finland, June 11 -13 2001.

- [36] P. Jiang, H. Chen, and Y. Wang, "Robot visual servoing for curve tracking," in *Proceedings of the 13th IFAC World Congress*, pp. 337–342, San Francisco, USA, 1996.
- [37] Y. Shen, Y.-H. Liu, K. Li, J. Zhang, and A. Knoll, "Aymptotic motion control of robot manipulators using uncalibrated visual feedback," in *Proceedings of the 2001 IEEE Conference on Robotics and Automation*, pp. 241–246, Seoul, Korea, May 21–26 2001.
- [38] Y. Mezouar and F. Chaumette, "Design and tracking of desirable trajectories in the image space by integrating mechanical and visibility constraints," in *Proceedings of the 2001 IEEE Conference on Robotics and Automation*, pp. 731–736, Seoul, Korea, May 21–26 2001.
- [39] G. X. Ritter and J. N. Wilson, *Handbook of Computer Vision Algorithms in Image Algebra*. Boca Raton: CRC Press, 2000.
- [40] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, Sept. 1996.
- [41] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*. New Jersey: Prentice Hall, 1998.
- [42] 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, Oct. 1992.
- [43] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton: CRC Press, 1994.
- [44] G. Wei and S. Ma, "Implicit and explicit camera calibration: Theory and experiments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 469–480, May 1994.

- [45] R. Hartley and A. Zisserman, *Multiple View Geometry*, 2nd ed. Cambridge, UK: Cambridge University Press, 2000.
- [46] A. W. Fitzgibbon, "Simultaneous linear estimation of multiple view geometry and lens distortion," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 125–132, Kauai, Hawaii, Dec. 2001.
- [47] C. Brauer-Burchardt and K. Voss, "A new algorithm to correct fish-eye- and strong wide-angle-lens-distortion from single images," in *International Conference on Image Processing*, pp. 225–228, Oct. 2001.
- [48] R. J. Mierlo, "ODIS and the use of computer vision," CSOIS Technical Report, Department of Electrical and Computer Engineering, Utah State University, 2001.
- [49] L. Ma, "Localization using yellow line," CSOIS Technical Report, Department of Electrical and Computer Engineering, Utah State University, 2001.
- [50] D. K. Katsoulas and D. I. Kosmopoulos, "An efficient depalletizing system based on 2D range imagery," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 305–312. IEEE, 2001.
- [51] N. S. Flann, K. L. Moore, and L. Ma, "A small mobile robot for security and inspection operations," in *Proceedings of 1st IFAC Conference on Telematics Applications in Automation and Robotics*, pp. 1–6. Weingarten, Germany: IFAC, July 2001.
- [52] J. Heikkila, "Geometric camera calibration using circular control points," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1066–1077, Oct. 2000.
- [53] T. Tamaki, T. Yamamura, and N. Ohnishi, "Unified approach to image distortion," in *International Conference on Pattern Recognition*, pp. 584–587, Aug. 2002.
- [54] C. E. Pearson, Ed., *Handbook of Applied Mathematics: Selected Results and Methods*, 2nd ed. New York: Van Nostrand Reinhold Company, 1983.

- [55] Z. Zhang, "On the epipolar geometry between two images with lens distortion," in *International Conference on Pattern Recognition*, pp. 407–411, Vienna, Aug. 1996.
- [56] B. Tordoff, *Active control of zoom for computer vision*. Ph.D. dissertation, University of Oxford, Robotics Research Group, Department of Engineering Science, 2002.
- [57] K. Kanatani, "Model selection for geometric inference," in *The 5th Asian Conference on Computer Vision*, Jan. 2002.
- [58] M. T. El-Melegy and A. A. Farag, "Nonmetric lens distortion calibration: Closed-form solutions, robust estimation and model selection," in *IEEE International Conference on Computer Vision*, Nice, France, 2003.
- [59] Z. Zhang, "Experimental data and result for camera calibration," Microsoft Research Technical Report, <http://rese-arch.microsoft.com/~zhang/calib/>, 1998.
- [60] L. Ma, "Camera calibration: a USU implementation," CSOIS Technical Report, ECE Department, Utah State University, <http://arXiv.org/abs/cs.CV/0307072>, May, 2002.
- [61] L. Ma, Y. Chen, and K. L. Moore, "A family of simplified geometric distortion models for camera calibration," <http://arxiv.org/abs/cs.CV/0308003>, 2003.
- [62] J. M. Mendel, "Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications," *Proceedings of the IEEE*, vol. 79, no. 3, pp. 278–305, Mar. 1991.
- [63] C. P. Lu, G. D. Hager, and E. Mjolsness, "Fast and globally convergent pose estimation from video images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 610–622, June 2000.
- [64] B. K. Ghosh and C. F. Martin, "Homogeneous dynamical systems theory," *IEEE Transactions on Automatic Control*, vol. 47, no. 3, pp. 462–472, Mar. 2002.

- [65] J. K. Aggarwal and N. Nandhakumar, "On the computation of motion from sequences of images - A review," *Proceedings of the IEEE*, vol. 76, no. 8, pp. 917-935, Aug. 1988.
- [66] C. Stiller and J. Konrad, "Estimating motion in image sequences, a tutorial on modeling and computation of 2D motion," *IEEE Signal Processing Magazine*, pp. 70-91, July 1999.
- [67] T. S. Huang and A. N. Netravali, "Motion and structure from feature correspondences: A review," *Proceedings of the IEEE*, vol. 82, no. 2, pp. 252-268, Feb. 1994.
- [68] T. Chen, W.-C. Lin, and C.-T. Chen, "Artificial neural networks for 3-D motion analysis - part II: Nonrigid motion," *IEEE Transactions on Neural Networks*, vol. 6, no. 6, pp. 1394-1401, Nov. 1995.
- [69] H. R. Cho, K. M. Lee, and S. U. Lee, "A new robust 3D motion estimation under perspective projection," in *IEEE International Conference on Image Processing*, pp. 660-663, 2001.
- [70] K. I. Diamantaras, T. Papadimitrion, M. G. Strintzis, and M. Roumeliotis, "Total least squares 3-D motion estimation," in *IEEE International Conference on Image Processing*, pp. 923-927, 1998.
- [71] K. I. Diamantaras and M. G. Strintzis, "Camera motion parameter recovery under perspective projection," in *IEEE International Conference on Image Processing*, pp. 807-810, 1996.
- [72] A. M. Tekalp, *Digital Video Processing*, 1st ed. New Jersey: Prentice Hall PTR, 1995.
- [73] Y. T. Wu, *Estimating three-dimensional motion parameters from feature-based image sequence: a perspective-system approach*. Ph.D. dissertation, Washington University, 1991.

- [74] S. Soatto, R. Frezza, and P. Perona, "Motion estimation on the essential manifold," in *Proceedings of the European Conference on Computer Vision*, pp. 61–72, London, 1994.
- [75] S. Soatto, P. Perona, R. Frezza, and G. Picci, "Motion estimation via dynamic vision," in *Proceedings of the IEEE International Conference on Decision and Control*, 1994.
- [76] G. Calvagno, R. Rinaldo, and L. Sbaiz, "Three-dimensional motion estimation of objects for video coding," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 1, pp. 86–97, Jan. 1998.
- [77] T. Chen, W.-C. Lin, and C.-T. Chen, "Artificial neural networks for 3-D nonrigid motion analysis," in *International Joint Conference on Neural Network*, pp. 420–425, June 1992.
- [78] T. Chen, W.-C. Lin, and C.-T. Chen, "Artificial neural networks for 3-D motion analysis - part I: Rigid motion," *IEEE Transactions on Neural Networks*, vol. 6, no. 6, pp. 1386–1393, Nov. 1995.
- [79] D. Tzovaras, N. Ploskas, and M. G. Strintzis, "Rigid 3-D motion estimation using neural networks and initially estimated 2-D motion data," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 1, pp. 158–165, Feb. 2000.
- [80] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.
- [81] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice*. New Jersey: Prentice Hall, Information and Systems Sciences Series, 1993.
- [82] S. D. Blostein and R. M. Chann, "The use of image plane velocity measurements in recursive 3-D motion estimation from a monocular image sequence," in *IEEE International Conference on Electrical and Computer Engineering*, pp. 797–801, Sept. 1994.

- [83] A. Chiuso, P. Favaro, H. Jin, and S. Soatto, "Structure from motion causally integrated over time," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 523–535, Apr. 2002.
- [84] A. Chiuso, P. Favaro, and H. Jin, "3-D motion and structure from 2-D motion causally integrated over time: Implementation," in *Proceedings of the European Conference on Computer Vision*, pp. 734–750, 2000.
- [85] B. K. Ghosh, M. Jankovic, and Y. T. Wu, "Some problems in perspective system theory and its application to machine vision," in *International Conference on Intelligent Robots and Systems*, pp. 139–146, July 1992.
- [86] B. K. Ghosh, M. Jankovic, and E. P. Loucks, "On the problem of observing motion and shape," in *IEEE International Conference on Automation, Robotics, and Computer Vision*, pp. 653–657, Singapore, Nov. 1994.
- [87] J. P. Hespanha, "State estimation and control for systems with perspective outputs," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 2208–2213, Las Vegas, Nevada, USA, Dec. 2002.
- [88] S. Takahashi, B. K. Ghosh, and S. Louis, "Canonical forms and parameter identification problems in perspective systems," *Fourth Special Issue on Linear Systems and Controls, Linear Algebra and Its Applications*, pp. 701–717, 2002.
- [89] S. Takahashi and B. K. Ghosh, "Canonical forms and orbit identification problems in machine vision," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 5175–5181, 2001.
- [90] S. Takahashi, B. K. Ghosh, and H. Inaba, "Kronecker canonical forms for perspective systems with applications to parameter identification," in *Proceedings of the American Control Conference*, pp. 2215–2219, 2000.

