

**UNCALIBRATED ROBOTIC VISUAL SERVO
TRACKING FOR LARGE RESIDUAL PROBLEMS**

A Thesis
Presented to
The Academic Faculty

by

Jomkwun Munnae

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
The George W. Woodruff School of Mechanical Engineering

Georgia Institute of Technology
December 2010

Copyright © 2010 by Jomkwun Munnae

UNCALIBRATED ROBOTIC VISUAL SERVO TRACKING FOR LARGE RESIDUAL PROBLEMS

Approved by:

Professor Harvey Lipkin,
Committee Chair
The George W. Woodruff School of
Mechanical Engineering
Georgia Institute of Technology

Professor Nadar Sadegh
The George W. Woodruff School of
Mechanical Engineering
Georgia Institute of Technology

Professor Aldo A. Ferri
The George W. Woodruff School of
Mechanical Engineering
Georgia Institute of Technology

Professor Ayanna MacCalla Howard
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Wayne Daley
Georgia Tech Research Institute
Georgia Institute of Technology

Date Approved: 12 November 2010

To my parents,

Mingkwun and Sinothai Munnae

ACKNOWLEDGEMENTS

My foremost gratitude goes to my adviser, Prof. Harvey Lipkin, for his considerable effort in constructing me piece by piece to become a fine researcher. Through these years, he has shown me what it is, and how to be an extraordinary teacher. He is one of the most intellectual and knowledgeable persons I have ever met yet he is respectful to everyone, including his students. I truly appreciate him for never once making me feel uncomfortable in asking trivial questions and repeatedly explaining the answers until I absolutely understand. I am also profoundly grateful for those hours of editing and proofreading my thesis. Thank you for being hard on me and guiding me from the beginning until the end. Without your help this thesis would not have been possible. It is my privilege having you as my teacher and advisor.

I also owe my deepest gratitude to my FPTD division chief Gary McMurray, who never lost faith in me, even though in the toughest days I almost gave up myself. His trust and flexibility in relying on my inner strength and personality to complete the work make him an exceptional supervisor. I am grateful for his effort in finding me financial support till the completion of my study. Without his help, my work would have not been successful. Thank you for always being there. Your leadership, vision, and personality makes you the best boss an employee can ask for and it is my honor to have worked with you.

I would like to thank Prof. Nader Sadegh, Prof. Aldo A. Ferri, Prof. Ayanna MacCalla Howard, and Dr. Wayne Daley for serving as committee members, reviewing my dissertation, and providing valuable feedback.

I am also grateful to all my Georgia Tech academic professors, researchers, and staff for teaching and helping me to earn valuable knowledge in completion of this

work.

I truly appreciate the Georgia Tech Research Institute (GTRI) for providing me financial support and work opportunities through the Shackelford fellowship. My heartfelt thanks go to our director of GTRI's Aerospace, Transportation & Advanced Systems Laboratory (ATAS), Mr. Rusty Roberts, and all GTRI board members, especially Dr. Dennis Folds and Dr. Lora Weiss, for their generosity and encouragement. Being a part of GTRI is a once in a lifetime experience that allowed me to fully focus on my graduate study yet provided me the opportunity to work closely with industry.

I am also indebted to all of my colleagues at FPTD building in supporting me intellectually, physically, and mentally. Without their help my experience at Tech would not have been as memorable. Finally, I cannot thank enough all my family members, dearest friends, and all significant people in my life helping me get through good and bad times in the United States. Thank you for always believing in me. Each of you has significantly contributed in my success.

Jomkwun Munnae

TABLE OF CONTENTS

DEDICATION	iii
ACKNOWLEDGEMENTS	iv
LIST OF TABLES	x
LIST OF FIGURES	xv
LIST OF SYMBOLS OR ABBREVIATIONS	xxv
SUMMARYxxviii
I INTRODUCTION	1
1.1 Motivation	1
1.2 Literature Review	2
1.2.1 Visual Servoing Background	2
1.2.2 Literature Review of Uncalibrated Visual Servoing	4
1.2.3 Summary	7
1.3 Contribution	7
1.4 Organization	8
II NEWTON'S METHOD AND QUASI-NEWTON METHODS BACKGROUND	11
2.1 Newton's Method	11
2.1.1 Newton's Method for a One-Variable Scalar Function	11
2.1.2 Newton's Method for a Multiple-Variable Function	13
2.2 Quasi-Newton Methods	17
2.2.1 Broyden's method	17
2.2.2 Hessian Approximation	18
2.3 Damped Hessian	20
2.4 Line Search	22
2.5 Trust-Region Methods	27
2.6 Summary	30

III	DYNAMIC BROYDEN'S METHOD WITH RECURSIVE-LEAST-SQUARES (RLS) UPDATE	33
3.1	Overview	34
3.2	Previous Work in Nonlinear Visual Servoing Optimization	37
3.3	Dynamic Quasi-Newton Method via Recursive Least Squares Estimation	40
3.3.1	Theoretical Fundamental	40
3.3.2	Simulation and Experimental Results	43
3.4	Limitations	46
3.4.1	Large Initial Error	46
3.4.2	Selection of Optimal Forgetting Factor	47
3.4.3	Ill-conditioned Hessian Matrix	49
3.5	Summary	50
IV	MODIFIED METHODS FOR THE LARGE-RESIDUAL VISUAL SERVOING PROBLEM	52
4.1	Introduction	53
4.2	Approximation of the Hessian for Large-Residual Problems Background	54
4.2.1	Approximation of the Whole Hessian	55
4.2.2	Approximation of the Residual S	57
4.2.3	The Gill-Murray Method	60
4.3	Review of Large-Residual Visual Servoing Problems	61
4.3.1	Kim et al. Studies	62
4.3.2	Fu et al. Studies	63
4.4	Modified Hessian Approximations for the Large-Residual Visual Servoing Problem	68
4.4.1	Approximation of the whole Hessian H_k using the DBFGS update	69
4.4.2	Approximation of the residual S_k using the BFGS method	75
4.4.3	Approximation of the residual S_k using the MBFGS update	76
4.5	Convergence Analysis of the MBFGS Method	80

4.5.1	Convergence Analysis of the BFGS method	84
4.5.2	Convergence Analysis of the MBFGS method	90
4.5.3	Proofs of Lemmas	97
4.6	Rate of Convergence of the MBFGS method	112
4.7	Switching MBFGS-DB Algorithm	115
4.8	Summary	119
V	DYNAMIC ADAPTIVE FORGETTING FACTOR ALGORITHM . . .	123
5.1	Introduction	124
5.2	RLS Adaptive Filters Overview	125
5.3	Variable Forgetting Factor (VFF) Algorithms in Adaptive Filtering	129
5.3.1	RLS Algorithm with Adaptive Memory	130
5.3.2	Gradient-Based VFF RLS Algorithm (GVFF-RLS)	131
5.3.3	Gauss-Newton VFF RLS Algorithm (GN-VFF-RLS)	135
5.4	Dynamic Adaptive Forgetting Factor (DAFF) Algorithms for Visual Guide Control	139
5.4.1	Previous Work on a VFF Algorithm for Uncalibrated Visual Servoing	140
5.4.2	DAFF Method	141
5.5	Summary	148
VI	SIMULATION	149
6.1	Overview	150
6.2	System Description	152
6.2.1	Robot System	152
6.2.2	Camera System	152
6.2.3	Target Trajectories	154
6.2.4	Assumptions	155
6.3	Performance Evaluation of the Switching Algorithms for Large Residual Tracking	155
6.3.1	The RRR robot with a circular trajectory	158

6.3.2	The Effect of Partitioned and Non-Partitioned Broyden's Estimator for Approximating the Jacobian	161
6.3.3	The Effect of Switching Criterion	167
6.3.4	The PUMA 560 Robot	173
6.3.5	Conclusion	196
6.4	Performance Evaluation of the switching MBFGS-DB Algorithm with LMA	201
6.4.1	RRR Robot	203
6.4.2	PUMA 560 robot	208
6.4.3	Conclusion	219
6.5	Performance Evaluations of the DAFF Method versus the Existing VFF Algorithms	220
6.5.1	Effect of Parameter Values on Each VFF Algorithm	221
6.5.2	Performance Evaluation of Various VFF Algorithms	228
6.5.3	Comparison between the settling time t_s vs. the cycle time t_{cyc}	265
6.5.4	Conclusion	266
6.6	Performance Evaluation of the Switching MBFGS-DB with various VFF algorithm and with/without LMA	268
6.6.1	Conclusion	277
6.7	Summary	277
6.7.1	Conclusion	279
VII	CONCLUDING REMARKS	281
7.1	Contributions	282
7.2	Summary	284
7.3	Future Work	289
	REFERENCES	291
	VITA	298

LIST OF TABLES

5.1 The RLS and the DGN-PBM Parameter Equivalencies [29]	140
6.1 A summary of methods to improve the dynamic quasi-Gauss Newton algorithms	151
6.2 The DH parameters of a RRR robot.	152
6.3 Camera parameters.	154
6.4 A summary of methods for estimating residual \hat{S}_k : Dynamic BFGS, DFN-BFGS, MBFGS, and Fu's Method	157
6.5 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\lambda = 0.5$, and $v = 0.3$ for all algorithms.	159
6.6 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes using NP-Jacobian estimation for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\omega = 0.45$ rad/s, $\lambda = 0.5$, and $v = 0.3$ for all algorithms except the Fu-DB algorithm where $v = 0.5$ is used.	166
6.7 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes using NP-Jacobian estimation for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\omega = 0.9$ rad/s, $\lambda = 0.5$, and $v = 0.3$ for all algorithms.	166
6.8 A summary of the MBFGS algorithm varying in Jacobian approximation and strategies for \hat{S}_k inclusion using the RRR robot with two eye-to-hand cameras starting at $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$ with $\lambda = 0.5$	172
6.9 The RMS error and the settling time comparison between the switching, Scheme 1, and Scheme 2 MBFGS-DB algorithms with/without partitioned Broyden's method.	173
6.10 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the circular trajectory using a fixed forgetting factor $\lambda = 0.5$	180
6.11 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes with ALT Scheme 1 and 2.	185
6.12 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking four feature points of a circular trajectory moving at $\omega = 0.9$ rad/s using a fixed $\lambda = 0.5$ and v selected to ensure convergence.	186

6.13 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the cycloidal trajectory using a fixed forgetting factor $\lambda = 0.5$ and v selected to ensure convergence.	193
6.14 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the cycloidal trajectory using a fixed forgetting factor $\lambda = 0.5$ and v selected to ensure convergence.	195
6.15 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the helical trajectory using a fixed forgetting factor $\lambda = 0.5$, $v_x = 5$ mm/s, and v selected to ensure convergence.	196
6.16 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the helical trajectory using a fixed forgetting factor $\lambda = 0.5$, $v_x = 10$ mm/s, and v selected to ensure convergence.	201
6.17 The RMS error and the settling time comparison of the RRR robot for various residual \hat{S}_k approximation schemes with/without the LMA for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\lambda = 0.5$, and $v = 0.3$. The partitioned Broyden's estimator is used for Jacobian \hat{J}_k approximation.	204
6.18 The RMS error and the settling time comparison of the RRR robot for various residual \hat{S}_k approximation schemes with/without the LMA for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\lambda = 0.5$, and $v = 0.3$. The NP Broyden's estimator is used for Jacobian \hat{J}_k approximation.	205
6.19 Various starting RRR robot configurations.	205
6.20 The RMS error and the settling time comparison of the switching MBFGS-DB algorithm with/without the LMA using $v = 0.3$ and $\lambda = 0.5$ at various starting RRR robot configurations.	208
6.21 The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes using the PUMA 560 robot to track a circular trajectory using $\lambda = 0.5$ and $v = 0.3$	212
6.22 Various starting PUMA 560 robot configurations.	213
6.23 The RMS error and the settling time comparison for the switching MBFGS-DB algorithm with/without the LMA approach using $v = 0.3$ and $\lambda = 0.5$ at various starting PUMA 560 robot configurations tracking a circular trajectory.	213
6.24 The RMS error and the settling time comparison for the switching MBFGS-DB algorithm with/without the LMA using $v = 0.3$ and $\lambda = 0.5$ at various starting PUMA 560 robot configurations tracking a cycloidal trajectory.	219

6.25 RMS error and t_s and settling time t_s for λ values in the FFF strategy.	222
6.26 RMS error and t_s comparison of the VS-ARLS algorithm for different values of η_1 with $\lambda_k \in [0.60, 0.95]$.	223
6.27 RMS error and t_s comparison of the VS-ARLS algorithm with $\eta_1 = 0.05$ for various ranges of $[\lambda_{min}, \lambda_{max}]$.	224
6.28 RMS error and t_s comparison of the GN-VFF-RLS algorithm for various values of η_2 with $\lambda_k \in [0.85, 0.95]$.	224
6.29 RMS error and t_s comparison of the GN-VFF-RLS algorithm for various ranges of $[\lambda_{min}, \lambda_{max}]$ with $\eta_2 = 0.01$.	225
6.30 RMS error and t_s comparison of the DAFF method for various values of τ with $\lambda_k \in [0.2, 0.95]$.	225
6.31 The RMS error and t_s comparison of the DAFF method with various values of τ with $\lambda_k \in [0.20, 0.90]$.	226
6.32 RMS error and t_s comparison of the DAFF method for various values of τ with $\lambda_k \in [0.50, 0.90]$.	226
6.33 RMS error and t_s comparison of the DAFF method for various values of τ with $\lambda_k \in [0.50, 0.95]$.	226
6.34 RMS error and t_s for the Alt scheme between λ_{low} and λ_{high} .	227
6.35 Selected parameters values for each forgetting factor scheme.	228
6.36 The RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras tracking one feature point of a circular target moving at $\omega = 0.9$ rad/s and $v = 0.3$ for the no additional noise scenario.	229
6.37 The RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras tracking one feature point of a circular target moving at $\omega = 0.9$ rad/s. The switching criterion $v = 0.5$ is used when $\pm\frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.	233
6.38 RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras perpendicularly arranged. The target moves in a circular trajectory with an angular speed of $\omega = 0.9$ rad/s. $\pm\frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.	239