- [91] T. Loucks, B. K. Ghosh, and J. Lund, "An optical flow based approach for motion and shape parameter estimation in computer vision," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 819–823, 1992.
- [92] A. Teel, R. Kadiyala, P. Kokotovic, and S. Sastry, "Indirect techniques for adaptive input-output linearization of non-linear systems," *International Journal of Control*, vol. 53, no. 1, pp. 193–222, 1991.
- [93] B. K. Ghosh, E. P. Loucks, and M. Jankovic, "An introduction to perspective observability and recursive identification problems in machine vision," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 3229–3234, 1994.
- [94] K. Kanatani, *Group-Theoretical Methods in Image Understanding*. London: Springer Verlag, 1990.
- [95] H. Inaba, A. Yoshida, R. Abdursul, and B. K. Ghosh, "Observability of perspective dynamical systems," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 5157–5162, Sydney, Dec. 12-15 2000.
- [96] J. P. Hespanha, "State estimation and control for systems with perspective outputs," University of California, Santa Barbara, Tech. Rep., Feb. 2002.
- [97] A. P. Aguiar and J. P. Hespanha, "Minimum-energy state estimation for systems with perspective outputs and state constraints," in *Proceedings of the IEEE Conference on Decision and Control*, Dec. 2003.
- [98] Y. Fang, M. Feemster, D. Dawson, and N. Jalili, "Active interaction force identification for atomic force microscope applications," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 3678–3683, Las Vegas, Nevada, USA, Dec. 2002.
- [99] X. Xia and M. Zeitz, "On nonlinear continuous observers," *International Journal of Control*, vol. 66, no. 6, pp. 943–954, 1997.

- [100] C. N. Hernandez, S. P. Banks, and M. Aldeen, "Observer design for nonlinear systems using linear approximations," *IMA Journal of Mathematical Control and Information*, vol. 20, pp. 359–370, 2003.
- [101] M. Tomas-Rodriguez and S. P. Banks, "Linear approximations to nonlinear dynamical systems with applications to stability and spectral theory," *IMA Journal of Mathematical Control and Information*, vol. 20, pp. 89–103, 2003.
- [102] C. Nguyen and T. Lee, "Design of a state estimator for a class of time-varying multivariable systems," *IEEE Transactions on Automatic Control*, vol. 30, no. 2, pp. 179–182, Feb. 1985.
- [103] C. Geyer and K. Daniilidis, "Catadioptric projective geometry," *International Journal of Computer Vision*, vol. 45, no. 3, pp. 223–243, 2001.
- [104] R. Swaminathan, M. D. Grossberg, and S. K. Nayar, "A perspective on distortions," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 2003.
- [105] S. K. Nayar, "Catadioptric omnidirectional camera," in *IEEE International Conference on Computer Vision and Pattern Recognition*, Puerto Rico, June 1997.
- [106] W. E. Dixon, Y. Fang, D. M. Dawson, and T. J. Flynn, "Range identification for perspective vision systems," *IEEE Transactions on Automatic Control*, vol. 48, no. 12, pp. 2232–2238, 2003.
- [107] S. Takahashi and B. K. Ghosh, "Motion and shape parameters identification with vision and range," in *Proceedings of the American Control Conference*, pp. 4626–4631, Arlington, VA, June 25–27 2001.
- [108] S. Takahashi and B. K. Ghosh, "Motion and shape identification with vision and range," *IEEE Transactions on Automatic Control*, vol. 47, no. 8, pp. 1392–1396, Aug. 2002.

- [109] S. Takahashi and B. K. Ghosh, "Parameter estimation under perspective and orthographic projections using laser range finder," *Transactions of the Society of Instrument and Control Engineers (SICE)*, vol. 39, no. 2, 2003.
- [110] L. Zhang and B. K. Ghosh, "Three dimensional structure estimation and planning with vision and range," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 2515–2520, 2000.
- [111] Z. Zhang and B. K. Ghosh, "A multisensor fusion approach to shape estimation using a mobile platform with uncalibrated position," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pp. 205–210, 1999.
- [112] J.-X. Xu and Y. Tan, *Linear and Nonlinear Iterative Learning Control*. London: Springer, 2003.
- [113] T. Sugie and T. Ono, "An iterative learning control law for dynamical systems," *Automatica*, vol. 27, no. 4, pp. 729–732, 1991.
- [114] M. Ghosh, "Spatial based iterative learning control with vision sensor for 2D path tracking using a 2-axis gimbal robot," Master's thesis, Utah State University, Logan, April 2004.
- [115] N. S. Flann, K. L. Moore, and L. Ma, "A small mobile robot for security and inspection operations," in *IFAC Preprints of the First IFAC Conference on Telematics Applications in Automation and Robotics (TA2001)*, pp. 79–84, Weingarten, Germany, July 24–26 2001.
- [116] N. S. Flann, K. L. Moore, and L. Ma, "A small mobile robot for security and inspection operations," *Control Engineering Practice*, vol. 10, pp. 1265–1270, 2002.
- [117] K. Moore, N. Flann, R. S., M. Frandsen, Y. Chung, J. Martin, M. Davidson, R. Maxfield, and C. Wood, "Implementation of an omni-directional robotic inspection system (ODIS)," in *Proceedings of SPIE Conference on Robotic and Semi-Robotic Ground Vehicle Technology*. Orlando, FL: SPIE, May 2001.

- [118] H. G. Nguyen and J. P. Bott, "Robotics for law enforcement: beyond explosive ordnance disposal," SPAWAR Systems Center, San Diego, CA, Technical Report 1839, Nov. 2000.
- [119] E. Poulson, J. Jacob, R. Gunderson, and B. Abbot, "Design of a robotic vehicle with self-contained intelligent wheels," in *Proceedings of SPIE Conference on Robotic and Semi-Robotic Ground Vehicle Technology*, vol. 3366, pp. 68–73, Orlando, FL, USA, 1998.
- [120] K. L. Moore and N. S. Flann, "A six-wheeled omnidirectional autonomous mobile robot," *IEEE Control Systems*, vol. 20, no. 6, pp. 53–66, 12 2000.
- [121] M. Davidson and V. Bahl, "The scalar ϵ -controller: A spatial path tracking approach for ODV, Ackerman, and differentially-steered autonomous wheeled mobile robots," in *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 175–180. Seoul, Korea: IEEE, 2001.
- [122] Z. Song, Y. Chen, L. Ma, and Y. C. Chung, "Some sensing and perception techniques for an omnidirectional ground vehicle with a laser scanner," in *IEEE International Symposium on Intelligent Control*, 2002.
- [123] Z. Song, "Some localization strategies for mobile robots," Master's thesis, Utah State University, Logan, May 2003.
- [124] H. Shah, V. Bahl, J. Martin, N. S. Flann, and K. L. Moore, "Intelligent behavior generator for autonomous mobile robots using planning-based AI decision making and supervisory control logic," in *International Conference on Unmanned Robotic Vehicles*, Orlando, FL, April 2002.
- [125] R. R. Murphy, *Introduction to AI Robotics*. Cambridge, USA: MIT Press, November 2000.

- [126] J. Borenstein and Y. Koren, "Histogramic in-motion mapping for mobile robot obstacle avoidance," *IEEE Journal of Robotics and Automation*, vol. 7, no. 4, pp. 535–539, 1991.
- [127] J. Borenstein and Y. Koren, "The vector field histogramic fast obstacle avoidance for mobile robots," *IEEE Journal of Robotics and Automation*, vol. 7, no. 3, pp. 278–288, June 1991.
- [128] B. Holt, J. Borenstein, Y. Koren, and D. Wehe, "OmniNav: Obstacle avoidance for large non-circular, omnidirectional mobile robots," in *International Symposium on Robotics and Manufacturing*, Montpellier, France, May 1996.
- [129] I. Ulrich and J. Borenstein, "VFH⁺: Reliable obstacle avoidance for fast mobile robots," in *IEEE International Conference on Robotics and Automation*, pp. 1572–1577, Leuven, Belgium, May 16–21 1998.
- [130] H. Yang, J. Borenstein, and D. Wehe, "Double-VFH: Reliable obstacle avoidance for large, non-point, omni-directional mobile robots," in *IEEE International Conference on Robotics and Automation*, Pittsburgh, PA, April 1999.
- [131] I. Ulrich and J. Borenstein, "VFH*: Local obstacle avoidance with look-ahead verification," in *IEEE International Conference on Robotics and Automation*, pp. 2505–2511, San Francisco, CA, April 2000.
- [132] "Devantech SRF04 ultrasonic ranger specification," <http://www.acroname.com/robotics/parts/R93-SRF04.html>.
- [133] G. Dudek, P. Freedman, and I. M. Rekleitis, "Just-in-time sensing: efficiently combining sonar and laser range data for exploring unknown worlds," in *IEEE International Conference on Robotics and Automation*, pp. 667–671, April 1996.
- [134] B. Giesler, R. Graf, and R. Dillmann, "Fast mapping using the Log-Hough transformation," in *International Conference on Intelligent Robots and Systems*, pp. 1702–1707, 1998.

- [135] J. Guivant, E. Nebot, and S. Baiker, "High accuracy navigation using laser range sensors in outdoor applications," in *International Conference on Robotics and Automation*, April 2000.
- [136] B. Kluge, C. Kohler, and E. Prassler, "Fast and robust tracking of multiple moving objects," in *IEEE International Conference on Robotics and Automation*, May 2001.
- [137] C.-C. Wang and C. Thorpe, "Simultaneous localization and mapping with detection and tracking of moving objects," in *IEEE International Conference on Robotics and Automation*, May 2002.
- [138] "SICK LMS 221-30206 outdoor 2-D laser scanning system," <http://www.sick.de>.

Appendices

Appendix A

Non-Vision-Based Sensing and Perception

A.1 Template Matching

As part of the CSOIS effort on ODIS, a template matching approach was used for obstacle recognition from laser data. In this section, a portion of this research is described. This material is adapted and in some cases excerpted from [115, 116]. The original ideas are credited to the first author of [115, 116]. However, the implementation of these ideas was a contribution of this dissertation.

A.1.1 Robot Platform Introduction

The Utah State University Omnidirectional Inspection System is a small, man-portable mobile robotic system that can be used for autonomous or semi-autonomous inspection under vehicles in a parking area [35, 115, 117]. Customers for such a system include military police and other law enforcement and security entities [118]. The robot features 1) three “smart wheels” [119, 120] in which both the speed and direction of the wheel can be independently controlled through dedicated processors; 2) a vehicle electronic capability that includes multiple processors; and 3) a sensor array with a laser, sonar and IR sensors, and a video camera. A unique feature in ODIS is the notion of the “smart wheel” [119] developed by CSOIS, which has resulted in the so-called T-series of Omnidirectional Vehicles (ODV) [120]. With the ODV technique, the robots can achieve complete control of the vehicle’s orientation and motion in a plane, thus making the robots almost holonomic - hence “omnidirectional.”

ODIS employs a novel parameterized command language for intelligent behavior generation [35]. A key feature of the ODIS control system is the use of an object recognition system that fits models to sensor data. These models are then used as input parameters to the motion and behavior control commands [115]. Figure 1.2 shows the mechanical layout

of the ODIS robot. The robot is 9.8 cm tall and weighs approximately 20 kgs. The mechanical subsystem of the ODIS vehicle is based on three omnidirectional wheel assemblies mounted within a stressed-skin chassis, which also contains the vetronics subsystem, the battery and power distribution subsystem, sensors, and a camera gimbal subsystem. The ODIS vetronics architecture includes seven different processing nodes: the Master Node, three Sensor Nodes, the Camera Node, and three Wheel Nodes. A user interface is maintained through two independent wireless communications links, one connecting the master node to an external joystick and a second that connects the master node to the Planner Node, Planner GUI, and Vehicle GUI, all of which reside off-vehicle. There are two types of sensing functions on ODIS. For navigation ODIS uses a combination of GPS, a fiber optic gyro (FOG), and odometry based on wheel encoder measurements. For environmental sensing, ODIS uses three types of sensors. Ten ultrasonic sensors are used together with thirty-two infrared (IR) sensors and one laser distance measurement sensor. The system also has a camera node to control the camera pan/tilt mechanism that is used to point the camera relative to the vehicle. However, the camera is currently designed for the under car inspection video transmission to a base station computer not intended for decision-making or control. For more detailed description, see [35, 115, 117].

Figure A.1 shows the behavior control architecture that has been developed. Starting from the “inside out,” the control architecture contains two inner motion-control loops. The inner most loop is the wheel-level control, which acts to drive each smart wheel to its desired steering and drive speed set-points. The wheel-level controller uses simple PID control algorithms. Around the inner loop is the path-tracking controller. This loop derives the set points need by the wheel-level control in order to force the vehicle to follow a desired path in space, where a path is defined as an arc in inertial space (with a prescribed velocity along the arc) and an associated vehicle yaw motion. The path-tracking controller uses a newly-developed spatial tracking control algorithm that is described in more detail in [121]. Finally, the behavior controller, using object locations determined by the object manager, determines x , y , and yaw path command set points

to be given to the path-tracking controller. The behavior controller implements what is called a delayed commitment strategy, whereby scripts are instantiated with sensor data at run-time. A more detailed discussion of the delayed commitment approach can be found in [35], including discussion of the specifics of the parameterized command language and the signal flow through the system.

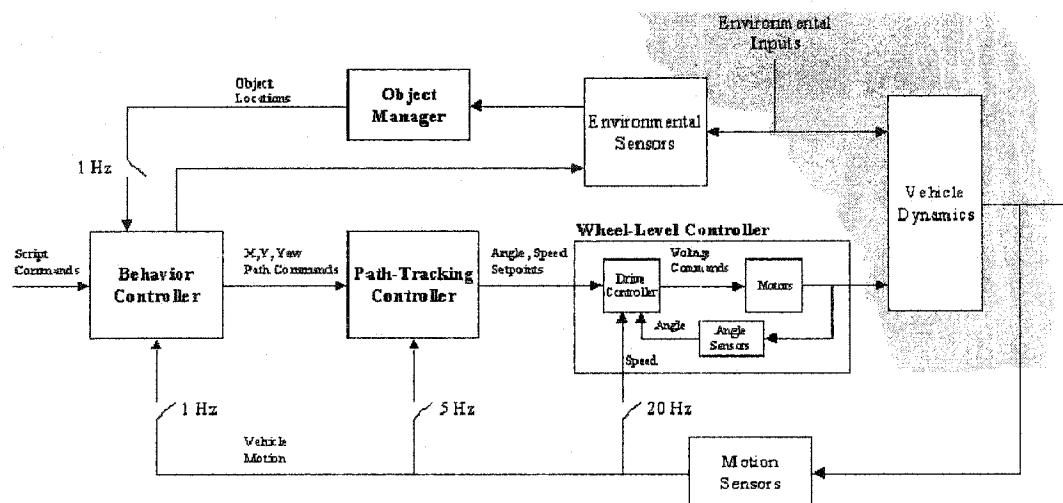


Fig. A.1: The behavior control system architecture of ODIS.

A.1.2 Object Manager

In the remainder of this section, we focus on the most critical component in the architecture: the object manager. The object manager block shown in fig. A.1 is the primary computational engine of the delayed commitment strategy [115, 116].

A.1.2.1 Problem Definition

The problem that the object manager solves is described as follows:

Given:

- 1) Sensor commands that instruct ODIS to scan known or unknown objects.
- 2) Sensor data that has been processed through the navigation system and transformed into global coordinates.

- 3) Parameterized object models that constrain the shape and relationship among objects and sub-objects.

Find: An instantiation of an object model (and individual object with associated sub-objects).

Such that: The error between the sensor data and the model is minimized.

A.1.2.2 Basic Approach

The approach that is used in the object manager is parameterized model fitting with constraint propagation. Thus, our overall strategy can also be described as one of model-based sensing. In our approach, knowledge of the anticipated objects is incorporated into the object creation process in two principal ways.

First, the command scripts include sensor commands that directs the sensors to scan for objects that are anticipated due to the structure of the task. For example, when approaching a known parking stall (from the original map), an acquired command will be included in the script that directs ODIS to look in the stall. This command will include a unique symbol, such as CAR01, which, if a car is detected, will be bound to the newly created object. This variable binding enables the newly detected object to be referenced later in the script for specifying paths relative to the object or for acquiring corresponding sub-objects, such as wheels.

Second, the object recognition process includes default models of the anticipated objects such as cars, tires and curbs. In this way, the anticipated shape, orientation and relative position of these objects can be used to constrain the search for an actual object when processing the actual sensor data. For example, by using a model of a car, once two sides have been detected using sonar data, the position of all four tires can be postulated and then actively searched for using the onboard laser.

A.1.2.3 Object Manager Overview

Figure A.2 gives an overview of the object manager. There are three sub-systems that directly connect with the object manager. The off-board planner sends known objects

to the object manager when the user loads a map into the GUI. This enables the object manager to account for sensor data originating from already known objects (such as curbs or posts in the parking lot). Once new objects have been acquired by the object manager, they are sent to the planner for use in formulating new plans and for display on the user's GUI.

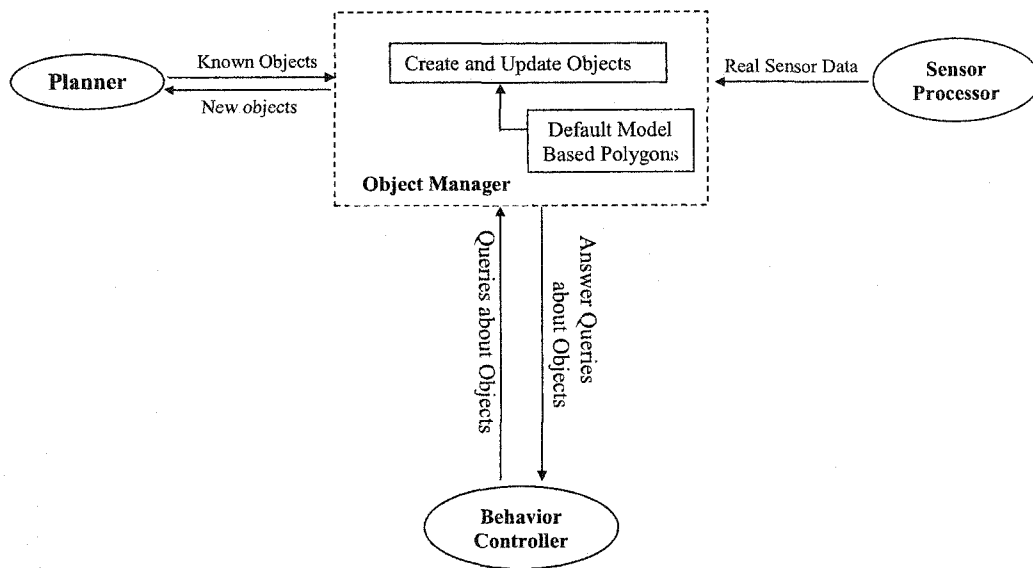


Fig. A.2: Object manager architecture.

The sensor processor is responsible for executing sensor commands (described earlier), collecting raw sensor data from the IR, sonar and laser, and filtering and transforming the relative data into globally referenced data suitable for processing by the object manager. The final interface is with the behavior controller, where queries about objects in the environment are serviced. It is this interface that enables commands such as “translate(3, line_bisect_face (CAR01))” to be converted into exact paths, by resolving (at run time) the “line_bisect_face (CAR01)” into a line in space, that can in turn be passed to the path controller.

A.1.2.4 Object Manager Algorithms

Currently all models are parameterized polygons, where the parameters determine the shape, scale, rotation and translation of the polygon. Associated with the parameters is a set of constraints that consist of linear equations and equalities among the values and bounds on the values. In this way, relationships among parameters of the object and among the object and its corresponding sub-objects can be described and utilized during fitting. An example of a linear constraint equality would be that the back tires of a car are parallel to the side face of a car. An example of a bound would be that the tires cannot exceed the car polygon and that the front tires can be rotated relative to the back tires by only a small angle. This paragraph is adapted and in some cases excerpted from [115, 116].

The process of fitting a polygon model is illustrated in figs. A.3 and A.4 using real laser data and simulated data, respectively. Notice how only points that are not accounted for by other objects are included in the set. First, a convex hull is computed around the points and possible corner points identified. Second, all possible lines are fit using the points that lie between each of the possible corners. Third, a dynamic programming approach is used to find the minimum error sequence of lines around the shape through a process of consolidation of lines (and elimination of corners). It is here that known angle constraints between faces of the polygon are applied to identify the best fitting corners of the polygon. This paragraph is adapted and in some cases excerpted from [115, 116].