6.39 The RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras tracking one feature point of a circular target moving at $\omega = 0.9$ rad/s. The switching criterion $v = 0.5$ is used when ± 1 mm noise is added to the EE location in addition to $\pm \frac{1}{2}$ pixel uniform quantization noise added to the target and EE feature points.	244
6.40 The RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras tracking one feature point of a circular target moving at $\omega = 0.9$ rad/s. The switching criterion $v = 0.5$ is used when ± 1 mm noise is added to the EE location in addition to $\pm \frac{1}{2}$ pixel uniform quantization noise added to the target and EE feature points.	245
6.41 The RMS error and the settling time comparison for VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a circular target moving at $\omega = 0.45$ rad/s. The switching criterion is $v = 0.3$ and no additional noise is added.	249
6.42 The RMS error and the settling time comparison for VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a circular target moving at a faster angular speed $\omega = 0.9$ rad/s. The switching criterion is $v = 0.3$ and no additional noise is added.	249
6.43 The RMS error and the settling time comparison for VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a circular target moving at the angular speed $\omega = 0.45$ rad/s. The switching criterion is $v = 0.5$. Uniform quantization noise of ± 1 pixel is added to the target feature points.	250
6.44 The RMS error and the settling time comparison for VFF schemes using the PUMA 560 robot with two eye-in-hand cameras, each tracking four feature points of a circular target moving at the angular speed $\omega = 0.45$ rad/s. The switching criterion $v = 0.5$. Uniform quantization noise of ± 1 pixel is added to the target feature points.	257
6.45 The RMS error and the settling time comparison for various VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a square target moving at the speed 50 mm/s. The switching criterion $v = 0.3$. No additional noise is added.	263
6.46 The RMS error and the settling time comparison of various VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a square target moving at the speed 50 mm/s. The switching criterion $v = 0.3$ is used with ± 1 mm uniform quantization noise added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.	265

6.47 The RMS error and the settling time comparison of the DAFF and the Alt schemes using the PUMA 560 robot with two eye-in-hand cameras, each tracking four feature points of a square target moving at a speed 50 mm/s. The switching criterion is $v = 0.3$. Uniform quantization ± 1 mm noise is added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.	266
6.48 The cycle time (t_{cyc}) and the settling time (t_s) comparison of the switching MBFGS-DB with DAFF method for tracking various trajectories with varying target speed using v selected to ensure convergence.	267
6.49 The RMS error and the settling time comparison of the switching MBFGS-DB algorithm with/without LMA and a variety of the VFF algorithms. The PUMA 560 robot with an eye-in-hand camera is used for tracking four feature points of a cycloidal target. The switching criterion $v = 0.3$ is used with uniform quantization noise of ± 1 pixel added to the target feature points.	269
6.50 The RMS error and the settling time comparison of the DGN-PBM algorithm with/without LMA and a variety of the VFF algorithms. The PUMA 560 robot with an eye-in-hand camera is used for tracking four feature points of a cycloidal target. The switching criterion $v = 0.3$ is used with uniform quantization noise of ± 1 pixel added to the target feature points.	276

LIST OF FIGURES

2.1 A geometrical representation of an affine model.	12
2.2 Pseudo-code of Newton's method for solving unconstrained optimization problems	15
2.3 An example of an exact line search [1] starting from x_0 to reach the solution x^* along the steepest-descent direction at each iteration.	24
2.4 Pseudo-code for backtracking line search with Newton's method	25
2.5 Pseudo-code for the BFGS algorithm with the backtracking line-search method	26
2.6 An example of a trust region [76] with a radius δ_k centered at x_k where the next update x_{k+1} is restricted to be inside the trust region.	28
3.1 For an eye-to-hand system the camera is remote from the robot and the target.	35
3.2 For an eye-in-hand system the camera is attached to the robot EE.	36
3.3 A pseudo-code for the DBM-RLS	42
3.4 A pseudo-code for the DGN-PBM method	44
3.5 A simulation result of a six DOF robot with an eye-in-hand system using the DGN-PBM method when an initial error is significant shown in the task space view.	48
3.6 A simulation result of an eye-in-hand, six DOF robot using the DGN-PBM method with a constant λ shown in the task space view.	49
4.1 Pseudo-code for the algorithm in [25]	67
4.2 A pseudo-code for the DBFGS update	74
4.3 Pseudo-code for the DFN-BFGS approach	77
4.4 Pseudo-code for the MBFGS approach	81
4.5 Pseudo-code for the switching MBFGS-DB algorithm	120
5.1 Transversal filter [32].	127
5.2 A pseudo-code for the RLS algorithm [32].	129
5.3 A pseudo-code for the RLS algorithm with adaptive memory [32].	132
5.4 A pseudo-code for the GVFF-RLS algorithm [41].	136
5.5 A pseudo-code for the GN-VFF-RLS algorithm [71].	139

5.6	A pseudo-code for the VS-ARLS algorithm [29]	142
5.7	The expected $\ f_k\ $ values for ideal tracking performance and the hypothesized λ_k values in corresponding to $\ f_k\ $ with respect to time (s).	145
5.8	Pseudo-code for the DAFF method with the switching MBFGS-DB algorithm	147
6.1	(a)The RRR robot, (b) The PUMA 560 robot.	153
6.2	Pseudo-code for the switching DBFGS-DB, DFN-BFGS-DB, MBFGS-DB, and Fu-DB algorithms	160
6.3	The camera space of the RRR manipulator with two eye-to-hand camera configurations tracks a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ for all algorithms except the switching Fu-DB where $v = 0.5$ is used.	162
6.4	The task space view showing one camera and one target point for clarity (the others are similar) for the RRR manipulator with two eye-to-hand cameras tracking a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ for all algorithms except the switching Fu-DB where $v = 0.5$ is used.	163
6.5	The error norm of the RRR manipulator with two eye-to-hand camera configurations tracks a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ for all algorithms except the switching Fu-DB where $v = 0.5$ is used.	164
6.6	The performance comparison between the switching MBFGS-DB and NP-MBFGS-DB algorithms (a)-(b) the image plane, (c)-(d) the task space view, and (e)-(f) error norm of the RRR manipulator using two eye-to-hand cameras tracking a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$	165
6.7	Pseudo-code for the MBFGS-DB Scheme 1 using a similar scaling method presented in [17, 19]	169
6.8	Pseudo-code for the MBFGS-DB Scheme 2 using a hybrid method similar to [49]	171
6.9	The camera space of the RRR manipulator with two eye-to-hand cameras using (a) P-Switching ($v = 0.3$), (b) NP-Switching ($v = 0.3$), (c) P-Scheme 1, (d) NP-Scheme 1, (e) P-Scheme 2, and (f) NP-Scheme 2. A circular target trajectory is moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$	174

6.10 The task space view showing one camera and one target point using (a) P-Switching ($v = 0.3$), (b) NP-Switching ($v = 0.3$), (c) P-Scheme 1, (d) NP-Scheme 1, (e) P-Scheme 2, and (f) NP-Scheme 2. A circular target trajectory is moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$	175
6.11 The camera space of the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ (except the DBFGS-DB $v=0.55$).	177
6.12 The task space view showing camera and one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ (except the DBFGS-DB $v=0.55$).	178
6.13 The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ (except the DBFGS-DB $v=0.55$).	179
6.14 The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a circular trajectory using the DBFGS-DB with Scheme 1 and 2.	181
6.15 The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a circular trajectory using the DFN-BFGS-DB with Scheme 1 and 2.	182
6.16 The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a circular trajectory using the MBFGS-DB with Scheme 1 and 2.	183
6.17 The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a circular trajectory using the Fu-DB with Scheme 1 and 2.	184
6.18 The task space view showing camera and one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.90$ rad/s. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.	187
6.19 The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.90$ rad/s. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.	188

6.20 The camera space of the PUMA 560 manipulator with an eye-in-hand camera configuration tracks four feature points of a cycloid target trajectory. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.	190
6.21 The task space view showing camera and one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a cycloid target trajectory. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.	191
6.22 The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration tracks four feature points of a cycloid target trajectory. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.	192
6.23 The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a fast cycloidal trajectory.	194
6.24 The camera space of the PUMA 560 manipulator with an eye-in-hand camera configuration tracks four feature points of a helical target trajectory. The forgetting factor is $\lambda = 0.5$, $v_x = 5$ mm/s, and v is selected to ensure convergence.	197
6.25 The task space view showing camera and one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a helical target trajectory. The forgetting factor is $\lambda = 0.5$, $v_x = 5$ mm/s, and v is selected to ensure convergence.	198
6.26 The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration tracks four feature points of a helical target trajectory. The forgetting factor is $\lambda = 0.5$, $v_x = 5$ mm/s, and v is selected to ensure convergence.	199
6.27 The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a helical target trajectory moving with a faster speed in x direction. The forgetting factor is $\lambda = 0.5$, $v_x = 10$ mm/s, and v is selected to ensure convergence.	200
6.28 The camera space comparison of implementing the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting RRR robot configurations.	206

6.29 The camera space comparison of implementing the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting RRR robot configurations.	207
6.30 The camera space of the PUMA 560 manipulator with the eye-in-hand camera configuration using various switching algorithms implemented with the LMA to track four feature points of the circular target trajectory moving at $\omega = 0.45$ rad/s and $v = 0.3$. The forgetting factor is $\lambda = 0.5$	209
6.31 The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration using various switching algorithms with the LMA tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$	210
6.32 The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration using various switching algorithms implemented with the LMA to track four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$	211
6.33 The camera space comparison of the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s, $v = 0.3$, and $\lambda = 0.5$	214
6.34 The task space view showing one camera and one target point for clarity for the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s, $v = 0.3$, and $\lambda = 0.5$	215
6.35 The camera space comparison of the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a cycloidal target trajectory using $v = 0.3$ and $\lambda = 0.5$	216
6.36 The task space of the EE motion using the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a cycloidal target trajectory using $v = 0.3$ and $\lambda = 0.5$	217

6.37 The RMS tracking error comparison of the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a cycloidal target trajectory using $v = 0.3$ and $\lambda = 0.5$	218
6.38 The camera space of the RRR manipulator with two eye-to-hand cameras tracking one feature point of the circular target trajectory moving at $\omega = 0.90$ rad/s. Various VFF algorithms for λ_k are implemented with the switching MBFGS-DB with $v = 0.3$	230
6.39 The task space view showing one target point for clarity (the other views are similar) for the RRR manipulator with two eye-to-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s.	231
6.40 The error norm (top) and the forgetting factor λ_k (bottom) for the RRR manipulator with two eye-to-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The forgetting factor λ_k The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s.	232
6.41 The camera space of the RRR manipulator with two eye-to-hand cameras tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s. Various VFF algorithms for λ_k are implemented with the switching MBFGS-DB with $v = 0.5$ for which $\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.	234
6.42 The task space view showing one camera and one target point for clarity (the other view is similar) for the RRR manipulator with two eye-to-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s for which $\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.	235
6.43 The camera space of the RRR manipulator with two eye-to-hand cameras perpendicularly arranged tracking one feature point of the circular target trajectory moving at $\omega = 0.90$ rad/s. Various VFF algorithms for λ_k are implemented into the switching MBFGS-DB with $v = 0.5$ for which $\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.	237

6.44 The task space view showing one camera and one target point for clarity (the camera view is similar) for the RRR manipulator with two eye-to-hand cameras perpendicularly arranged using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s for which $\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.	238
6.45 A perpendicular camera arrangement where one camera is pointed in the z direction while the other camera is pointed into the $-x$ direction is used with the RRR robot for noise compensation.	240
6.46 The camera space of the RRR manipulator with two eye-to-hand cameras tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s. Various VFF algorithms for λ_k are implemented with the switching MBFGS-DB with $v = 0.5$ for which ± 1 mm noise is added to the EE location in addition to $\pm \frac{1}{2}$ pixel uniform quantization noise added to the target and EE feature points	241
6.47 The task space view showing one camera and one target point for clarity (the other view is similar) for the RRR manipulator with two eye-to-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s for which ± 1 mm noise is added to the EE location in addition to $\pm \frac{1}{2}$ pixel uniform quantization noise added to the target and EE feature points.	242
6.48 The task space view showing one camera and one target point for clarity (the camera views are similar) for the RRR manipulator with two eye-to-hand cameras perpendicularly arranged using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s for which ± 1 mm noise is added to the EE location in addition to $\pm \frac{1}{2}$ pixel uniform quantization noise added to the target and EE feature points.	243
6.49 The camera space of the PUMA 560 manipulator with the eye-in-hand camera tracking four feature points of the circular target trajectory moving at $\omega = 0.45$ rad/s. VFF algorithms for λ_k calculation are implemented into the switching MBFGS-DB with $v = 0.3$. No additional noise is added.	246

6.50 The task space view showing one camera and one target point for clarity (the other views are similar) of the PUMA 560 manipulator with the eye-in-hand camera tracking four feature points of the circular target trajectory moving at $\omega = 0.45$ rad/s. VFF algorithms for λ_k calculation are implemented into the switching MBFGS-DB with $v = 0.3$. No additional noise is added.	247
6.51 The error norm of the PUMA 560 manipulator with an eye-in-hand camera tracking four feature points of the circular target trajectory moving at $\omega = 0.45$ rad/s. VFF algorithms for λ_k calculation are implemented into the switching MBFGS-DB with $v = 0.3$. No additional noise is added.	248
6.52 The camera space of the PUMA 560 manipulator with an eye-in-hand camera tracking four feature points of the circular target trajectory moving at $\omega = 0.45$ rad/s. Various VFF algorithms for λ_k calculation are implemented into the switching MBFGS-DB with $v = 0.5$. Uniform quantization noise of ± 1 pixel is added to the target feature points. . .	251
6.53 The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k calculation and $v = 0.5$. The robot is tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. Uniform quantization noise of ± 1 pixel is added to the target feature points. . .	252
6.54 The error norm (top) and the forgetting factor λ_k (bottom) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. Uniform quantization noise of ± 1 pixel is added to the target feature points.	253
6.55 The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with two eye-in-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. Each camera is tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. Uniform quantization noise of ± 1 pixel is added to the target feature points.	255
6.56 The task space in YZ view showing one target point comparison between the one and two eye-in-hand cameras used for the PUMA 560 manipulator using the switching MBFGS-DB with the DAFF and Alt algorithms ($v = 0.5$) with uniform quantization noise of ± 1 pixel added to the target feature points. Each camera is tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s.	256