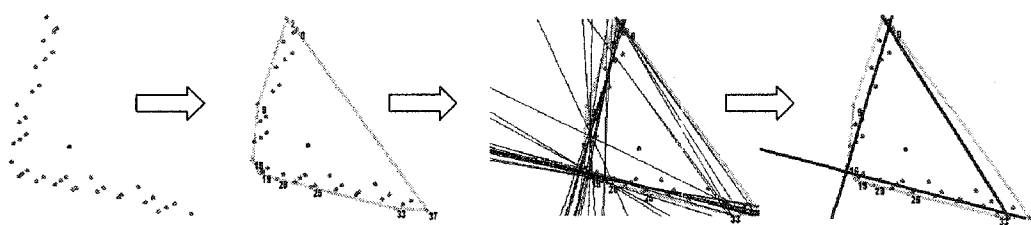


Fig. A.3: Rectangle model fitting algorithm using real laser data.

Once the polygon has been fit to the data, the parameters of this polygon are extracted (such as width, length and rotation) and propagated through the constraints to fix or

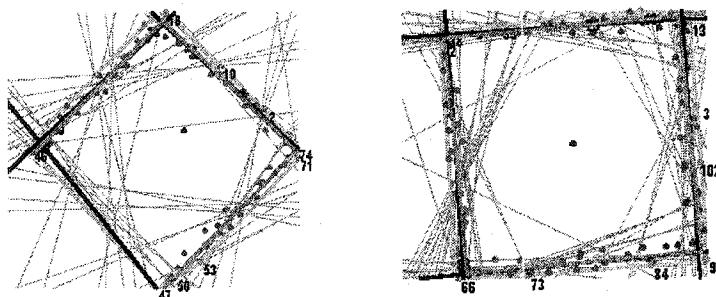


Fig. A.4: Rectangle model fitting algorithm using simulated points.

further constrain other parameters. It is in this way that the system can predict the location of a car's tires from its body, determine the orientation of the car, or determine the front from the back of a car. This paragraph is adapted and in some cases excerpted from [115, 116].

Finally, the fit model is used to create a new object (and associated sub-objects) that are placed in the object manager's database and associated with the named variable from the sensor commands (such as CAR01). In this way, future path commands can refer to the properties of CAR01 to determine paths. This paragraph is adapted and in some cases excerpted from [115, 116].

Besides the dynamic model-based fitting algorithm described in this section, we have, as our fitting tools, other circle or ellipse fitting algorithms, such as algebraic circle fit, circle fit with known radius, ellipse fit, plane fit, and 3-D corner fit algorithms that are documented in [122, 123].

A.1.2.5 Discussion and Conclusions

This section has described a small, man-portable mobile robotic system that can be used for autonomous or semi-autonomous inspection under vehicles in a parking area. The mechanical and vehicle electronic capabilities of the robot have been described, as has the functional approach to behavior generation. A detailed discussion of ODIS's object recognition system was given. This system fits models to sensor data. These models are then used as input parameters to the motion and behavior control commands.

A.2 Range Sensors for Obstacle Avoidance

Range sensors, such as sonar, IR, and laser, have been widely used for obstacle avoidance. In the multi-robot security system described in [124], the T2E robot (in fig. 1.4) is to be deployed to work together with one or more ODIS robots to provide a variety of security functions for parking areas, both autonomously and semi-autonomously. The ODIS is designed to perform under-vehicle inspection of cars, while the T2E serves as a “marsupial mothership” for general surveillance and protection of ODIS. As soon as a mission¹ starts its execution, various reactive agents in the system, namely, the safety and obstacle avoiding agent and the localizing agent, start acting. This section focuses on the safety and obstacle avoiding agent (for simplicity, we call it the safety agent hereafter), whose main function is to help the system analyze and estimate the dynamics of objects in the environment in a “better” way, so as to prevent the robot from running into dangers or being a danger itself to others. To accomplish this task satisfactorily, the robot T2E needs to be able to deal with a dynamic environment and must have capabilities such as motion detection and obstacle avoidance. For a simple start, we assume that no one, including people, objects, and cars, will hurt the robot by intention, which means no one will try to hurt or chase the robot deliberately and endlessly. This assumption does not imply that the environment is always safe. The environment is still dynamic and there are still risks that the robot needs to watch out for, e.g., an un-cautious driver or a child. However, the risks are reduced to a reasonable level that the robot can handle. More precisely, this assumption allows us to begin with a simple and reliable algorithm for a moderately dynamic environment rather than shooting for a highly complex algorithm directly. Thus, the initial task becomes to develop necessary algorithms based on the available sensors so that the robot will not run into objects and cause damage to both itself and the objects. The basic idea is to fuse all the sensor data from the sonars and the 2-D laser into a local map that always accompanies the robot, and make decisions based on this map. In this way, the task is decomposed into two subtasks: Map Building and Decision Making.

¹A mission is a set of high-level tasks issued by the user, to be executed by the system.

The inputs and output of the safety agent are shown in fig. A.5, where the inputs can include a world map (to match with sensor data for unknown objects) and the current executing path/mission (for decision making). Other inputs to the safety agent are: 1) sensor data: data from 26 sonars and the 2-D laser in Body-Fixed Coordinate System (BFCS); 2) feedback information: the robot's position and orientation (yaw) in the Inertial Coordinate System (ICS); and 3) the robot's velocity vector in the BFCS. In the stage of map building, the robot uses its position, orientation, and the on-board sensor data to build a continuously updated map. Using this map, the safety agent will invoke a reactive behavior, if it judges a need, based on the robot's velocity vector. The output of the safety agent is a throttle value corresponding to different decisions. This throttle is sent to the Supervisory Task Controller (STC) of the robot, described in [124], which is responsible for the successful realizations of a mission, to decide the robot's actual velocity by multiplying the throttle with the desired velocity.

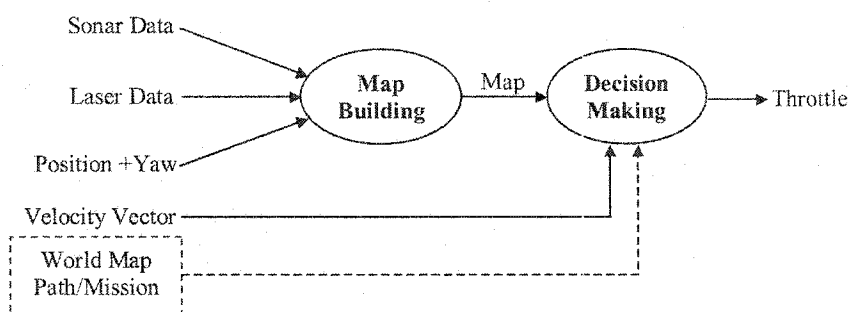


Fig. A.5: Flow chart of the safety agent.

There are two commonly used map building techniques: the Histogrammic In-Motion Mapping (HIMM) and the Bayesian method. Both of these are grid-based. The Bayesian method updates all the grids inside a sector for each sonar reading in a probabilistic way and the HIMM only updates grids on the sonar axis. According to the performance comparison of these two methods [125], the Bayesian method is more accurate and the HIMM has significant computational advantages. Though the HIMM algorithm itself is well-suited to model inaccurate and noisy range-sensor data by nature, existing works

only focus on sonar data without further investigation of the issues that might arise when trying to fuse sonar data with other kinds of sensor data. In [125], it is even said that the HIMM is limited to sonars. This section discusses some issues that are encountered in our application to build a HIMM map for collision avoidance with both the sonar and the laser data. The fundamental causes of these issues along with their possible solutions are also discussed.

A.2.1 HIMM Map Building

The map-building algorithm presented here is a modified HIMM that not only deals with sonar data, but also 2-D laser data, which is much more accurate and reliable. One question thus arises as to how to fuse these two kinds of sensor data together. They can either be built into one map or two independent maps. We choose to build these two kinds of sensor data into one map, based on the intuition that one map is computationally inexpensive. The map built by HIMM is not a global map. Using the same notation as in the HIMM papers [126, 127, 128, 129, 130, 131], it is called an active window and is always associated with the robot as the robot moves around.

The 26 sonars on T2E are the SRF04 sonars that offer ranging information from approximately 3 cm to 3 m with the beam pattern shown in fig. A.6 [132]. In order to fully protect the robot, 3 sonars are placed in the front, 3 on the back, 6 on either side, and 2 on the 4 corners. In this way, 100% sensing coverage of 1.8m and 80% coverage of 2.1m around the robot are achieved by only sonars, as shown in fig. A.7. The typical scanning time of sonars is approximately 100 ms for obtaining a full panorama.

Next, laser scanners have become more and more important for mobile robots in mapping [133, 134], navigation [135], and detection/tracking of moving objects [136, 137]. The 2-D scanning laser that has been selected is the SICK LMS 221-30206 outdoor scanning system, whose typical sensing range is 30 m with 10% reflectivity [138]. The laser has a maximum scanning angle of 180° and a response time of as low as 13 ms based on angle resolution of the scan. This laser has also been used successfully on previous. The sensing coverage of this 2-D laser is shown in fig. A.8, where a sensing range of 5 m is used only

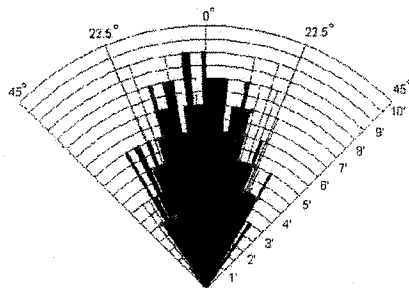


Fig. A.6: Beam pattern of the SRF04 sonars on T2E (in inch).

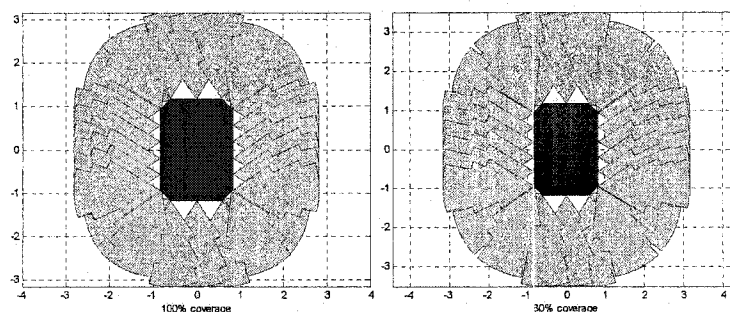


Fig. A.7: 100% and 80% sensing coverage around T2E (in meter).

for illustration. Because of the shape of T2E and the positions and sensing ranges of the equipped sensors, the size of the active window is chosen to be $10 \times 15 \text{ m}^2$ and each grid is $0.1 \times 0.1 \text{ m}^2$.