6.57 The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking four feature points of a square target trajectory moving at a speed 50 mm/s. No additional noise is added.	259
6.58 The error norm (top) and the forgetting factor λ_k (bottom) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking four feature points of a square target trajectory moving at a speed 50 mm/s. No noise is added.	260
6.59 The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking four feature points of a square target trajectory moving at a speed 50 mm/s. ± 1 mm uniform quantization noise is added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.	261
6.60 The error norm (top) and the forgetting factor λ_k (bottom) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking four feature points of a square target trajectory moving at a speed 50 mm/s. ± 1 mm uniform quantization noise is added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.	262
6.61 The task space view showing one camera and one target point (left column), the error norm and λ_k (right column) of the PUMA 560 manipulator with two eye-in-hand cameras using the switching MBFGS-DB with the DAFF and the Alt algorithms with $v = 0.3$. Each camera tracks four feature points of a square target trajectory moving at a speed 50 mm/s. Uniform quantization ± 1 mm noise is added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.	264
6.62 The camera space of the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.	270

6.63 The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.	271
6.64 The error norm and λ_k plots of the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB for various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal target trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.	272
6.65 The camera space of the PUMA 560 manipulator with the eye-in-hand camera using the DGN-PBM for various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.	273
6.66 The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the DGN-PBM for various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.	274
6.67 The error norm and λ_k plots of the PUMA 560 manipulator with an eye-in-hand camera using the DGN-PBM for various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal target trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.	275

LIST OF SYMBOLS OR ABBREVIATIONS

BD	Broyden-Dennis method.
BFGS	BroydenFletcherGoldfarbShanno method.
BFGS-QN	Quasi-Newton method with the BFGS method.
DBFGS	Dynamic BFGS algorithm.
DBFGS-DB	Switching modified DBFGS-dynamic Broyden algorithm.
DBM-RLS	Dynamic Broyden's method using recursive least-squares estimation.
DFN-BFGS	Dynamic Full Newton method with BFGS algorithm.
DFN-BFGS-DB	Switching modified DFN-BFGS-dynamic Broyden algorithm.
DGN-PBM	Dynamic Gauss-Newton algorithm with partitioned Broyden's method.
ECL	Endpoint closed-loop.
EOL	Endpoint open-loop.
FFF	Fixed forgetting factor algorithm.
Fu-DB	Switching modified Fu-dynamic Broyden algorithm.
GN-VFF-RLS	Gauss-Newton variable forgetting factor RLS.
GVFF-RLS	Gradient-based VFF RLS algorithm.
IBVS	Image-based visual servo system.
LMA	Levenberg-Marquardt algorithm.
MBFGS	Modified BFGS.
MBFGS-DB	Switching modified BFGS-dynamic Broyden algorithm.
MBFGS-QN	Quasi-Newton method with the MBFGS method.
NP	The non-partitioned Broyden's method.
P	The partitioned Broyden's method.
PBVS	Position-based visual servo system.
RLS	Recursive least-squares algorithm.
VFF	Variable forgetting factor algorithm.

VS-ARLS	Uncalibrated visual servoing using adaptive RLS algorithm.
\hat{H}_k	An approximation of H_k in which only the residual S_k is estimated.
$\hat{H}_{d,k}$	The modified Hessian matrix at iteration k.
α	The Levenberg-Marquardt parameter.
\check{f}_k	$\check{f}_k = \frac{\ f_k\ }{f_{max}}.$
Δf	$\Delta f = f_k - f_{k-1}$, the change in the image error.
δ	A trust region size.
η_1	Learning rate of the VS-ARLS algorithm.
η_2	Learning rate of the GN-VFF-RLS algorithm.
\hat{H}	An estimated Hessian.
\hat{H}_k	An estimated Hessian at iteration k.
\hat{J}_k	An estimated Jacobian at iteration k.
\hat{S}_k	An estimated Residual at iteration k.
λ	Forgetting factor.
Λ_k	$\Lambda_k = 1 - \lambda_k.$
$(\cdot)_k$	Denotes k th iteration.
$\ \cdot\ $	l_2 norm of a vector.
$\ \cdot\ _F$	Frobenius norm of a matrix.
ω	An angular speed.
τ	Time Constant.
θ	Robot joint angles.
\tilde{J}_k	$\tilde{J}_k = \begin{bmatrix} \hat{J}_{k-1} & \left(\hat{f}_t\right)_{k-1} \end{bmatrix}.$
\tilde{h}	$\tilde{h} = \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix}.$
v	Switching criterion.
φ	Switching parameter.
D	A diagonal matrix.

$E[\cdot]$	Expected value.
f_k	Error in the image space at iteration k.
f_{max}	$f_{max} = \max \{\ f_1\ , \ f_2\ , \dots, \ f_k\ \}.$
g_k^*	$g_k^* = \left[J_{k+1}^T (f_{k+1} + \frac{\partial f_{k+1}}{\partial t} h_t) \right] - \left[J_k^T (f_k + \frac{\partial f_k}{\partial t} h_t) \right].$
g_k	$g_k = J_{k+1}^T f_{k+1} - J_k^T f_k.$
H	The Hessian matrix.
H_k	The Hessian matrix at iteration k.
h_t	$h_t = t_k - t_{k-1}$, an increment of time.
h_θ	$h_\theta = \theta_k - \theta_{k-1}$, an increment of θ .
J	$\frac{\partial y}{\partial \theta}$, the composite Jacobian.
J_k	The Jacobian at iteration k.
m_k	The affine model of f .
P_k	Estimate of the inverse of the correlation matrix of h_θ .
q_k	The quadratic model of the objective function F .
t_s	Settling time.
t_{cyc}	Cycle time.
y	Robot EE feature points.
y^*	Target feature points.
z_k^*	$z_k^* = J_{k+1}^T f_{k+1} - J_k^T f_{k+1}.$

SUMMARY

In visually guided control of a robot, a large residual problem occurs when the robot configuration θ is not in the neighborhood of the target acquisition configuration θ^* . Most existing uncalibrated visual servoing algorithms use quasi-Gauss-Newton methods which are effective for small residual problems. The solution used in this study switches between a full quasi-Newton method for large residual case and the quasi-Gauss-Newton methods for the small case. Visual servoing to handle large residual problems for tracking a moving target has not previously appeared in the literature.

For large residual problems various Hessian approximations are introduced including an approximation of the entire Hessian matrix, the *dynamic BFGS (DBFGS)* algorithm, and two distinct approximations of the residual term, the *modified BFGS (MBFGS)* algorithm and the *dynamic full Newton method with BFGS (DFN-BFGS)* algorithm. Due to the fact that the quasi-Gauss-Newton method has the advantage of fast convergence, the quasi-Gauss-Newton step is used as the iteration is sufficiently near the desired solution. A switching algorithm combines a full quasi-Newton method and a quasi-Gauss-Newton method. Switching occurs if the image error norm is less than the switching criterion, which is heuristically selected.

An adaptive forgetting factor called the *dynamic adaptive forgetting factor (DAFF)* is presented. The DAFF method is a heuristic scheme to determine the forgetting factor value based on the image error norm. Compared to other existing adaptive forgetting factor schemes, the DAFF method yields the best performance for both convergence time and the RMS error.

Simulation results verify validity of the proposed switching algorithms with the

DAFF method for large residual problems. The switching MBFGS algorithm with the DAFF method significantly improves tracking performance in the presence of noise. This work is the first successfully developed model independent, vision-guided control for large residual with capability to stably track a moving target with a robot.

CHAPTER I

INTRODUCTION

1.1 *Motivation*

Visual sensing is critical for many biological systems but its use in robotic applications is still relatively limited. One of the reasons is that the vision systems typically require calibration and frequently this calibration drifts due to operating and environmental factors. Uncalibrated visual servoing, on the contrary, has advantages over the model-based visual control by eliminating requirement of system modeling and camera calibration.

A number of studies formulate uncalibrated visual servoing as nonlinear optimization problems in which Newton's method and quasi-Newton methods are typically used for finding a solution. These methods are powerful techniques that give quadratic convergence if certain assumptions apply. As a result, various algorithms incorporate them to give effective tracking performance in uncalibrated systems.

One of the major challenge of the quasi-Gauss-Newton algorithms is that they are limited to only the zero- or small-residual cases. The residual S_k is a second order differential term appears in the Hessian H_k matrix. For a visual control problem, a zero- or small-residual problem refers to the case that the initial robot configuration θ_0 is close to the target acquisition configuration θ^* so S_k is likely small and can be ignored. Consequently, the quasi-Newton method becomes the quasi-Gauss-Newton method when the residual S is excluded. Often the residual S is computationally expensive and is set to zero.

However, there exists circumstances where the initial robot configuration θ_0 is not near the desired target configuration θ^* and the residual S_k becomes significant.

This is called the *large-residual* problem. In this case quasi-Gauss-Newton method is less appropriate and the full quasi-Newton method (including S) should be applied. Visually guided control to handle large residual problems for *moving* target tracking has not appeared in the literature.

1.2 Literature Review

The objective of this study is to develop an uncalibrated visual servoing system for the large residual problem that accurately tracks a moving target with fast convergence. Since uncalibrated visual servoing is a sub-set of visual based control, an overview of visual based control is discussed in Section 1.2.1. Then various studies of uncalibrated visual servoing related to large residual problems and adaptive forgetting factors are presented in Section 1.2.2. The chapter is concluded in Section 1.2.3.

1.2.1 Visual Servoing Background

A sensory system that substantially improves autonomous system capabilities as well as increases the versatility of robotic applications is visual-sensing. Using visual information to control a robot end-effector position in relation to a target is referred to as visual servoing. The algorithm has significantly played an important role in a wide range of robotic applications such as grasping, teleoperation, missile tracking, or aircraft landing.

In 1979 Hill and Park [33] introduced the first taxonomy of visual servoing. In 1980, Weiss and Sanderson [79] organized visual servo systems into four classifications: dynamic look-and-move, direct visual servo, positioned-based, and imaged-based visual servo systems. Later Hutchinson et al. [36] suggest two categories, position-based and image-based visual servo control. They also distinguish between two degrees of system observability: *endpoint open-loop* (EOL) and *endpoint closed-loop* (ECL) systems. An EOL system only observes the target while an ECL system observes both the target and robot end-effector.

Position-based visual servoing (PBVS) uses visual information in conjunction with a knowledge of the robot kinematic model, the geometric target model, and the camera model to minimize error between the current and the desired robot position in the task space. The desired robot position is obtained by extracting, interpreting, and transforming image features to approximate target position in relation with the camera pose. The system is remarkably sensitive to a precise knowledge of kinematic robot model and accurate camera calibration. Examples of PBVS developments are presented in [14, 80]. Because it requires robot and camera models, this scheme does not serve the purpose of the research objective.

When the error between the robot end-effector and target position is directly described in terms of image feature parameters, this type of visual servo control is known as image-based visual servo system (IBVS). This scheme servos a robot end-effector such that the image error is minimized without requiring a spatial robot pose estimation and is more robust to robot or camera calibration errors. When a robot kinematic model is available, errors mostly occur in the computation of the image Jacobian or interaction matrix - a map between the robot joint onto the camera space space. This scenario requires camera calibration to determine intrinsic and extrinsic camera parameters. Intrinsic parameters include focal length, radial distortion components, pixel sampling components, etc. Extrinsic parameters describe the position and orientation of the camera with respect to a global coordinate frame or an arbitrary coordinate frame depending on the system configuration. Two common camera configurations used in visual servo control are a fixed-camera system known as an *eye-to-hand* camera system, and an *eye-in-hand* system where a camera is attached to the robot end-effector.

1.2.2 Literature Review of Uncalibrated Visual Servoing

The objective of this research is to develop a model-free visual-guided control that effectively tracks a moving target for large residual problems. Uncalibrated visual servoing control does not require a priori knowledge of camera and robot models.

Hosoda and Asada [34, 35] and Jagersand et al. [37] demonstrate the Jacobian estimation using the Broyden rank-one method to servo a robot end-effector for static target tracking. Piepmeyer et al. [57] demonstrate a dynamic quasi-Newton approach in which the dynamic Broyden's method with an exponentially weighted (by forgetting factor λ) recursive least square (RLS) scheme is introduced. It estimates the composite Jacobian which combines the camera image Jacobian and the robot kinematic Jacobian to track a moving target tracking with a stationary camera. In [59] the algorithm is extended to an eye-in-hand system where a camera is attached to an end-effector for a moving target tracking. Simulation and experimental results verify stable and convergent tracking for moving targets with both stationary and eye-in-hand cameras. The details of these algorithms are reviewed in Chapter 3.

Piepmeyer et al. [57, 59] use quasi-Gauss-Newton based algorithms, which assume zero- or small-residuals so the residual S_k is neglected in the Hessian approximation. Consequently, for large residual problems the algorithms either slowly converge or diverge. To overcome this problem, Fu et al. [25] use a secant method to approximate the residual \hat{S}_k . The estimation of residual \hat{S}_k is done by an algorithm proposed by Dennis et al. [19, 17] for solving nonlinear least squares problems. This algorithm uses a trust region method for guaranteeing global convergence, and the dynamic Jacobian estimation method presented in [57]. Even though simulation results show improved convergence with desirable accuracy, this algorithm is only applied to a stationary target.

Kim et al. [40, 38, 39] also propose an uncalibrated visual servoing for large residual problems. Similar to [25], the full Newton's method and the secant approximation

are used to calculate robot joint angles for static target tracking. However, the mathematical formula for updating the residual \hat{S}_k is different. Although simulation results show improvement of the system control, the trajectories of the robot end-effector exhibit significant oscillation along target tracking. Due to the close relation to the work presented in this thesis the details of these algorithms [25, 39] are presented in Chapter 4.

The secant model is also used in Bonković et al. [5] with the so-called *population-based generalization* method to update the composite Jacobian. The major disadvantage of this method is considerably more computational cost and complexity of the Jacobian calculation compared to the Broyden method. The trade off between Jacobian estimation complexity and system performance improvement has not been investigated.

Miura et al. [46] present an uncalibrated visual servoing method using a modified simplex method and a Newton-like method to optimally move the robot to a desired position. One of the major problems is that an incorrect Jacobian estimation sometimes occurs due to the large motion of the robot between the vertices of the simplex. A variety of different methodologies are proposed to approximate the Jacobian. For examples, a depth-independent Jacobian proposed by Wang et al. [78] is used to estimate linearized camera parameters on-line. Qian and Su [63] propose to use the Kalman filtering technique to approximate the Jacobian components. Similarly, Lv and Huang [44] introduce using a Kalman filter to estimate the Jacobian elements while using a fuzzy logic adaptive controller to improve stability. These algorithms do not study uncalibrated visual control for large residual problems.

Bilen et al. [4] present a experimental comparison of calibrated and uncalibrated image based visual servoing in a microsystem application that requires high precision. Based on their experiment, the calibrated method performed better than the uncalibrated visual servoing with better settling time, accuracy, and precision if timing is

the task priority. However, there was not large difference between the two approaches. Uncalibrated visual servoing gives more flexibility control to the task since camera calibration is tedious and error prone.

Ozgur and Unel [52] present an experimental validation of the dynamic quasi-Gauss-Newton algorithm presented in [57, 59] for micropositioning and trajectory following tasks. The dynamic quasi-Gauss-Newton algorithm is shown to position and trajectory track in a robust manner with micron accuracies. An alternative controller called Optimal controller [64] is used to evaluate the performance of the dynamic quasi-Gauss-Newton algorithm with its dynamic Gauss-Newton controller. the dynamic Gauss-Newton controller yields better results in positioning and following a square trajectory while the Optimal controller performs better for circular and sinusoidal trajectory tracking. It is mentioned that if time is crucial for the task that the calibrated visual control might be a better choice. This work confirms the feasibility of the algorithms proposed in [57, 59], even for microassembly tasks where a high level of precision is required.

To improve convergence and precision, this research develops a methodology for adaptively selecting an appropriate forgetting factor λ_k at each iteration. Due to the limited number of RLS algorithms presented for uncalibrated visual servoing, there exists little literature about an adaptive λ_k . One important work is presented in [29] where an adaptive recursive least square (ARLS) algorithm is presented. The λ_k value is calculated by solving an optimization problem. Then it is used in the dynamic quasi-Gauss-Newton method [59] for uncalibrated visual servoing control. The ARLS shows a smaller mean-square error as compared to using a fixed forgetting factor. More details of this work are presented in Chapter 5. Since the variable forgetting factor (VFF) algorithms have been widely studied in RLS adaptive filtering to improve the performance of the RLS algorithm, various VFF studies are reviewed in Section 5.3.

1.2.3 Summary

Although several uncalibrated visual servo system have been developed with various methods to approximate the Jacobian, only a few studies focus on large residual problems which are restricted to only tracking a stationary target. The dynamic quasi-Gauss-Newton algorithms in [57, 59] are shown to be the most robust and stable algorithms in both simulations and experiments in a number of studies and the research presented in this thesis uses them as a foundation.

It should be noted that this section gives an overview of the previous work related to uncalibrated visual servoing for large residual problems and more detail literature references are given as individual topics are discussed in each chapter.

1.3 Contribution

This work develops a novel uncalibrated visual guided control with an adaptive forgetting factor for large residual problems. To solve a difficulty encountered in the dynamic quasi-Gauss-Newton algorithms [57, 59] for handling large residual problems, the full quasi-Newton method is investigated. The full quasi-Newton is argued to offer superior tracking due to the inclusion of the residual S_k term in the Hessian approximation. Despite the fact that the residual S_k is usually difficult to determine analytically, various algorithms to approximate the Hessian matrix are proposed. One solution is to approximate the whole Hessian using the *dynamic BFGS (DBFGS)* algorithm. The second solution is to approximate the residual \hat{S}_k using the *modified BFGS (MBFGS)* or *dynamic full Newton method with BFGS (DFN-BFGS)* algorithms, by assuming that the Jacobian J_k is already available.

To ensure fast convergence and stability yet attain robust tracking performance, a switching algorithm that alternates between the proposed full quasi-Newton method

and the dynamic quasi-Gauss-Newton method for a given switching criterion is introduced and has lead to the switching DBFGS-DB, the MBFGS-DB, and the DFN-BFGS-DB algorithms.

To further improve the performance of the switching algorithms, an adaptive forgetting factor λ_k called the *dynamic adaptive forgetting factor (DAFF)* is introduced. This is a heuristic method to adapt λ_k with respect to the image error norm $\|f_k\|$. Although there are several existing variable forgetting factor (VFF) schemes presented in the RLS adaptive filtering literature, these algorithms are complex and do not appear effective for uncalibrated visual servoing.

Simulation results show that the switching MBFGS-DB algorithm with the DAFF method consistently yields the best results for a variety of robot degrees-of-freedom, camera configurations, trajectories, and target speeds. The DAFF algorithm is shown to significantly offer the best overall tracking accuracy and convergence, especially in the presence of noise. The number of cameras and the camera arrangement are shown to significantly affect noise compensation. These results validate the effectiveness of the switching MBFGS-DB algorithm with the DAFF scheme to improve tracking performance for large residual problems in the presence of noise.

1.4 Organization

The thesis is organized in the following manner:

Chapter 1 discusses the motivation of the research, the contributions, the uncalibrated visual servoing literature review, and outlines the study.

Chapter 2 provides the theoretical background of Newton and quasi-Newton methods for solving unconstrained optimization problems.

Chapter 3 reviews the dynamic quasi-Gauss-Newton algorithms either with or without partitioning for Broyden's method developed by Piepmeyer et al. [57, 59].

Since this work is built on the dynamic quasi-Gauss-Newton algorithms, the mathematical background of these algorithms are fundamental to the study in this thesis. A few major difficulties of these algorithms, which have lead to derivation of the proposed algorithms, are discussed.

Chapter 4 develops the DBFGS, DFN-BFGS, MBFGS methods for residual approximation for large residual problems. The derivation of these algorithms are analogous to the BFGS algorithm. Two distinct methods for solving large residual problems are presented: i) approximation of the whole Hessian matrix, ii) approximation of the residual term included in the Hessian matrix with an assumption that linear portion $J^T J$ is available. The DBFGS algorithm is an approximation of the whole Hessian matrix, while the DFN-BFGS and the MBFGS algorithms are the approximations of the residual terms. Unlike the Hessian approximation using the BFGS method, the DBFGS algorithm includes the dynamic term $\frac{\partial f}{\partial t} h_t$ into the secant equation used for the Hessian approximation. The DFN-BFGS method, on the other hand, is derived by directly employing the BFGS algorithm to approximate the residual term, while the MBFGS method is derived by modifying a denominator of a term in the DFN-BFGS formula. Its significance is due to the fact that curvature information of the objective function is enforced into the residual approximation. A convergence proof is given for the MBFGS method. A hybrid between the dynamic quasi-Gauss-Newton method and the (full) Newton method, with the approximated residual using a proposed residual approximation method is developed. A discussion of the alternative hybrid methods and a heuristic selection of the switching criterion are presented.

Chapter 5 presents the derivation of the DAFF algorithm. This method is inspired by observing λ_k behavior that is in analogy to a step response of a first-order differential system. Various VFF algorithms that are widely studied in RLS adaptive filtering areas are also reviewed. The effect of the DAFF method on improving tracking performance is compared with various VFF algorithms in Chapter 6.

Chapter 6 simulates the proposed switching algorithms, the switching MBFGS-DB, DBFGS-DB, and DFN-BFGS-DB algorithms on three and six DOF robot with various camera configurations and trajectories. Comparison is made between the proposed switching algorithms and dynamic quasi-Newton methods with or without partitioning for Broyden’s method. Overall, the switching MBFGS-DB algorithm provides the best results in terms of RMS errors, t_s , and stability for a higher DOF robot, a complex trajectory, or a fast target speed. The different VFF algorithms presented in Chapter 5 are investigated to validate tracking improvement. Comparison is made between the DAFF method, various existing VFF algorithms, and a fixed forgetting factor that are implemented in the switching MBFGS algorithms for large residual problems with the presence of noise. The switching MBFGS with the DAFF algorithm consistently yields the fastest convergence and the smallest RMS tracking for a variety of trajectories and target speeds. The effects of multiple cameras and the camera arrangement on noise compensation are investigated. The effect of the Levenberg-Marquardt algorithm (LMA) implemented with various switching algorithms and the adaptive forgetting factor schemes are presented. The LMA only marginally improves tracking performance for all tested cases.

Chapter 7 summarizes the results and discusses possible future research.

In summary, this work derives and simulates uncalibrated visual servoing with an adaptive forgetting scheme for large residual problems. This work is the first successfully developed visual guided control to handle large residual problems for uncalibrated moving target tracking.

CHAPTER II

NEWTON'S METHOD AND QUASI-NEWTON METHODS BACKGROUND

In this study visual servoing is formulated as a nonlinear optimization problem and algorithms such as Newton's method and quasi-Newton methods are typically used for finding a solution. The theoretical background of Newton's method for scalar and multi-variable functions is presented in Section 2.1. Quasi-Newton methods in which the derivative of a function is approximated rather than analytically calculated is discussed in Section 2.2. Section 2.3 presents the modified Newton's method in which a Hessian matrix is adjusted to guarantee positive definiteness. Section 2.4 and Section 2.5 provide the basic fundamentals of line search and trust-region methods respectively. The summary of this chapter is presented in Section 2.6.

2.1 *Newton's Method*

2.1.1 Newton's Method for a One-Variable Scalar Function

In numerical analysis, *Newton's method* or *the Newton-Raphson method* is often used to solve for the root of a function. For a one-variable scalar function $f(x)$ where $x \in \mathbb{R}$ Newton's method can be derived from Taylor series approximation of $f(x)$ around x_k ,

$$f(x) = f(x_k) + f'(x_k)(x - x_k) + \dots \quad (2.1)$$

Dropping the higher order term yields an *affine model*¹ $m_k(x)$ that retains the constant and linear portion of $f(x)$,

$$m_k(x) = f(x_k) + f'(x_k)(x - x_k) \quad (2.2)$$

Figure 2.1 shows a geometrical representation of an affine model in which $f(x)$ is approximated about x_k .

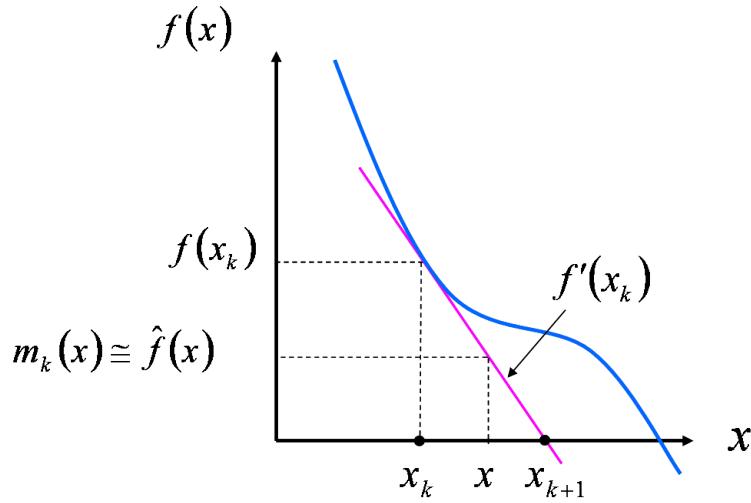


Figure 2.1: A geometrical representation of an affine model.

Newton's method is obtained by setting (2.2) to zero,

$$0 = f(x_k) + f'(x_k)(x - x_k) \quad (2.3)$$

where $f'(x)$ is the first-order derivative of $f(x)$. Solving (2.3) for $x = x_{k+1}$ gives

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \quad (2.4)$$

This is called *Newton's method* for successively finding a root of a function $f(x)$. Newton's method quickly converges to a solution if the initial value (x_0) is relatively close to the solution (x^*). The process is repeated until a sufficiently-accurate root is found.

¹an affine model corresponds to an affine subspace through $(x, F(x))$ in which a line does not necessarily pass through the origin (a line must pass the origin in a linear subspace)

If $f(x)$ is a linear function, Newton's method locates the solution in one iteration. For nonlinear problems, this process generates a sequence of points that is expected to converge to a solution. For a complex nonlinear function, a quadratic model is often used to approximate $f(x)$.

Newton's method is also well-known to be used for finding local minima or local maxima of an objective function $F(x)$, which is known as optimization problem. In this case, an affine model $M_k(x)$ of $F(x)$ is

$$M_k(x) = F(x_k) + F'(x_k)(x - x_k) \quad (2.5)$$

Newton's method is obtained by solving for the root of $F'(x) = 0$ as

$$0 = F'(x_k) + F''(x_k)(x - x_k) \quad (2.6)$$

Solving (2.6) yields

$$x_{k+1} = x_k - \frac{F'(x_k)}{F''(x_k)} \quad (2.7)$$

where $F''(x)$ is the second-order derivative of the objective function $F(x)$.

2.1.2 Newton's Method for a Multiple-Variable Function

Consider a nonlinear function $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$ that is continuously differentiable at $x \in \mathbb{R}^m$. A solution x^* where $f(x^*) = 0$ can be found using the Newton's method similar to (2.4). An affine model m_k that approximates $f(x)$ about x_k is

$$f(x) \cong m_k(x) = f(x_k) + J_k(x - x_k) \quad (2.8)$$

where $J_k \in \mathbb{R}^{m \times n}$ is the *Jacobian* matrix, $J_k = \left. \frac{\partial f}{\partial x} \right|_{x_k}$. A successive x_{k+1} can be approximated from the current x_k by solving

$$m_k(x_{k+1}) = f(x_k) + J_k(x_{k+1} - x_k) = 0$$

to give

$$x_{k+1} = x_k - J_k^{-1}f(x_k) \quad (2.9)$$

This is known as *Newton's method* for solving a set of m equations.