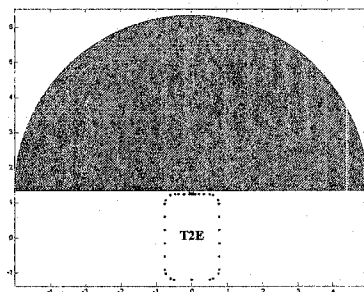


Fig. A.8: Typical laser coverage (radius is 5 m in this illustration).

Using HIMM [126, 127], each grid has a Certainty Value (CV) that can be incremented or decremented by sensor data. The CV is used to indicate how likely the grids are to be occupied by objects. The CV s are increased or decreased by sensor data until predefined

maximum or minimum values are reached. Using the same thresholds in the HIMM papers, CV_{max} is chosen to be 15. Similarly, CV_{min} is 0. The sonar data updating rule we use follows the ideas in [126, 127, 128, 129, 130, 131]. The laser data updating rule we use is our original contribution and is due to laser's accuracy. One important idea of the map building algorithm implemented in T2E is that we only rely on the sensor data to update the map. More specifically, we only rely on the sensor data to increase the CV s of the grids occupied by an obstacle and decrease the CV s when the obstacle is moving away.

HIMM Map Building with Sonars

In a typical HIMM updating rule, only one grid is incremented by a positive value (S^+) for each sonar range reading until a predefined maximum value is reached. The one that is incremented is the one that corresponds to the measured distance and lies on the sonar axis. All the grids that lie on the axis but are closer than the measured distance are incremented by a negative value (S^-) [126, 127].

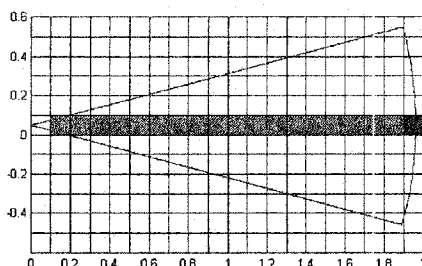


Fig. A.9: Sonar data updating rule.

HIMM Map Building with 2-D Laser

The updating rule for the 2-D laser data is similar to the case for sonar data. However, different values are used for increment (L^+) and decrement (L^-). The reason why the same updating rule can be applied to 2-D laser is because of laser data's high resolution (± 10 mm).

The 2-D laser is mounted at a height of approximately 1.1 m on T2E to detect the bumpers of cars in the parking lot. As the robot executes a mission, e.g. a "sweep" mission

to sweep the whole parking lot, the 2-D laser is always tilted up and down to be able to detect objects lower than the height of the 2-D laser. Thus, even though the HIMM map is a 2-D map, it is actually a representation of 3-D information. We only rely on the sensor data to update the map. Simply applying the above laser updating rule may cause trouble when there is a small object in the way. As the 2-D laser “looks” down, the laser “sees” the object and the CV s of corresponding grids get incremented; however, as the laser “looks” up, the laser fails to “see” the object and the CV s of the previously incremented grids get decremented. When this process goes on, the output decision of the safety agent will oscillate between STOP/SLOW and RESUME (refer to table A.2 for the decision logic). To overcome this problem, we deliberately introduce some delays when the decision needs to transmit from STOP \rightarrow RESUME, SLOW \rightarrow RESUME, and from STOP \rightarrow SLOW. In this way, the robot will only issue a RESUME when the obstacle moves away and no longer in the way.

Growth Rate Operator

The Obstacle Cluster Strength (OCS) is defined to be the sum of the squares of all CV s of an potential obstacle, though other definitions are feasible, but with different parameters and thresholds. In order to detect an object in time, the OCS should be built quickly. To avoid the scattering effect caused by in-motion sampling (which means when the robot is moving, sensor data is likely to scatter around the obstacle), a Growth Rate Operator (GRO) is applied to each range reading as it is projected onto the map. The GRO makes the increment of the certainty value of a certain grid faster when the neighbors of the grid have high CV s, which is done by convoluting CV_{ij} with a GRO mask. A 3×3 GRO mask is shown in table A.1. Adding the usual increment I^+ (L^+ or S^+), the new CV is $CV'_{ij} = CV_{ij} + I^+ + \sum_{p,q=-1}^{p,q=1} w_{p,q} CV_{i+p,j+q}$ [126, 127]. The disadvantage of GRO is that low-certainty areas adjacent to high-certainty areas build up to high CV s, resulting in a tendency to represent obstacles larger than they really are [126, 127]. Instead of using the fixed weight 0.5 for all adjacent neighbors, saturation can be applied to slow down the response of the system.

Table A.1: 3×3 GRO Mask

0.5	0.5	0.5
0.5	1.0	0.5
0.5	0.5	0.5

Projection from Previous Map

At any time instance, we have a previous map associated with the robot at the last position, containing all the processed sensing information so far within the active window. Along with this previous map, we have the most recent data set that is assumed to be collected at the current robot position. We first process the current data set within the current active window and then project the previous map into this current one. We call this method the map projection method. To do the projection, we simply calculate each grid center in the previous map and project it into the corresponding grid in the current active window if it is within this current active window. Doing it this way is different from keeping a rotation buffer of the past several sets of sensor data and using them to build the map around the robot at any time instance, which is computationally complex. This is because not only the locations detected by a sensor reading need to be processed, but all the grids on the axis connecting the detected locations and the sensor positions must be processed. The rotation buffer method will result in one grid being visited multiple times because of the intensity of laser data and the accuracy of both laser and sonar data, which are smaller than the grid size.

However, the map projection method has one significant disadvantage that will make the obstacles appear much larger even in the ideal case when there are no errors in the robot positions. This is shown in fig. A.10, where the robot is initially at 0 and the obstacle is at 3.2 (units are omitted in this illustration). Suppose the robot performs several consequent movements toward the obstacle with both its actual movements and measurements equal to 0.4 in all of the steps. The process of grids updating is illustrated in fig. A.10, where part 1) uses truncation operator and part 2) uses rounding-to-the-nearest. The grid marked with “S” indicates result from sensor data. Similarly, “P” indicates result from map projection.

The two graphs in fig. A.10 show that whether using truncation or rounding-to-the-nearest, the simple map projection method has a tendency to make the obstacles grow very quickly. To overcome this problem, we first build the map using the current data set and then do the projection from the previous map. When we do the projection, for each grid in the previous map that has a non-zero CV (CV_{pre}), after finding its projection in the current map, we search a neighborhood of this new location to see if there are grids whose CV is not equal to zero. If yes, we use CV_{pre} to update the certainty value of this neighbor; otherwise, we use CV_{pre} to update the certainty value of the grid after projection. In this way, the representation of obstacles in the resulting HIMM map depends more on the current data set than the projection and the projection works as a supplement to some degree.

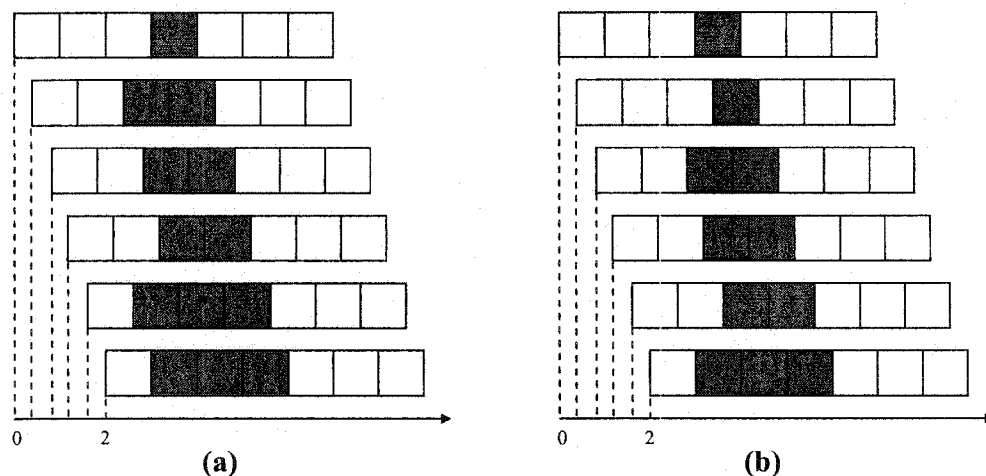


Fig. A.10: Illustration of errors introduced in the map projection.^{2 3}

²Obstacle position is 3.2, actual movement = measurement = 0.4

³(a) is by truncation and (b) is by rounding-to-the-nearest

A.2.2 Experimental Results

In order for the HIMM map to work properly, the robot is supposed to have “eyes” that can keep on watching its surroundings and each grid has a chance to be updated. This is the reason why the HIMM algorithm is well-known to work more accurately when

the robot is moving instead of standing still [126]. Actually this imposes no restrictions on the laser, because of the laser data's accuracy, high intensity, and the coarse grid size, all of which lead to the fact that the laser data updating rule can cover all the grids it "sees". But for sonars, if the robot is not moving, only grids on the sonar axis have the chance to be updated, which is an introduced "bias". In this case, because of the sonar data's sparse nature and the simple sonar data updating rule, if partial sonars happen to have erroneous readings, no other sonar readings can be used as complement or correction. As a result, there might be false detections, such as a detected obstacle does not actually exist, or an existing obstacle is not detected.

In the experimental setup shown in fig. A.11, there is a wall in front of the robot and many other static objects, such as tables and cylinder, are to the right of the robot about 0.8 m away. The left part of the robot is mainly an open area with the operator sitting 4.5 m away.

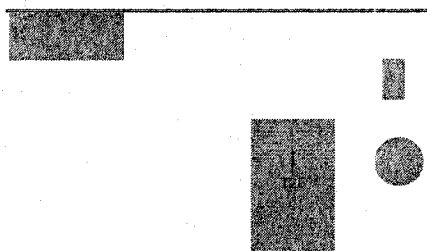


Fig. A.11: Experimental setup.