For unconstrained minimization problems, consider a scalar objective function $F(x) : \mathbb{R}^m \rightarrow \mathbb{R}$ where $m > 1$ is continuously differentiable at $x \in \mathbb{R}^m$ and a solution x^* is a locally/globally minima of the objective function $F(x)$. The derivation of Newton's method for m -dimensional minimization problems is similar to the one-dimensional function in Section 2.1.1. The Taylor series of $F(x)$ can be expressed as

$$F(x) = F(x_k) + \nabla F_k(x - x_k) + O(x^2) \quad (2.10)$$

where ∇F_k is the first partial derivatives of $F(x)$ with respect to the m variables and is called the *gradient* of F at x_k . Dropping the higher order term $O(x^2)$ in (2.10) yields an affine model $M_k(x)$ of $F(x)$ around x_k ,

$$F(x) \cong M_k(x) = F(x_k) + \nabla F_k(x - x_k) \quad (2.11)$$

The minimum of $F(x)$ can be found by solving

$$\begin{aligned} 0 &= \frac{\partial M_k(x)}{\partial x} \\ &= \nabla F_k + H_k(x - x_k) \end{aligned} \quad (2.12)$$

where $H_k = \nabla^2 F_k = \left. \frac{\partial^2 F(x)}{\partial x^2} \right|_{x_k}$ is the *Hessian* matrix of $F(x)$ at x_k . The Hessian H is an n by n matrix and is always symmetric if $F(x)$ is twice continuously differentiable [19]. Solving (2.12) at $x = x_{k+1}$ gives

$$(x_{k+1} - x_k) = -H_k^{-1} \nabla F_k \quad (2.13)$$

$$x_{k+1} = x_k - H_k^{-1} \nabla F_k \quad (2.14)$$

Equation (2.14) is known as *Newton's method* for solving a minimization problem where $-H_k^{-1} \nabla F_k$ is called the *Newton direction*. Figure 2.2 shows a pseudo-code algorithm utilizing Newton's method to solve unconstrained minimization problems.

A spacial case of unconstrained optimization is a nonlinear least-squares problem. It is well-known in data fitting applications in which the best fit is obtained by

Pseudo-code: Newton's method

Given: $F : \mathbb{R}^m \rightarrow \mathbb{R}$; $x \in \mathbb{R}^m$; $H \in \mathbb{R}^{n \times n}$; $\nabla F \in \mathbb{R}^{m \times 1}$

Initialize ϵ , x_0 , and x_1

Calculate $F(x_0)$

for $k = 1, \dots$ **do**

Calculate $F(x_k)$, ∇F_k , and H_k

Find $x^* \in \mathbb{R}^m$ for which $F(x)$ is minimized

if $\left| \frac{F(x_k) - F(x_{k-1})}{F(x_{k-1})} \right| < \epsilon$ **then**

break // Convergence criterion met.

end if

$x_{k+1} = x_k - H_k^{-1} \nabla F_k$ // $(k + 1)^{\text{th}}$ solution

end for

Figure 2.2: Pseudo-code of Newton's method for solving unconstrained optimization problems

minimizing the sum of squared error between a measured value and the approximated value from a model. Since visual servoing problems seek the robot joint angles θ that minimize the error between the robot and target features, they are cast as nonlinear least-squares problems. For this reason, Newton's and other related methods used for solving nonlinear least-squares problems in general are focused on in this chapter, then nonlinear least-squares visual servoing problems are discussed in Chapter 3.

The objective function $F(x)$ to be minimized is a function of squared error function $f(x)$,

$$\underset{x \in \mathbb{R}^m}{\text{Minimize}} \quad F(x) = \frac{1}{2} f^T(x) f(x) \quad (2.15)$$

where the error function $f(x) : \mathbb{R}^m \rightarrow \mathbb{R}^n$ is continuously differentiable at $x \in \mathbb{R}^m$. A

solution of (2.15) can be found by Newton's method given in (2.14). In this context,

$$\nabla F_k = J_k^T f(x) \quad (2.16)$$

$$H_k = J_k^T J_k + S_k \quad (2.17)$$

where

$$J_k = \frac{\partial f_k}{\partial x}$$

$$S_k = \frac{\partial J_k^T}{\partial x} f_k$$

Substituting them in (2.14) gives

$$x_{k+1} = x_k - (J_k^T J_k + S_k)^{-1} J_k^T f(x_k) \quad (2.18)$$

S_k is the second-order term of the Hessian H_k and is known as the *residual*. Often the residual S is computationally expensive so it is assumed to be zero and can be neglected if x_0 is in the neighborhood of a solution x^* (see Chapter 3). Then (2.18) reduces to

$$x_{k+1} = x_k - (J_k^T J_k)^{-1} J_k^T f(x_k) \quad (2.19)$$

and is known as the *Gauss-Newton method*.

Newton's method is a powerful technique that gives quadratic convergence if ∇F_k at the solution x^* is nonzero. However, the initial value x_0 is required to be sufficiently near the true solution x^* to guarantee convergence so Newton's method is often referred as a local technique. Further, the method will fail if the Hessian H_k is singular. The other main drawback is due to the requirement of the analytical derivatives of $F(x)$. When the derivatives are difficult to analytically obtain or unavailable and an approximation of either ∇F_k or the Hessian H_k is used instead, then Newton's method is called a *quasi-Newton method*.

Without a loss of generality, the capital letter described an objective function $F(x)$ that is a scalar function while the lower case letter described a function such as $f(x)$ where $x \in \mathbb{R}^n$.

2.2 Quasi-Newton Methods

In many applications a function $F(x)$ cannot be analytically expressed so its derivatives are not available and must be approximated. Approximation schemes such as a *finite-difference approximation* or a *secant approach* can be used to estimate the Jacobian J_k and the Hessian matrix H_k . The secant approximation of J_k is typically simpler than the secant approximation of the Hessian matrix H_k . The most popular secant method used for the Jacobian estimation is known as Broyden's method and is discussed in Section 2.2.1. Secant methods used for estimating H_k are reviewed in Section 2.2.2.

2.2.1 Broyden's method

Recall that the objective function $F(x)$ for a nonlinear least-squares problem is described as

$$F(x) = \frac{1}{2} f^T(x) f(x) \quad [2.15]$$

A solution of (2.15) can be found by Newton's method given in (2.18) or in (2.19). If a analytical $J_k = \left. \frac{\partial f}{\partial x} \right|_{x_k}$ is not available, the approximate \hat{J}_k can be determined using a secant technique.

The affine model of the error function $f(x)$ is defined as in (2.8),

$$m_k(x) = f(x_k) + J_k(x - x_k) \quad (2.8)$$

The model m_k is required to exactly represent the function at $x = x_{k-1}$ as

$$m_k(x_{k-1}) = f(x_{k-1}) \quad (2.20)$$

Substituting (2.20) into (2.8) at $x = x_{k-1}$ yields

$$f(x_{k-1}) = f(x_k) + J_k(x_{k-1} - x_k) \quad (2.21)$$

and rearranging gives,

$$J_k(x_k - x_{k-1}) = f(x_k) - f(x_{k-1}) \quad (2.22)$$

Equation (2.22) is known as the *secant equation*. Substituting $h_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = f(x_k) - f(x_{k-1})$ gives

$$J_k h_{k-1} = y_{k-1} \quad (2.23)$$

One of the most successful approaches to approximate J_k was introduced by C. Broyden in 1965 [19]. Broyden suggested using a rank-one update to estimate Jacobian \hat{J}_k from the previous Jacobian \hat{J}_{k-1} by taking a solution of secant equation (2.23) that gives the minimal changes between the current and previous Jacobians, i.e., minimizing the Frobenius norm of the Jacobian difference $\|\hat{J}_k(x) - \hat{J}_{k-1}(x)\|_F$ as

$$\hat{J}_k = \hat{J}_{k-1} + \frac{(y_{k-1} - \hat{J}_{k-1}h_{k-1}) h_{k-1}^T}{h_{k-1}^T h_{k-1}} \quad (2.24)$$

which is called *Broyden's method*. Dennis and Schnabel [19] give a proof to show that the quasi-Newton method using the Jacobian \hat{J}_k from Broyden's method converges to x^* superlinearly if the initial x_0 is close to x^* and the initial Jacobian $\hat{J}(x_0)$ is close to the actual initial Jacobian $J(x_0)$ if $\hat{J}(x^*)$ is nonsingular.

2.2.2 Hessian Approximation

When the Hessian matrix H requires a significant calculation cost, a secant method similar to the one presented in Section 2.2.1 can be used to efficiently approximate H . Hessian approximation methods, including the Davidon-Fletcher-Powell (DFP) update, the PSB (Powell-symmetric-Broyden) update, and the Broyden-Fletch-Goldfarb-Shanno (BFGS) method, are briefly discussed in this section.

Secant techniques similar to those used for Jacobian estimation can be applied to the Hessian approximation. The secant equation for the Hessian matrix H_k , analogous

to equation (2.23), is

$$H_k h_{k-1} = g_{k-1} \quad (2.25)$$

where $g_{k-1} = \nabla F(x_k) - \nabla F(x_{k-1})$. Similar to the Jacobian approximation, Broyden's method can be used to approximate H_k as

$$\hat{H}_k = \hat{H}_{k-1} + \frac{(g_{k-1} - \hat{H}_{k-1}h_{k-1})h_{k-1}^T}{h_{k-1}^T h_{k-1}} \quad (2.26)$$

where \hat{H}_k is an approximation of H_k . However, \hat{H}_k obtained from (2.26) is not guaranteed to be symmetric even if \hat{H}_k is symmetric. Since the Hessian matrix is always symmetric and often positive definite, the Powell-symmetric-Broyden (PSB) update can be used to ensure symmetric

$$\begin{aligned} \hat{H}_k &= \hat{H}_{k-1} + \frac{(g_{k-1} - \hat{H}_{k-1}h_{k-1})h_{k-1}^T + h_{k-1}(g_{k-1} - \hat{H}_{k-1}h_{k-1})^T}{h_{k-1}^T h_{k-1}} \\ &\quad - \frac{h_{k-1}^T(g_{k-1} - \hat{H}_{k-1}h_{k-1})h_{k-1}h_{k-1}^T}{(h_{k-1}^T h_{k-1})^2} \end{aligned} \quad (2.27)$$

The details of this method is presented in [19]. Even though this method converges superlinearly [7], a \hat{H}_{k+1} update using the PSB method may not be positive definite. The solution of Newton's method or a quasi-Newton method gives a critical point that can be either a minima, a maxima, or a saddle point. In order to ensure that this critical point is a minima, it requires the Hessian H_k to be positive definite. For this reason, Hessian approximations that preserve positive definiteness such as BFGS and DFP are more popular methods in solving nonlinear optimization.

The BFGS method [19] is

$$\hat{H}_k = \hat{H}_{k-1} + \frac{g_{k-1}g_{k-1}^T}{g_{k-1}^T h_{k-1}} - \frac{\hat{H}_{k-1}h_{k-1}h_{k-1}^T\hat{H}_{k-1}}{h_{k-1}^T\hat{H}_{k-1}h_{k-1}} \quad (2.28)$$

The DFP method [19] is

$$\begin{aligned} \hat{H}_k &= \hat{H}_{k-1} + \frac{(g_{k-1} - \hat{H}_{k-1}h_{k-1})g_{k-1}^T + g_{k-1}(g_{k-1} - \hat{H}_{k-1}h_{k-1})^T}{g_{k-1}^T h_{k-1}} \\ &\quad - \frac{h_{k-1}^T(g_{k-1} - \hat{H}_{k-1}h_{k-1})g_{k-1}g_{k-1}^T}{(g_{k-1}^T h_{k-1})^2} \end{aligned} \quad (2.29)$$

The main advantages of the BFGS and DFP methods is due to the resulting positive-definite \hat{H}_k approximation if the initial Hessian \hat{H}_0 is positive definite. The DFP method sometimes generates a numerically singular Hessian estimation so the BFGS method is preferred over the DFP method.

Regardless of the method for approximating the Hessian \hat{H} , the quasi-Newton step h_k is similar to the Newton step (2.13) and a update x_{k+1} is similar to the equation (2.14) (only that now the Hessian H_k is substituted with \hat{H}_k) as

$$h_k = -\hat{H}_k^{-1} \nabla F_k \quad (2.30)$$

or

$$x_{k+1} = x_k - \hat{H}_k^{-1} \nabla F_k \quad (2.31)$$

where $h_k = x_{k+1} - x_k$. The iteration quadratically converges to a minimum if the starting value x_0 is sufficiently close to the solution x^* . If the Hessian matrix \hat{H}_k is positive definite, the Newton direction $-\hat{H}_k^{-1} \nabla F(x_k)$ is guaranteed to be a descent direction, i.e., x_{k+1} yields a decreasing value of the objective function $F(x)$ such that $F(x_k + h_k) < F(x_k)$. In practice, the Hessian matrix \hat{H}_k is not necessarily positive definite at a point far from a minimum so the approximated model may not have a minimum. As a result, a modification suggested to assure the positiveness of \hat{H}_k is discussed in Section 2.3.

2.3 Damped Hessian

A critical point obtained from Newton's method or a quasi-Newton method is guaranteed a minima only if the Hessian matrix \hat{H}_k is positive definite. If the Hessian H_k or \hat{H}_k is not positive, some useful modifications are suggested in literature such as [19, 42, 45, 50] by including an additional term such as

$$H_{d,k} = H_k + \mu_k I \quad (2.32)$$

where I is the identity matrix and μ_k is a scalar multiplier that is sometimes referred as a *damping parameter*. If \hat{H}_k is safely positive definite then $\mu_k = 0$. Otherwise, $\mu_k > 0$ is selected to be sufficiently large so $H_{d,k}$ is guaranteed to be positive definite. The modified Hessian is sometimes called the *damped Hessian* and is used in a quasi-Newton method as

$$h_k = -H_{d,k}^{-1} \nabla f(x_k) \quad (2.33)$$

or

$$x_{k+1} = x_k - H_{d,k}^{-1} \nabla f(x_k) \quad (2.34)$$

This is called the *modified Newton's method* in [19].

The damped Hessian was first introduced by Levenberg [42] for nonlinear least-squares curve fitting applications. He proposed the *damped Gauss-Newton method* as

$$(J_k^T J_k + \mu_k I) h_k = -J_k^T f(x_k) \quad (2.35)$$

This is similar to the Gauss-Newton method in (2.19) except that now the Hessian $H_k = J_k^T J_k$ is replaced by $H_{d,k} = J_k^T J_k + \mu_k I$ where μ_k is updated at each iteration. The update is based on the rate of objective function reduction. For example, if the reduction rate of the objective function is fast, μ_k is decreased. On the other hand, if the reduction rate is inadequate, μ_k may be increased. Varying μ in fact interpolates the resulting solution between the Gauss-Newton algorithm (smaller μ) and gradient descent method (larger μ). The greatest disadvantage of this method occurs when μ becomes too large and inverting $(J^T J + \mu I)$ is not meaningful. As a result, Marquardt [45] proposed

$$(J_k^T J_k + \mu_k \mathbf{diag}(J_k^T J_k)) h_k = -J_k^T f(x_k) \quad (2.36)$$

in which the identity matrix I is substituted with the diagonal of $J_k^T J_k$ and the equation (2.36) is known as the *Levenberg-Marquardt algorithm (LMA)*. The brief details

of the LMA implementation developed by [47] to calculate the damping parameter μ and a diagonal matrix, instead of using $J^T J$ as in (2.36), are discussed in Chapter 6.

Marquardt [45] proved that the solution of minimizing the damped model (using the damped Hessian in (2.32)) is the same as of minimizing the original model in a restricted region. This region is referred to the *trust region* where a ball, for example, centered about the current iteration is created using a local model (usually a quadratic model) and is restricted to values which accurately model the function. A new iteration is limited to remain inside this a trusted region. This is an important concept in nonlinear optimization problems. In fact, there are two strategies that can be implemented with Newton's method and quasi-Newton methods to improve stability and convergence namely line search and trust region methods.

2.4 Line Search

In the line search technique a direction p_k is first determined and then a search along this direction from the current x_k is performed so that the next iteration yields a lower objective function value,

$$F(x_{k+1}) < F(x_k) \quad (2.37)$$

The objective of line search algorithm is to find a *step length* α along p_k by solving one-dimensional minimization subproblem,

$$\underset{\alpha>0}{\text{minimize}} \quad F(x_k + \alpha p_k) \quad (2.38)$$

A method that provides an exact solution of (2.38) is called an *exact line search*. On the other hand, if a solution of (2.38) is loosely estimated, i.e., a calculation of α yields a sufficient reduction in $F(x_k + \alpha p_k)$ so that it is approximately close to a minima, the method is called an *inexact line search*. An exact line search is usually expensive and unnecessary so an inexact line search is more practical.

The direction p_k that has been considered up to this point is the Newton direction or a quasi-Newton direction. For simplicity let $s_k \equiv H_k^{-1}\nabla F_k$ is the Newton direction and $\hat{s}_k \equiv \hat{H}_k^{-1}\nabla F_k$ is a quasi-Newton direction. The direction s_k or \hat{s}_k does not always yield a descent direction so $F(x_k + \alpha s_k)$ may not reduce. These directions are only guaranteed to be descent directions if H_k or \hat{H}_k is positive definite. Therefore, H_k or \hat{H}_k is always assumed to be positive definite when implemented with a line search method in this study. This requirement further emphasizes the advantage of the Hessian modification in Section 2.3. The next iterate x_{k+1} using a quasi-Newton direction, for example, is

$$x_{k+1} = x_k + \alpha_k \hat{s}_k \quad (2.39)$$

The other common search direction p_k is the *steepest-descent direction* where $p_k = -\frac{\nabla F_k}{\|\nabla F_k\|_2}$. This direction is the steepest downhill direction from the current iteration x_k . Without a loss of generality, p_k is used for the steepest-descent direction, while s_k represents the Newton direction for the remainder of this study. An exact line search example from [1] is shown in Figure 2.3 where a search along the steepest-descent direction starts from x_0 then converges to the solution x^* after a few iterations.

One advantage of the steepest descent method is that it only requires ∇F_k and not the Hessian H_k . However, in general the steepest descent method is not recommended due to its slower convergence compared to Newton's method or quasi-Newton methods. Moreover, this method is scale variance of x , while the Newton direction is not [19]. Since the steepest-descent method is less preferable, this section only focuses on Newton's method or a quasi-Newton method with an inexact line search algorithm.

Even though x_{k+1} satisfies the simple condition in (2.37), there is no assurance that x_{k+1} will converge to a solution. As a result, the inequality conditions known as the *Wolfe conditions* are often recommended with inexact line search algorithms [50],

1. $F(x_k + \alpha_k s_k) \leq F(x_k) + c_1 \alpha_k s_k^T \nabla F_k$

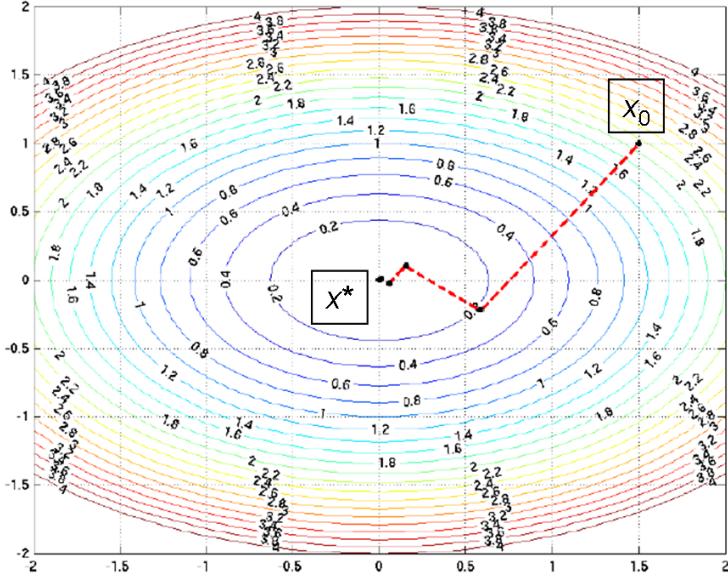


Figure 2.3: An example of an exact line search [1] starting from x_0 to reach the solution x^* along the steepest-descent direction at each iteration.

$$2. \quad s_k^T \nabla F(x_k + \alpha_k s_k) \geq c_2 s_k^T \nabla F_k$$

where $0 < c_1 < c_2 < 1$. It is typically recommended to select c_1 to be small and c_2 to be much larger. For example, Nocedal and Wright [50] recommend $c_1 = 10^{-4}$ and $c_2 = 0.9$ for the Newton or a quasi-Newton method. The first condition is known as the *Armijo rule* and imposes a sufficient reduction in $F(x)$. The second condition is known as the *curvature condition* and enforces a sufficient reduction of the slope of $F(x)$ for the step length α_k .

The search proceeds along a quasi-Newton direction \hat{s}_k until (2.37) is achieved. If $\alpha = 1$, the quasi-Newton step becomes the full quasi-Newton step which is usually recommended whenever (2.37) is satisfied [19]. If $x_k + \alpha_k \hat{s}_k$ is not acceptable then α_k will be reduced until a desired value is found; this strategy is called *backtracking method*. Common methods such as bisection, the golden section search, and polynomial/cubic methods are used to reduce the step length α .

Since a backtracking strategy prevents small steps, the curvature condition is

Pseudo-code: Backtracking line search method with the Armijo rule

Given: $F : \mathbb{R}^m \rightarrow \mathbb{R}$; $x \in \mathbb{R}^m$; $\alpha, \rho \in \mathbb{R}$; $\nabla F \in \mathbb{R}^{n \times 1}$; $c_1 \in (0, \frac{1}{2})$; $0 < l < u < 1$

Initialize ϵ , x_0 , and x_1

for $k = 1, \dots$ **do**

Determine a descent direction s_k

Start with $\alpha_k = 1$

while $F(x_k + \alpha_k s_k) \leq F(x_k) + c_1 \alpha_k s_k^T \nabla F_k$ **do**

$\alpha_k = \rho \alpha_k$ // for some $\rho \in [l, u]$

$\rho = \text{line_search}(\hat{h}_k)$ // Determine the line search gain

end while

$x_{k+1} = x_k + \alpha_k s_k$ // $(k+1)^{\text{th}}$ solution

end for

Figure 2.4: Pseudo-code for backtracking line search with Newton's method

typically not necessary in practice. The pseudo-code summary of backtracking line-search algorithm with only the Armijo rule is illustrated in Figure 2.4. Then pseudo-code is used to summarize a quasi-Newton method using BFGS to approximate the Hessian matrix with the backtracking line-search algorithm in Figure 2.5.

When the full quasi-Newton step does not satisfy condition (2.37), it might indicate that the model does not properly reflect the actual function in a neighborhood around the current x_k . Since line search algorithms only continually search along the quasi-Newton direction which is obtained from a ‘not-so-good’ model in this case, other searching methods known as *trust-region methods* may offer more suitable approaches.

Pseudo-code: The BFGS algorithm with backtracking line-search technique

Given: $F : \mathbb{R}^m \rightarrow \mathbb{R}$; $x \in \mathbb{R}^m$; $H \in \mathbb{R}^{n \times n}$; $\nabla F \in \mathbb{R}^{n \times 1}$; $s \in \mathbb{R}^{n \times 1}$; $\alpha, \rho \in \mathbb{R}$; $c_1 \in (0, \frac{1}{2})$; $0 < l < u < 1$

Initialize ϵ , x_0 , and x_1

for $k = 1, \dots$ **do**

Calculate $F(x_k)$, ∇F_k , and H_k

if $\left| \frac{F(x_k) - F(x_{k-1})}{F(x_{k-1})} \right| < \epsilon$ **then**

break // Convergence criterion met.

end if

Calculate \hat{H}_k

$$\begin{aligned}\hat{h}_{k-1} &= x_k - x_{k-1} \\ g_{k-1} &= \nabla F_k - \nabla F_{k-1}\end{aligned}$$

$$\hat{H}_k = \hat{H}_{k-1} + \frac{g_{k-1}g_{k-1}^T}{g_{k-1}^T h_{k-1}} - \frac{\hat{H}_{k-1}h_{k-1}h_{k-1}^T\hat{H}_{k-1}}{h_{k-1}^T\hat{H}_{k-1}h_{k-1}}$$

$$\hat{s}_k = -\hat{H}_k^{-1}\nabla F_k \quad // \text{ Determine the BFGS step}$$

Start with $\alpha_k = 1$

while $F(x_k + \alpha_k \hat{s}_k) \leq F(x_k) + c_1 \alpha_k \hat{s}_k^T \nabla F_k$ **do**

$$\alpha_k = \rho \alpha_k \quad // \text{ for some } \rho \in [l, u]$$

$$\rho = \text{line_search}(\hat{h}_k) \quad // \text{ Determine the line search gain}$$

end while

$$x_{k+1} = x_k + \alpha_k \hat{s}_k \quad // (k+1)^{\text{th}} \text{ solution}$$

end for

Figure 2.5: Pseudo-code for the BFGS algorithm with the backtracking line-search method

2.5 Trust-Region Methods

Trust region methods share some duality to line search approaches. While line search methods seek for an appropriate step size α along a known searching direction, trust region algorithms first calculate a proper size of the trust region and then choose an appropriate step direction. A trust region is referred to a restricted neighborhood around the current iteration x_k of which a local model adequately represents a function $F(x)$. For a nonlinear function, a quadratic model $M_{quad,k}$ is a more appropriate model than an affine model and it can be described as

$$F(x) \cong M_{quad,k}(x) = F(x_k) + \nabla F_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T H_k(x - x_k) \quad (2.40)$$

This local quadratic model is minimized to solve the following constrained optimization subproblem at each iteration,

$$\text{minimize} \quad M_{quad,k}(x_k + h_k) = F(x_k) + \nabla F_k^T h_k + \frac{1}{2} h_k^T H_k h_k \quad (2.41)$$

$$\text{subject to} \quad \|h_k\|_2 \leq \delta_k \quad (2.42)$$

where $\|\cdot\|_2$ refers to the Euclidean norm and $\delta_k > 0$ determines the size of a trust-region. Typically a trust region is determined as a sphere centered at the current x_k with a radius δ_k in which the update x_{k+1} is limited to only stay inside this region where the current model is trusted as shown in Figure 2.6. This is an example of a trust region of a function $F(x_1, x_2) = -10x_1^2 + 10x_2^2 + 4 \sin(x_1 x_2) - 2x_1 + x_1^4$ [76] where the red dot is the center of the trust region at $x_k = (0.7, -3.3)$ and the successive x_{k+1} is limited to be within the radius δ_k from the current x_k .

The solution of (2.41) is restricted by the size of the trust region δ_k . If this solution is undesirable, the trust-region radius will be decreased which in fact alters the direction of the resulting solution. This behavior is the major difference between a trust region method and a line search algorithm in which a search direction is never changed.

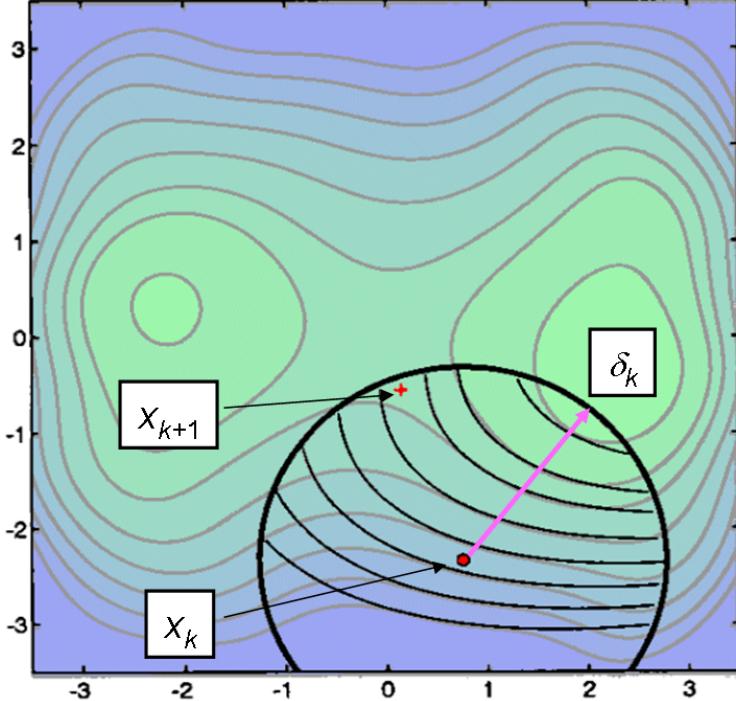


Figure 2.6: An example of a trust region [76] with a radius δ_k centered at x_k where the next update x_{k+1} is restricted to be inside the trust region.