The HIMM maps built by the sonar and laser data are shown in fig. A.12 when the robot moves toward the wall (not exactly perpendicular to the wall) and is stopped by the safety agent around 1 m away from the wall. Since the wall totally cuts off the sensing range of all sensors, it is enough to plot only the $10 \times 10 \text{ m}^2$ map that is a partial representation of the original $10 \times 15 \text{ m}^2$ map. In fig. A.12, the dots forming a rectangle around the center indicate the sonar positions, which is also the robot boundary. The other dots are one set of laser readings and the circles are one set of sonar readings from the 26 sonars. In the implementation, sensor readings that are outside a reasonable and necessary sensing range are not considered (sonar: $[0.16, 4] \text{ m}$, laser: $[0.2, 50] \text{ m}$).

The HIMM maps being built are sufficient for collision avoidance. However, they are not very accurate. This is mainly due to the erroneous sonar readings, for which reason, the use of sonar is often confined to collision avoidance rather than exact mapping [133].

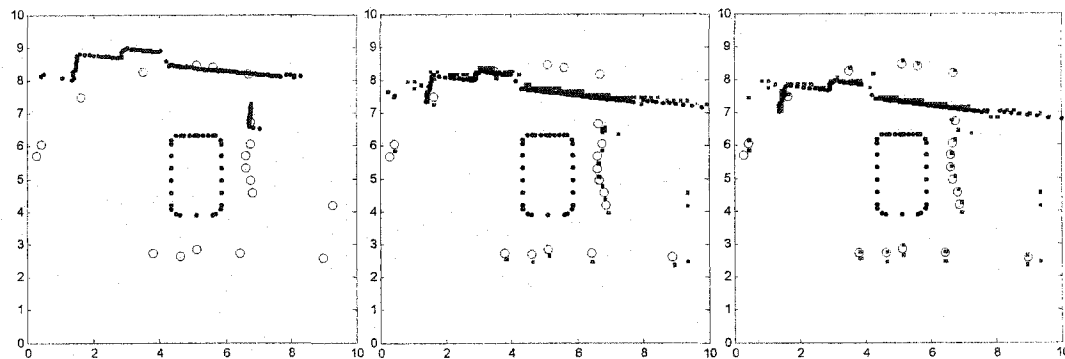


Fig. A.12: The $10 \times 10 \text{ m}^2$ HIMM maps built when T2E moves toward a wall in an indoor room.

Besides the GRO operator, another reason that objects will appear larger is the error in the robot position. Even though the HIMM map is built in the BFCS, the projection from the previous map still uses the robot position in the ICS. If the accumulative error in robot position becomes greater than half of the grid size, even for static object, its representation in the HIMM map by sensor readings will be different from its representation by projection. While this problem has no adverse effect when the robot goes toward the obstacles and needs to stop or slow down, it does have effect when the obstacles move away and the path is cleared, in which case the new sensor readings should be able to decrease the *CVs* of the grids originally occupied by the obstacles. As mentioned before, this requires all the grids have chances to be updated, which again emphasizes the assumption that the robot is moving instead of standing still. Furthermore, this imposes a more rigorous requirement for the decision making strategy discussed next, that is, the decision making strategy should be able to skip sparse isolated grids with low *CVs*.

A.2.3 Decision Making Strategy

The decision making strategy includes choosing decision regions, deciding whether or not there are obstacles inside the decision regions, and applying decision logic to decide if any action is needed.

Choose Decision Regions Based on the Velocity Vector

The decision regions are determined by the robot's desired velocity vector in the BFCS. Both its magnitude and direction are of importance. As shown in fig. A.13, given a velocity vector v , a rectangle box ABE_1E_2 is calculated as the stop region. Similarly, $E_2E_1F_1F_2$ is the slow region. These two rectangles are calculated in a same manner, but based on different scalars. E_1E_2 is perpendicular to the direction of v . The distance from the center of the robot to the center of E_1E_2 is proportional to the magnitude of v . The size AB is chosen such that the rectangle ABE_1E_2 will enclose the robot no matter what orientation the robot is currently holding. In this way, we can treat the robot as a point-size robot, eliminating the consideration for its shape and orientation. When the scalars are too large such that the points E_1E_2 or F_1F_2 are outside the active window, we will only consider the part that lies inside to check if the robot needs to stop or slow down.

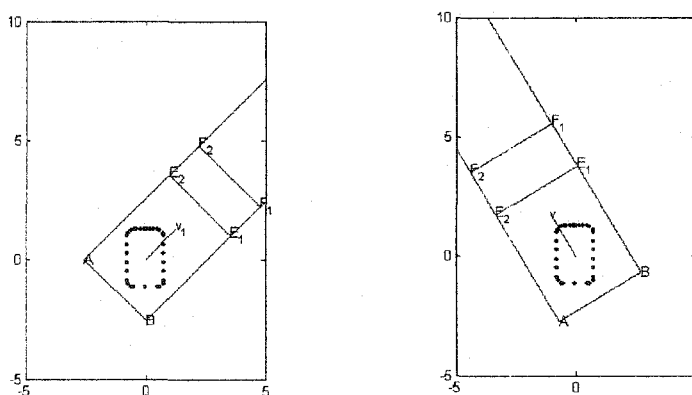


Fig. A.13: Decision making based on the velocity vector.

Is there obstacle inside a region?

The decision about whether or not there is obstacle inside one decision region is based on the following three factors: 1) number of potential obstacles inside the given region; 2) OCS of the largest potential obstacle; and 3) number of grids that are occupied by the largest potential obstacle. The potential obstacles are ordered by their OCS and the largest potential obstacle is the one with the highest OCS. To do this, we first loop through each grid that lies inside the given region and apply the 8-connectivity idea in image processing to group individual grids into potential obstacles, keeping track of the OCS and the number of grids occupied by each potential obstacle [39].

The final decision logic to tell if there is obstacle inside a given region is: output “yes” if the following three conditions are all true:

- 1) Number of potential obstacles \geq a predefined threshold `OBSTACLE_EXIST_NUMBER`.
- 2) OCS of the largest potential obstacle \geq
a predefined threshold `OBSTACLE_EXIST_THRESHOLD_VOTE`.
- 3) Number of grids occupied by the largest potential obstacle \geq
a predefined threshold `OBSTACLE_EXIST_THRESHOLD_AREA`.

The task to decide if there is obstacle inside a given region is not a trivial one. It is always a trade-off between fast response and reliability. To be fast, we can either decrease the thresholds or change the logic to be based on only one condition, rather than three; to be reliable, we would like to output “yes” when several conditions are met simultaneously. Due to the fact that there are unavoidable erroneous sensor readings, decisions that are made on relatively smaller thresholds will frequently output false alarms. The procedure to find a suitable logic along with the corresponding thresholds is a process of trial-and-error and we cannot claim that the logic we have used is an optimal one. However, during the testing, we do get the understanding that the *CV* always plays an important role in making the right decision.

The thresholds we choose are:

- 1) OBSTACLE.EXIST_NUMBER = 1.
- 2) OBSTACLE.EXIST_THRESHOLD_VOTE = 200.
- 3) OBSTACLE.EXIST_THRESHOLD_AREA = 1.

Taking the above thresholds, the decisions turn out to rely on the OCS of the largest potential obstacle only. This is reasonable because it is normal to have just one obstacle (at least in our parking lot application) and this obstacle just occupies one grid, e.g. a pole. However, the other two concerns are still helpful for further testing and other applications with finer grids.

Decision Logic

The decision logic and the throttle values are shown in table A.2. Currently, the T2E robot operates in a STOP-RESUME manner. If the robot judges a need to issue STOP or SLOW commands, it has to wait till the obstacles move away to issue a RESUME. The obstacles can be dynamic. But currently, the robot does not have the capability to response in a “smarter” way, which will require the robot to have such abilities as motion detection and obstacle avoidance. The state machine that illustrates all the transitions is shown in fig. A.14, where R_STOP denotes the region to check for STOP and R_SLOW for SLOW.

Table A.2: Decision Logic

Situation	Decision	Throttle
Nothing is in the way	NONE	
Something (Sth) is no longer in the way	RESUME	1.0
Sth is in the way and very close to robot	STOP	0.0
Sth is in the way and near the robot	SLOW	0.5

A.2.4 Parameters Thresholds via Testing

In the implementation, there are many parameters/thresholds that need to be tuned via exhaustive testing. The parameters/thresholds used in T2E are listed in table A.3 for

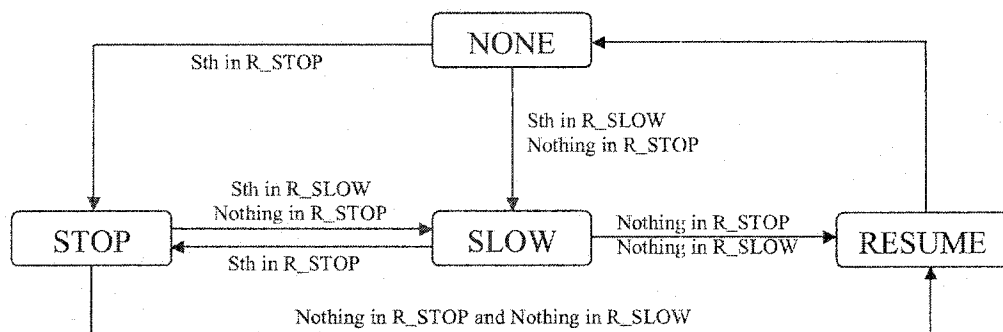


Fig. A.14: State machine illustrating transitions among all states.

the purpose of being helpful to others who are applying HIMM to their own applications. Special consideration should be given to the case when different sensors with totally different kinds of firing mechanisms are fused together. The tuning of the two important parameters I^+ and I^- does not fit in with the common sense that the more accurate the sensor is, the higher the I^+ . Besides the sensor's resolution, other factors include the firing rate and the intensity of the sensor data. When two kinds of sensors have the same coverage, the resulting HIMM map comes to depend heavily on the sensor with higher increment and firing rate.