The concept of trust region methods, though this terminology was not used originally, was introduced in [42]. Recall the damped Gauss-Newton method (2.35),

$$(J_k^T J_k + \mu_k I) h_k = -J_k^T f(x_k) \quad [2.35]$$

An improved version of (2.35) in [45] leads to the LMA as briefly described in Section 2.3. However, the LMA heuristically adjusts the damping parameter μ_k according to how $F(x)$ is effectively reduced in the previous step and no clear connection between (2.35) with trust region methods was presented in the original work. The connection between the LMA with trust region methods was firmly established in [47]. Although various strategies of calculating μ_k and a diagonal matrix instead of the identity matrix in (2.35) are introduced, this study uses the algorithm presented in [47] as a basis. The detailed history of trust region methods can be found in [9].

In [19] it is shown that the solution of (2.41) is the same as the modified quadratic model $M_{mod,k}$ in which the Hessian H_k in (2.40) is now substituted by the damped

Hessian \tilde{H}_k (2.32) as

$$F(x) \cong M_{mod,k}(x) = F(x_k) + \nabla F_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T H_{d,k}(x - x_k) \quad (2.43)$$

when h_k is constrained by

$$\|h_k\|_2 \leq \delta_k$$

So (2.41) is solved by

$$h_k(\mu_k) = -(\hat{H}_k + \mu_k I)^{-1} J_k^T f_k = -H_{d,k}^{-1} J_k^T f_k \quad (2.44)$$

If $\mu_k = 0$ and $\|h_k(0)\| \leq \delta_k$, then (2.44) is a quasi-Gauss-Newton step $h_k = -\hat{H}_k^{-1} J_k^T f_k$.

Otherwise, (2.44) is the solution of (2.41) for any unique $\mu_k > 0$ such that $\|h_k(\mu_k)\| = \delta_k$ [50].

If $\delta_k < \|H_{d,k}^{-1} J_k^T f_k\|_2$ then μ_k is needed to be solved for such that

$$\|h_k(\mu_k)\|_2 = \|H_{d,k}^{-1} J_k^T f_k\|_2 \cong \delta_k \quad (2.45)$$

which is a nonlinear equation in μ_k . Various methods are reviewed in [19, 50] to approximate μ_k in this situation.

The direction of (2.44) is altered away from a quasi-Newton direction to the steepest descent direction if the damping parameter μ_k is increased [43]. It is also mentioned in [19] that varying the value of μ_k allows a smooth changing direction of $h_k(\mu_k)$ between a quasi-Gauss-Newton direction (when $\mu_k = 0$) and the steepest-descent direction where $h_k(\mu_k) \cong -\frac{1}{\mu_k} J_k^T f_k$ when μ_k is large.

Since (2.44) is constrained to satisfy $\|h_k(\mu_k)\|_2 \cong \delta_k$, changing the size of trust region δ_k changes the direction of $h_k(\mu_k)$. When δ_k is very small, the direction of $h_k(\mu_k)$ is close to the steepest-descent direction. If the current solution x_k is far from the desired solution x^* , the algorithm uses a steepest-descent method in which it typically guarantees convergence but the rate of convergence is only linear. However, if the current x_k is close to x^* , the trust region algorithm behaves like a quasi-Gauss-Newton method; taking an advantage of a quadratic convergence rate of a quasi-Newton method.

The effects of changing μ to the change of δ are summarized in [77] as

- Decreasing μ results in increasing δ so the direction of the resultant solution moves closer to the Newton direction
- Increasing μ results in decreasing δ so the direction of the resultant solution moves closer to the steepest descending direction

The size of trust region δ_k can be controlled based on how well the predicted model fits the actual function. In general, the size of trust region increases if the current model well predicts the function and decreases if the model poorly estimates the function. The most common strategy is to compare the actual reduction of the function with a predicted value obtained from its quadratic model $M_{quad,k}$ as in [23]. The ratio ρ_k between the actual and predicted reduction of the function at current iteration can be described as

$$\rho_k = \frac{F(x_k + s_k) - F(x_k)}{M_{quad,k}(x_k + s_k) - M_{quad,k}(x_k)} \quad (2.46)$$

If ρ_k is close to unity, the current model well approximates the actual reduction of the function. Hence, the trust region can be expanded for the next iteration. If $\rho_k > 0$ but not close to unity, the trust-region radius maintains its same size. Otherwise, the trust region should be contracted since the current model poorly agrees with the actual function reduction. Examples of updating the trust-region radius can be found in [50] and [23].

2.6 Summary

In an unconstrained minimization problem a minima of an objective function $F(x)$ where $x \in \mathbb{R}^m$ is determined by using Newton's method:

$$x_{k+1} = x_k - H_k^{-1} \nabla F_k \quad [2.14]$$

where $H_k = \nabla^2 F(x) = \left. \frac{\partial^2 F(x)}{\partial x^2} \right|_{x_k}$ is the *Hessian* matrix of $F(x)$ at x_k .

For a nonlinear least-squares problem, a spacial case of unconstrained optimization, the objective function $F(x)$ is defined as

$$\underset{x \in \mathbb{R}^m}{\text{Minimize}} \quad F(x) = \frac{1}{2} f^T(x) f(x) \quad [2.15]$$

where $f(x)$ is an error function. A local minima of $F(x)$ can be calculated using Newton's method in (2.14) where

$$H_k = J_k^T J_k + S_k$$

$$\nabla F_k = J_k^T f(x_k)$$

Substituting these terms into (2.14) gives

$$x_{k+1} = x_k - (J_k^T J_k + S_k)^{-1} J_k^T f(x_k) \quad [2.18]$$

If the Jacobian J_k is not analytically available, it can be approximated using Broyden's method,

$$\hat{J}_k = \hat{J}_{k-1} + \frac{(y_{k-1} - \hat{J}_{k-1} h_{k-1}) h_{k-1}^T}{h_{k-1}^T h_{k-1}} \quad [2.24]$$

In the case that the Hessian H_k expensively requires computation cost, a variety of Hessian approximations can be used. For example, the BFGS method,

$$\hat{H}_k = \hat{H}_{k-1} + \frac{g_{k-1} g_{k-1}^T}{g_{k-1}^T h_{k-1}} - \frac{\hat{H}_{k-1} h_{k-1} h_{k-1}^T \hat{H}_{k-1}}{h_{k-1}^T \hat{H}_{k-1} h_{k-1}} \quad [2.28]$$

As a result, Newton's method using either the approximate Jacobian \hat{J}_k , the approximate Hessian \hat{H}_k , or a combination of both yields a quasi-Newton method,

$$x_{k+1} = x_k - \hat{H}_k^{-1} \nabla F_k \quad [2.31]$$

If S_k in the Hessian H_k is neglected, (2.18) becomes the Gauss-Newton method,

$$x_{k+1} = x_k - (J_k^T J_k)^{-1} J_k^T f(x_k) \quad [2.19]$$

Newton's method or a quasi-Newton method is guaranteed to converge to a minima only if H_k is positive definite. In a case that H_k is not positive definite, a damped Hessian $H_{d,k}$ is often applied as

$$x_{k+1} = x_k - H_{d,k}^{-1} \nabla f(x_k) \quad [2.34]$$

where

$$H_{d,k} = H_k + \mu_k I \quad [2.32]$$

The identity matrix I is sometimes replaced by a diagonal matrix D_k . An algorithm such as the LM algorithm is used to update the damping parameter μ_k and a diagonal matrix D_k so that $H_{d,k}$ is positive definite.

To improve stability and convergence of Newton's method or a quasi-Newton method, line searches and trust region methods are recommended. Line search methods seek for a proper step length along a known searching direction while trust region methods first calculate a size of the trust region and then select an appropriate direction.

CHAPTER III

DYNAMIC BROYDEN'S METHOD WITH RECURSIVE-LEAST-SQUARES (RLS) UPDATE

The control of mechanical systems such as a robotic manipulator often relies on an accurate nominal model. However as systems grow in size and complexity so do the models. Further, inclusion of effects that is more difficult to successfully model such as friction, backlash, and viscoelasticity may be necessary to increase response and fidelity for control. As systems rely less on modeling they usually compensate for it by increased sensing. Visual sensing is critical for many biological systems but its use in robotic applications is still relatively limited. One of the reasons is that the vision systems typically require calibration and frequently this calibration significantly drifts during operation due to operating and environmental factors. This chapter reviews an uncalibrated visual servoing algorithm using a robot manipulator to track a moving target without an a priori model of either the robot or the camera. The algorithm is introduced by Piepmeyer et al. [57, 59, 60] who develop a dynamic Broyden's method to estimate a compound Jacobian, the transformation from robot joint velocities to the target velocities on the camera plane. It is used to generate robot joint commands via a dynamic quasi-Newton method that solves a nonlinear least squares problem.

An overview of visual servoing schemes is presented in Section 3.1. Some earlier studies implementing quasi-Newton methods for image-based visual servoing problems related to this development are summarized in Section 3.2. Section 3.3 reviews fundamental development of the *dynamic Broyden's method using recursive least-squares estimation* (DBM-RLS) for a stationary camera. Then the study is extended

to the *dynamic Gauss-Newton algorithm with partitioned Broyden's method* (DGN-PBM) where a camera is attached to a robot end-effector. A few major limitations of these algorithms are discussed in Section 3.4. Then the chapter is summarized in Section 3.5.

3.1 Overview

In robotic applications visual servoing is an algorithm in which a robotic manipulator can be controlled to track a target using visual information from one or more cameras. As a target and a robotic manipulator are viewed and an error between them is determined, the robotic system is controlled to reach the target in the direction that minimizes the error. In image-based visual servoing (IBVS) systems the error is formed on the camera image planes, usually between selected feature points on the target and the robotic end-effector (EE). For properly selected features, making the errors in the image plane vanish also make errors in the task space vanish.

The velocity relationship (and similarly the small motion relationship) between robot EE feature points y in the image plane and the robot joint angles θ is

$$\dot{y} = J\dot{\theta} \quad (3.1)$$

where

$$J = J_{camera}J_{robot}$$

is the composite Jacobian that directly maps the robot joint rate $\dot{\theta}$ into the image feature velocity \dot{y} . Individually, the robot Jacobian maps joint rates into an EE twist $T = J_{robot}\dot{\theta}$, and the camera Jacobian transforms the EE twist into the image velocity $\dot{y} = J_{camera}T$ of the selected feature points. The dimensions of J_k are determined by the number of image feature coordinates and the number of actuators.

An error f is formed from the target image feature vector y^* and the EE image feature vector y in units of pixels and can be determined directly from the camera

image. For example, if a robot is tracking a moving target using a stationary camera system the error f is given as

$$f(\theta, t) = y(\theta) - y^*(t) \quad (3.2)$$

where the EE image feature vector y is only a function of the joint angles θ and the target image feature vector y^* is only a function of time t . The form of the error f is dependent on the camera system setup. Two different vision system settings are discussed in the context of a single camera though it immediately extends to multiple cameras.

1. Eye-to-hand system - the camera system is stationary and views both the end-effector and the target as in Figure 3.1. There are two distinct situations:
 - (a) Static target - the robot is controlled to reach a static location y^* and the error is $f(\theta) = y(\theta) - y^*$
 - (b) Moving target - the robot is controlled to track a moving target $y^*(t)$ and the error is $f(\theta, t) = y(\theta) - y^*(t)$

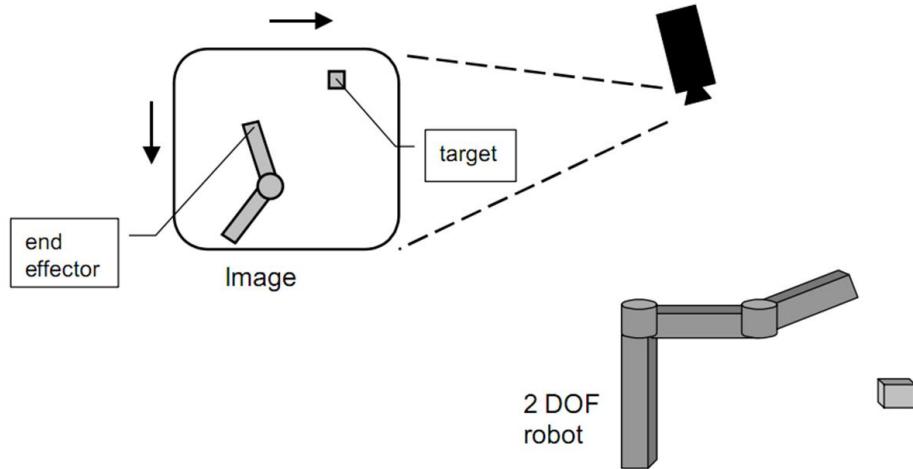


Figure 3.1: For an eye-to-hand system the camera is remote from the robot and the target.

2. Eye-in-hand system - the camera system is attached to the EE. The EE image features are seen as stationary points, with one often coinciding with the camera optical axis at the origin of the image frame, as in Figure 3.2. The two distinct situations are:

- (a) Static target - the robot is controlled to reach a static location y^* and the error is $f(\theta) = y - y^*(\theta)$
- (b) Moving target - the robot is controlled to track a moving target $y^*(t)$ and the error is $f(\theta, t) = y - y^*(\theta, t)$

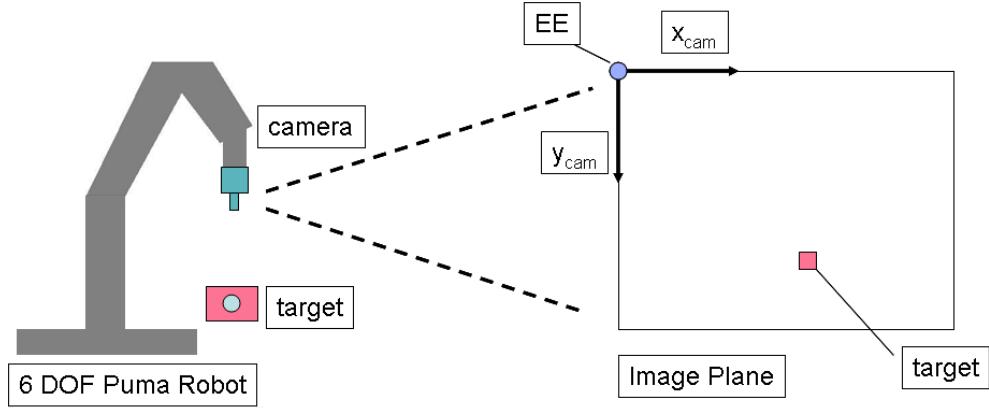


Figure 3.2: For an eye-in-hand system the camera is attached to the robot EE.