Table A.3: Thresholds via Testing

Parameters/Thresholds	Value
STOP_SCALAR	2.0
SLOW_SCALAR	3.0
SONAR_PLUS S^+	5
SONAR_MINUS S^-	-2
LASER_PLUS L^+	4
LASER_MINUS L^-	-2
CV_{min}	0
CV_{max}	15
GRO_WINDOW_SIZE	2
OBSTACLE_EXIST_NUMBER	1
OBSTACLE_EXIST_THRESHOLD_VOTE	200
OBSTACLE_EXIST_THRESHOLD_AREA	1

A.2.5 Discussion and Conclusions

The HIMM algorithm can work as a good platform for fusion of different kinds of sensor data, as long as other readings from other sensors have sensor models. The application of the simple sonar updating rule to 2-D laser is direct due to the high resolution of the 2-D laser, but may cause several problems for compatible cooperation. In order for the HIMM algorithm to work accurately, proper tuning of the parameters (examples are listed in table A.3) are required. In our application of the HIMM algorithm, the T2E robot can operate reliably in a STOP-RESUME manner in a dynamic parking lot environment at a typical speed of 0.75 m/second after exhaustive tuning of these parameters and thresholds.

Sensing systems with vision alone, or with discontinuous firing of range sensors, can also perform the job, with the advantage of alleviating the use of many range sensors and mimicking human beings, who use vision systems to estimate relative depth.

Appendix B

Derivations

B.1 Derivations Supporting 3-D Motion Estimation in Section 4.2

Nonlinear Optimization Formulation When $\gamma \neq 0$

When $\gamma \neq 0$ and using the intrinsic matrix in (4.6), equation (4.4) becomes

$$\begin{aligned}
 \lambda' \begin{bmatrix} u'_i \\ v'_i \\ 1 \end{bmatrix} &= A \left(\lambda R A^{-1} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} + \mathbf{t} \right), \\
 &= A \left(\lambda R \begin{bmatrix} 1/\alpha & -\gamma/(\alpha\beta) & 0 \\ 0 & 1/\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} + \mathbf{t} \right), \\
 &= \lambda A \left(\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} u_i/\alpha - \gamma/(\alpha\beta)v_i \\ v_i/\beta \\ 1 \end{bmatrix} + \begin{bmatrix} t_1/\lambda \\ t_2/\lambda \\ t_3/\lambda \end{bmatrix} \right), \\
 &= \lambda \begin{bmatrix} \alpha & \gamma & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11}(u_i/\alpha - \gamma/(\alpha\beta)) + r_{12}v_i/\beta + r_{13} + t_1/\lambda \\ r_{21}(u_i/\alpha - \gamma/(\alpha\beta)) + r_{22}v_i/\beta + r_{23} + t_2/\lambda \\ r_{31}(u_i/\alpha - \gamma/(\alpha\beta)) + r_{32}v_i/\beta + r_{33} + t_3/\lambda \end{bmatrix}.
 \end{aligned}$$

Derivation of Equation (4.12)

From (4.11), $\frac{1}{Z_i^c} = [\Delta x_i - \frac{1}{f}\omega_1 x_i y_i + \omega_2(f + \frac{x_i^2}{f}) - \omega_3 y_i]/(ft_1 - x_i t_3)$. Then,

$$\begin{aligned}
 (ft_1 - x_i t_3)\Delta y_i &= (ft_2 - y_i t_3) \left[\Delta x_i - \frac{1}{f}\omega_1 x_i y_i + \omega_2(f + \frac{x_i^2}{f}) - \omega_3 y_i \right] \\
 &\quad + \left[-\frac{1}{f}\omega_2 x_i y_i + \omega_1(f + \frac{y_i^2}{f}) - \omega_3 x_i \right] (ft_1 - x_i t_3) \\
 &= t_2 f \Delta x_i - t_2 \omega_1 x_i y_i + t_2 \omega_2 (f^2 + x_i^2) - t_2 \omega_3 f y_i \\
 &\quad - t_3 \Delta x_i y_i + t_3 \omega_1 \frac{1}{f} x_i y_i^2 - t_3 \omega_2 (f + \frac{x_i^2}{f}) y_i + t_3 \omega_3 y_i^2 \\
 &\quad - t_1 \omega_2 x_i y_i + t_1 \omega_1 (f^2 + y_i^2) - t_1 \omega_3 f x_i
 \end{aligned}$$

$$+t_3\omega_2\frac{1}{f}x_i^2y_i - t_3\omega_1(f + \frac{y_i^2}{f})x_i + t_3\omega_3x_i^2.$$

That is,

$$\begin{aligned} 0 &= f\Delta y_i t_1 - f\Delta x_i t_2 + (\Delta x_i y_i - \Delta y_i x_i)t_3 \\ &\quad - (f^2 + y_i^2)t_1\omega_1 - (f^2 + x_i^2)t_2\omega_2 - (x_i^2 + y_i^2)t_3\omega_3 \\ &\quad + x_i y_i(t_1\omega_2 + t_2\omega_1) + f x_i(t_1\omega_3 + t_3\omega_1) + f y_i(t_2\omega_3 + t_3\omega_2), \end{aligned}$$

which can be written in the form of (4.12) for the i -th point.

B.2 Derivations for the PDS Theory in Section 4.3

Given the Riccati dynamics (4.1) with (4.42), (4.44), and (4.45), prove equations (4.43), (4.46), and (4.47).

Derivation of Equation (4.43)

Denote $\mathbf{X} = [X, Y, Z]^T$. From $\bar{p}X + \bar{q}Y + \bar{s}Z + 1 = 0$, $\mathbf{X}^T P = -1 \Rightarrow \mathbf{X}^T \dot{P} + \dot{\mathbf{X}}^T P = 0$.

Thus,

$$-\mathbf{X}^T \dot{P} = \dot{\mathbf{X}}^T P = (A\mathbf{X} + \mathbf{b} + F\bar{\mathbf{X}})^T P = \mathbf{X}^T A^T P + \mathbf{b}^T P + \bar{\mathbf{X}}^T F^T P. \quad (\text{B.1})$$

First, consider $\mathbf{b}^T P$:

$$\begin{aligned} \mathbf{b}^T P &= [b_1 \ b_2 \ b_3] [\bar{p} \ \bar{q} \ \bar{s}]^T = -(\bar{p}X + \bar{q}Y + \bar{s}Z)(b_1\bar{p} + b_2\bar{q} + b_3\bar{s}) \\ &= -[X \ Y \ Z] \begin{bmatrix} \bar{p} \\ \bar{q} \\ \bar{s} \end{bmatrix} (b_1\bar{p} + b_2\bar{q} + b_3\bar{s}) = -\mathbf{X}^T \begin{bmatrix} b_1\bar{p}^2 + b_2\bar{p}\bar{q} + b_3\bar{p}\bar{s} \\ b_1\bar{p}\bar{q} + b_2\bar{q}^2 + b_3\bar{q}\bar{s} \\ b_1\bar{p}\bar{s} + b_2\bar{q}\bar{s} + b_3\bar{s}^2 \end{bmatrix} \\ &= -\mathbf{X}^T \begin{bmatrix} b_1 & b_2 & b_3 & 0 & 0 & 0 \\ 0 & b_1 & 0 & b_2 & b_3 & 0 \\ 0 & 0 & b_1 & 0 & b_2 & b_3 \end{bmatrix} \begin{bmatrix} \bar{p}^2 \\ \bar{p}\bar{q} \\ \bar{p}\bar{s} \\ \bar{q}^2 \\ \bar{q}\bar{s} \\ \bar{s}^2 \end{bmatrix} = -\mathbf{X}^T B \bar{P}. \end{aligned} \quad (\text{B.2})$$

Then, consider $\bar{\mathbf{X}}^T F^T P$:

$$\begin{aligned} \bar{\mathbf{X}}^T F^T P &= \begin{bmatrix} X^2 \\ XY \\ XZ \\ Y^2 \\ YZ \\ Z^2 \end{bmatrix}^T \begin{bmatrix} f_1 & 0 & 0 \\ f_2 & f_1 & 0 \\ f_3 & 0 & f_1 \\ 0 & f_2 & 0 \\ 0 & f_3 & f_2 \\ 0 & 0 & f_3 \end{bmatrix} \begin{bmatrix} \bar{p} \\ \bar{q} \\ \bar{s} \end{bmatrix} = (f_1 X + f_2 Y + f_3 Z) [X, Y, Z] \begin{bmatrix} \bar{p} \\ \bar{q} \\ \bar{s} \end{bmatrix} \\ &= (f_1 X + f_2 Y + f_3 Z) (\bar{p} X + \bar{q} Y + \bar{s} Z) = \mathbf{X}^T \mathbf{f}. \end{aligned} \quad (\text{B.3})$$

From (B.1), (B.2), and (B.3), $\dot{P} = -\mathbf{f} - A^T P + B\bar{P}$.

Derivation of Equation (4.46)

Because $\bar{p}X + \bar{q}Y + \bar{s}Z + 1 = 0$, we have $[\bar{p}, \bar{q}, \bar{s}, 1] [X_1, Y_1, Z_1, W_1]^T = 0$. Denote $\mathbf{X}_1 = [X_1, Y_1, Z_1]^T$, we have:

$$P^T \dot{\mathbf{X}}_1 + \dot{W}_1 = -[\dot{\bar{p}}, \dot{\bar{q}}, \dot{\bar{s}}] \mathbf{X}_1 = (\mathbf{f}^T + P^T A - \bar{P}^T B^T) \mathbf{X}_1. \quad (\text{B.4})$$

In a similar manner as (B.3), we get $-\bar{P}^T B^T \mathbf{X}_1 = P^T \mathbf{b} W_1$. So, (B.4) becomes $P^T \dot{\mathbf{X}}_1 + \dot{W}_1 = P^T A \mathbf{X}_1 + P^T \mathbf{b} W_1 + \mathbf{f}^T \mathbf{X}_1$. That is $\dot{\mathcal{X}} = \mathcal{A}\mathcal{X}$.

Derivation of Equation (4.47)

Let $\mathbf{X} = [X, Y, Z]^T$. From $\bar{p}X + \bar{q}Y + \bar{s}Z + 1 = 0$, we have $[X, Y, Z, 1][p, q, s, w]^T = 1$.

Thus,

$$\mathbf{X}^T \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{s} \end{bmatrix} + \dot{w} = -(\mathbf{X}^T A^T + \mathbf{b}^T + \bar{\mathbf{X}}^T F^T) \begin{bmatrix} p \\ q \\ s \end{bmatrix}. \quad (\text{B.5})$$

In a similar manner as (B.3), we get $\bar{\mathbf{X}}^T F^T [p, q, s]^T = \mathbf{X}^T \mathbf{f} w$. So, (B.5) becomes

$$\mathbf{X}^T \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{s} \end{bmatrix} + \dot{w} = -\mathbf{X}^T A^T \begin{bmatrix} p \\ q \\ s \end{bmatrix} - \mathbf{X}^T \mathbf{f} w - \mathbf{b}^T \begin{bmatrix} p \\ q \\ s \end{bmatrix}. \quad (\text{B.6})$$

Thus, $\dot{\mathcal{P}} = -\mathcal{A}^T \mathcal{P}$.

Derivation of Optical Flow Dynamics

Given the Riccati dynamics (4.1) and (4.42), (4.45), prove that under the perspective projection $y_1 = X/Z$, $y_2 = Y/Z$, where the focal length f is assumed to be 1, the optical flow dynamics is of the form (4.64).

Proof: From $y_1 = fX/Z$, $\dot{y}_1 = f \frac{\dot{X}Z - \dot{Z}X}{Z^2}$. So

$$\begin{aligned}
 \frac{\dot{y}_1}{f} &= \frac{\dot{X}Z - \dot{Z}X}{Z^2} = \frac{\dot{X}}{Z} - \frac{X}{Z} \frac{\dot{Z}}{Z} \\
 &= \frac{a_{11}X + a_{12}Y + a_{13}Z + b_1 + f_1X^2 + f_2XY + f_3XZ}{Z} \\
 &\quad - y_1 \frac{a_{31}X + a_{32}Y + a_{33}Z + b_3 + f_1XZ + f_2YZ + f_3Z^2}{Z} \\
 &= a_{11}y_1 + a_{12}y_2 + a_{13} + b_1/Z - y_1(a_{31}y_1 + a_{32}y_2 + a_{33} + b_3/Z) \\
 &= a_{13} + (a_{11} - a_{33})y_1 + a_{12}y_2 - a_{31}y_1^2 - a_{32}y_1y_2 + \frac{1}{Z}(b_1 - b_3y_1). \quad (B.7)
 \end{aligned}$$

Since $pX + qY + sZ + w = 0$, we have $\frac{1}{Z} = -\frac{1}{w}(py_1 + qy_2 + s)$. So, (B.7) becomes

$$\begin{aligned}
 \frac{\dot{y}_1}{f} &= \underbrace{(a_{13} - b_1\bar{s})}_{d_1} + \underbrace{[a_{11} - a_{33} - b_1\bar{p} + b_3\bar{s}]}_{d_3} y_1 + \underbrace{(a_{12} - b_1\bar{q})}_{d_4} y_2 \\
 &\quad + \underbrace{(b_3\bar{p} - a_{31})}_{d_7} y_1^2 + \underbrace{(b_3\bar{q} - a_{32})}_{d_8} y_1y_2. \quad (B.8)
 \end{aligned}$$

In a similar manner, we can have (d_2, d_5, d_6) .

B.3 Supporting 3-D Imaging Surfaces in Section 4.7

Derivation of Equation (4.130)

$$\begin{aligned}
 \therefore Z &= \frac{1}{n_3}[(n_1X + n_2Y + n_3Z) - (n_1X + n_2Y)] \Rightarrow \frac{Z}{L_{pl}} = \frac{1}{n_3}(1 - n_1y_1 - n_2y_2), \\
 \therefore \dot{y}_1 &= \frac{d}{dt}(X \frac{1}{L_{pl}}) = \dot{X} \frac{1}{L_{pl}} + X \frac{d}{dt}(\frac{1}{L_{pl}}) \\
 &= (a_{11}X + a_{12}Y + a_{13}Z + b_1)/L_{pl} - y_1(\rho_1y_1 + \rho_2y_2 + \rho_3 + \rho_0y_3) \\
 &= a_{11}y_1 + a_{12}y_2 + a_{13} \frac{1}{n_3}(1 - n_1y_1 - n_2y_2) + b_1y_3 - y_1(\rho_1y_1 + \rho_2y_2 + \rho_3 + \rho_0y_3) \\
 &= \frac{a_{13}}{n_3} + (a_{11} - a_{13} \frac{n_1}{n_3} - \rho_3)y_1 + (a_{12} - a_{13} \frac{n_2}{n_3})y_2 - \rho_1y_1^2 - \rho_2y_1y_2 + (b_1 - \rho_0y_1)y_3,
 \end{aligned}$$

where $(\rho_0, \rho_1, \rho_2, \rho_3)$ are defined in (4.131).

Derivation of Equation (4.133)

The basic idea of deriving (4.133) is to replace the y_3 term in (4.130) by

$$y_3 = \frac{1}{n_1 X + n_2 Y + n_3 Z},$$

which is calculated in the following:

$$\begin{aligned} Z &= pX + qY + r \\ \Rightarrow \frac{1}{n_3} [n_1 X + n_2 Y + n_3 Z - (n_1 X + n_2 Y)] &= pX + qY + r \\ \Rightarrow \frac{1}{n_3} - \frac{n_1}{n_3} y_1 - \frac{n_2}{n_3} y_2 &= py_1 + qy_2 + r \frac{1}{n_1 X + n_2 Y + n_3 Z} \\ \Rightarrow \frac{1}{n_1 X + n_2 Y + n_3 Z} &= \frac{1}{r} \left[\frac{1}{n_3} - \tilde{p}y_1 - \tilde{q}y_2 \right], \end{aligned}$$

with $\tilde{p} = p + \frac{n_1}{n_3}$ and $\tilde{q} = q + \frac{n_2}{n_3}$ as defined in (4.134).

Derivation of Equation (4.141)

$$\begin{aligned} \dot{y}_4 &= -\frac{X\dot{X} + Y\dot{Y} + Z\dot{Z}}{(X^2 + Y^2 + Z^2)^{3/2}} \\ &= \frac{X(a_{11}X + a_{12}Y + a_{13}Z + b_1) + Y(a_{21}X + a_{22}Y + a_{23}Z + b_2)}{(X^2 + Y^2 + Z^2)^{3/2}} \\ &\quad + \frac{Z(a_{31}X + a_{32}Y + a_{33}Z + b_3)}{(X^2 + Y^2 + Z^2)^{3/2}} \\ &= -\frac{a_{11}X^2 + a_{22}Y^2 + a_{33}Z^2 + (a_{12} + a_{21})XY + (a_{13} + a_{31})XZ}{(X^2 + Y^2 + Z^2)^{3/2}} \\ &\quad + \frac{(a_{23} + a_{32})YZ + (b_1X + b_2Y + b_3Z)}{(X^2 + Y^2 + Z^2)^{3/2}} \\ &= -\left[\sum_{i=1}^3 a_{ii}y_i^2 + (a_{12} + a_{21})y_1y_2 + (a_{13} + a_{31})y_1y_3 + (a_{23} + a_{32})y_2y_3 \right] y_4 \\ &\quad + (b_1y_1 + b_2y_2 + b_3y_3)y_4^2 \\ &= -y_4 \left[\sum_{i=1}^3 a_{ii}y_i^2 + \frac{1}{2} \sum_{i,j=1, i \neq j}^3 (a_{ij} + a_{ji})y_iy_j \right] - y_4^2 \sum_{i=1}^3 b_iy_i. \\ \dot{y}_k &= \frac{a_{k1}X + a_{k2}Y + a_{k3}Z + b_k}{\sqrt{X^2 + Y^2 + Z^2}} + y_k \frac{\dot{y}_4}{y_4}, \quad \text{for } k = 1, 2, 3 \\ &= \sum_{i=1}^3 a_{ki}y_i - y_k \left[\sum_{i=1}^3 a_{ii}y_i^2 + \frac{1}{2} \sum_{i,j=1, i \neq j}^3 (a_{ij} + a_{ji})y_iy_j \right] + \left[b_k - y_k \sum_{i=1}^3 b_iy_i \right] y_4. \end{aligned}$$

Vita

Published Journal Articles

- Rational Radial Distortion Models of Camera Lenses with Analytical Solution for Distortion Correction, Lili Ma, YangQuan Chen, and Kevin L. Moore, *International Journal of Information Acquisition*, Accepted.
- A Small Mobile Robot for Security and Inspection Operations, N.S. Flann, K. L. Moore, and Lili Ma, *Control Engineering Practice*, vol. 10, pp. 1265-1270, 2002.

Published Conference Papers

- Range Identification for Perspective Dynamic Systems Using Linear Approximation, Lili Ma, YangQuan Chen, and Kevin L. Moore, *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2004.
- Range Identification for Perspective Dynamic System with Single Homogeneous Observation, Lili Ma, YangQuan Chen, and Kevin L. Moore, *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2004.
- Blind Detection and Compensation of Camera Lens Geometrical Distortions, Lili Ma and YangQuan Chen, *SIAM Imaging Science*, 2004.
- Flexible Camera Calibration Using a New Analytical Radial Undistortion Formula with Application to Mobile Robot Localization, Lili Ma, YangQuan Chen, and Kevin L. Moore, *Int. Symposium on Intelligent Control (ISIC)*, 2003.
- Sonar and Laser Based HIMM Map Building for Collision Avoidance for Mobile Robots, Lili Ma and Kevin L. Moore, *International Symposium on Intelligent Control (ISIC)*, 2003.

- Wireless Visual Servoing for ODIS - An Under Car Inspection Mobile Robot, Lili Ma, Matthew Berkemeier, YangQuan Chen, Morgan Davidson, Vikas Bahl, and Kevin L. Moore, *IFAC World Congress*, Spain, July, 2002.
- Visual Servoing of an Omni-Directional Mobile Robot for Alignment with Parking Lot Lines, Matthew Berkemeier, Morgan Davidson, Vikas Bahl, YangQuan Chen, and Lili Ma, *IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2002.
- Some Sensing and Perception Techniques for an Omnidirectional Ground Vehicle with a Laser Scanner, Zhen Song, YangQuan Chen, Lili Ma, and You Chung Chung, *IEEE Int. Symposium on Intelligent Control (ISIC)*, October 2002.
- A Small Mobile Robot For Security and Inspection Operations, Flann NS, Moore KL, and Ma L, *IFAC Conference on Telematics Applications in Automation and Robotics*, July 2001.