For the remainder of the development the most general case is assumed $f \equiv f(\theta, t)$ since it applies to the special cases with small modifications.

The goal of visual servo control is to determine the joint angles θ that minimize the error f at any given time t . This problem can be described as a nonlinear least squares optimization problem where the objective function $F(\theta, t)$ to be minimized is a function of the squared error between the EE image features y and the desired target image features y^* as seen on the image plane

$$F(\theta, t) = \frac{1}{2} f^T(\theta, t) f(\theta, t) \quad (3.3)$$

and is homogeneous in the units of $(pixels)^2$.

The solution of (3.3) can be found using either Newton's method (if an analytic derivative of f is available) or using quasi-Newton methods (if an analytic derivative of f is not available). Because a priori knowledge of the robot kinematic model and the camera model is assumed unknown the derivatives of f are not analytically available and quasi-Newton methods are applicable. A novel uncalibrated visual servoing that utilizes a dynamic quasi-Newton method with the dynamic Broyden's method to track a moving target is introduced by Piepmeyer et al. [57, 59, 60]. This model independent visual servoing provides robust steady-state tracking behavior when significant changes in the robot kinematic model or in the orientation of the camera occur. The theoretical fundamentals of the algorithm is briefly reviewed in Section 3.3.

3.2 Previous Work in Nonlinear Visual Servoing Optimization

The earlier developments of uncalibrated visual servoing algorithms utilizing nonlinear least-square optimization with quasi-Newton methods are done by Hosoda and Asada [34] and Jagersand, Fuentes, and Nelson [37]. Hosoda and Asada [34] introduced a nonlinear least squares method with exponential weighting matrix to recursively estimate the Jacobian \hat{J}_k at the k^{th} iteration as

$$\begin{aligned}\hat{J}_k &= \hat{J}_{k-1} + \frac{\left(\Delta f - \hat{J}_{k-1} h_\theta\right) h_\theta^T P_{k-1}}{\lambda + h_\theta^T P_{k-1} h_\theta} \\ P_k &= \frac{1}{\lambda} \left(P_{k-1} - \frac{P_{k-1} h_\theta h_\theta^T P_{k-1}}{\lambda + h_\theta^T P_{k-1} h_\theta} \right)\end{aligned}\quad (3.4)$$

where $\Delta f = f_k - f_{k-1}$, $h_\theta = \theta_k - \theta_{k-1}$, P is a full rank weighting matrix, and $0 < \lambda \leq 1$ is called the *forgetting factor*. If $\lambda = 1$, then \hat{J}_k is estimated by averaging all past information. If $\lambda < 1$, old data is deweighted by increasing powers of λ so the calculation relies more on recent data and “forgets” older data. As a rough approximation the effective number of weighted iterations used in the calculation is

given by

$$n_{memory} \approx \frac{1}{1 - \lambda}$$

so when $\lambda = 1$ the memory is infinite. Though equation (3.4) is only applicable for the time-invariant Jacobian $J(\theta)$, slow motion control can be achieved by tuning the forgetting factor λ . The control algorithm they introduced (for the k^{th} iteration) is

$$\dot{\theta} = \hat{J}^+ \dot{y}^* + \left(I - \hat{J}^T \hat{J} \right) k - K \hat{J}^T f \quad (3.5)$$

where k is a gain vector and K is a positive definite gain matrix which must be selected. They also use the same Jacobian estimation (3.4) in conjunction with a known robot Jacobian in [35] for an adaptive hybrid control algorithm. The hybrid algorithm combines vision and force sensory information to calculate robot joint rate $\dot{\theta}$ as

$$\dot{\theta} = J_{robot}^{-1} (u_{force} + u_{img})$$

where u_{force} is the force feedback output and u_{img} is the control output based on the image information that is calculated by

$$u_{img} = \hat{J}_{img}^+ (\dot{y}^* - Kf)$$

where \hat{J}_{img}^+ is the pseudo-inverse of the Jacobian estimated by (3.4). Successful simulation and experimental results for a stationary camera are provided in [34] but the case of a moving target is not addressed, except when it can be approximated as stationary.

Jagersand et al. [37] propose using Broyden's method to estimate the Jacobian for tracking a static target scenario using a stationary camera system from

$$\hat{J}_{k+1} = \hat{J}_k + \frac{(\Delta f - \hat{J}_k h_\theta) h_\theta^T}{h_\theta^T h_\theta} \quad (3.6)$$

$$\Delta f = f_k - f_{k-1}$$

$$h_\theta = \theta_k - \theta_{k-1}$$

This estimated \hat{J}_k is then used in a quasi-Newton method to solve for the next step h_θ . In order to ensure the convergence of this algorithm, a trust region method is used to automatically adapt the maximum allowable step length α . The general idea of this trust-region method is similar to the trust-region concept presented in Chapter 2. The step size δ_k is a solution of the constrained equation,

$$\underset{\|\delta_k\| < \alpha_k}{\text{Minimize}} \left\| y_k - y^* + \hat{J}_k \delta_k \right\|^2 \quad (3.7)$$

The restriction $\|\delta_k\| < \alpha_k$ ensures that the robot never moves outside a region where the current model approximation cannot be trusted. The successive value of α_{k+1} is adjusted depending on the model agreement $d_k = \frac{\|\Delta y_{\text{measured}}\|}{\|\hat{J}\delta\|}$ as

$$\alpha_{k+1} = \begin{cases} \frac{1}{2}\alpha_k & \text{if } d_k \leq d_{lower} \\ \alpha_k & \text{if } d_{lower} < d_k \leq d_{upper} \\ \max(2\|\delta\|, \alpha) & \text{if } d_k > d_{upper} \end{cases}$$

Extensive experiments are performed with three, six, and twelve DOF robots to verify that this estimation can improve stability and convergence of the robot controllers.

These proposed quasi-Newton methods only address static target tracking using a stationary camera. Consequently, tracking convergence for a moving target is not guaranteed unless the appropriate derivatives are included [60]. As a result, Piepmeyer et al. [57, 59, 60] propose a dynamic Broyden's method to update \hat{J}_k as part of a quasi-Newton method for moving target tracking. They show that the algorithm converges asymptotically. The contributions of this novel method over the aforementioned algorithms are summarized as

1. Development of an uncalibrated visual servoing algorithm for moving target tracking that applies to camera systems that arbitrarily move and includes the eye-to-hand and eye-in-hand cases

- Derivation of the dynamic Broyden's update with a recursive least-squares technique as a part of the dynamic quasi-Newton's method that solve nonlinear visual servoing problems

3.3 Dynamic Quasi-Newton Method via Recursive Least Squares Estimation

3.3.1 Theoretical Fundamental

For moving-target tracking problems, the objective function $F(\theta, t)$ in (3.3) is generally a function of the robot joint angles θ and time t which are assumed independent. This section first considers the eye-to-hand case where the camera is stationary, then extend it to the eye-in-hand case with a moving camera.

The Newton method is derived by expanding (3.3) in a Taylor series about (θ, t) ,

$$F(\theta + h_\theta, t + h_t) = F(\theta, t) + F_\theta h_\theta + F_t h_t + O(h_\theta^2) \quad (3.8)$$

where F_θ and F_t are partial derivatives of F with respect to θ and t while h_θ and h_t are increments of θ and t respectively. At a given sampling period h_t , the function F is minimized by solving

$$\begin{aligned} 0 &= \frac{\partial F(\theta + h_\theta, t + h_t)}{\partial \theta} \\ 0 &= F_\theta + F_{\theta\theta} h_\theta + F_{\theta t} h_t + O(h_\theta^2) \end{aligned} \quad (3.9)$$

Dropping the higher order term $O(h_\theta^2)$ and substituting in $h_\theta = \theta_{k+1} - \theta_k$ yields Newton's method,

$$\theta_{k+1} = \theta_k - (F_{\theta\theta})^{-1}(F_\theta + F_{\theta t} h_t) \quad (3.10)$$

Equation (3.10) is referred as the *dynamic Newton's method* [57] due to the inclusion of the term $F_{\theta t} h_t$.

Expanding (3.10) in terms of $f(\theta, t)$ gives

$$F_\theta = J_k^T f_k \quad (3.11a)$$

$$F_{\theta\theta} = J_k^T J_k + S_k \quad (3.11b)$$

$$F_{\theta t} = J_k^T \frac{\partial f_k}{\partial t} \quad (3.11c)$$

$$S_k = \frac{\partial J_k^T}{\partial \theta} f_k \quad (3.11d)$$

$$J_k = \frac{\partial f_k}{\partial \theta} \quad (3.11e)$$

So (3.10) becomes,

$$\theta_{k+1} = \theta_k - (J_k^T J_k + S_k)^{-1} J_k^T (f_k + \frac{\partial f_k}{\partial t} h_t) \quad (3.12)$$

S_k is a second-order term in the second derivative of F and is known as the *residual*. There is a significant distinction between *zero-residual*, *small-residual*, and *large-residual* cases for nonlinear least-squares problems [19]. For example, in data-fitting application if x^* fits the model $m(x^*)$ of the measured data exactly then $S(x^*)$ is zero and this problem is called the *zero-residual* problem. However, in nonlinear optimization the residual S can be significant and often analytically unavailable or computationally expensive. So nonlinear least-squares algorithms typically assume zero- or small-residual condition. In those cases S is ignored so (3.12) reduces to

$$\theta_{k+1} = \theta_k - (J_k^T J_k)^{-1} J_k^T \left(f_k + \frac{\partial f_k}{\partial t} h_t \right) \quad (3.13)$$

which is known as the *Gauss-Newton method*. Note that $(J_k^T J_k)^{-1} J_k^T$ is the overconstrained pseudo-inverse of the full column rank matrix J_k . Since the range space of J_k is homogeneous in the units of pixels there are no problems of units noninvariance.

In order to eliminate the need for an analytical model to calculate J_k , [57, 59, 60] propose an explicit time dependent algorithm that modifies Broyden's rank one method in (3.6),

$$\hat{J}_k = \hat{J}_{k-1} + \frac{\left(\Delta f - \hat{J}_{k-1} h_\theta - \frac{\partial f_k(t)}{\partial t} h_t \right) h_\theta^T}{h_\theta^T h_\theta} \quad (3.14)$$

Pseudo-code: Dynamic Broyden's Method with RLS estimation (DBM-RLS)

Given: $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$, $P_0 \in \mathbb{R}^{n \times n}$, $\lambda \in (0, 1)$

Initialize: J_0 , θ_0 , θ_1 , and P_0

for $k = 1, \dots$ **do**

$$\begin{aligned}\Delta f &= f_k - f_{k-1} \\ h_\theta &= \theta_k - \theta_{k-1} \\ \hat{J}_k &= \hat{J}_{k-1} + (\lambda + h_\theta^T P_{k-1} h_\theta)^{-1} \left(\Delta f - \hat{J}_{k-1} h_\theta - \frac{\partial f_k(t)}{\partial t} h_t \right) h_\theta^T P_{k-1} \\ P_k &= \frac{1}{\lambda} \left(P_{k-1} - (\lambda + h_\theta^T P_{k-1} h_\theta)^{-1} (P_{k-1} h_\theta h_\theta^T P_{k-1}) \right) \\ \theta_{k+1} &= \theta_k - (\hat{J}_k^T \hat{J}_k)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)\end{aligned}$$

end for

Figure 3.3: A pseudo-code for the DBM-RLS

This method is called the *dynamic Broyden's method* and using it in (3.13) yields the *dynamic quasi Gauss-Newton method*,

$$\theta_{k+1} = \theta_k - (\hat{J}_k^T \hat{J}_k)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k}{\partial t} h_t \right) \quad (3.15)$$

The significant improvement of this method over the previous work is due to the dynamic time-dependent term $\frac{\partial f_k(t)}{\partial t} h_t$ in equations (3.14) and (3.15). In effect f_k is substituted by $f_k + \frac{\partial f_k(t)}{\partial t} h_t$ and that offers a better estimation of the next error term f_{k+1} which is used to compute the update θ_{k+1} .

The Broyden estimator only uses the most recent data but this can be adapted for exponentially deweighted data in a manner similar to (3.4) with a forgetting factor λ . The result is the *dynamic Broyden's method using recursive least square (RLS) estimation* or DBM-RLS and is summarized by the pseudo-code in Figure 3.3.

The proof showing convergence in the neighborhood of a moving target is presented in [60]. Furthermore, it is also shown that the standard Newton's method is not guaranteed convergence for a moving target.

To implement the algorithm it is necessary to approximate $\frac{\partial f_k}{\partial t}$ for which the eye-to-hand system it can be determined as a simple first order difference,

$$\frac{\partial f_k}{\partial t} = \frac{\partial (y(\theta) - y^*(t))}{\partial t} \Big|_k = -\frac{\partial y^*(t)}{\partial t} \Big|_k \simeq -\frac{y_k^* - y_{k-1}^*}{h_t} \quad (3.16)$$

For the eye-in-hand system a similar simple estimation is not possible because the target features vector y^* is now a function of both the robot joint angles θ and time t . y^* may change due to either robot movement, target motion, or a combination of both. Instead [59] introduces the *dynamic Gauss-Newton algorithm with partitioned Broyden's method* or DGN-PBM to simultaneously estimate \hat{J}_k and $\frac{\partial f_k}{\partial t}$. This is summarized by the pseudo-code in Figure 3.4 where the partitioned Jacobian $\tilde{J}_k = \begin{bmatrix} \hat{J}_k & (\hat{f}_t)_k \end{bmatrix}$ has $(\hat{f}_t)_k$ is approximating $\frac{\partial f_k}{\partial t}$.

3.3.2 Simulation and Experimental Results

3.3.2.1 The DBM-RLS Algorithm

The dynamic Broyden's method and the DBM-RLS algorithm are evaluated by simulations using the one, two, and six DOF robotic systems with a stationary camera system in [57]. From these results the dynamic Broyden's method demonstrates superior tracking performance over the static Broyden's method, but its tracking ability is degraded with the presence of noise. However, the DBM-RLS algorithm demonstrates a significant improvement to provide stable and convergent tracking even in the presence of system noise.

The stability and convergence of the DBM-RLS algorithm is also analyzed experimentally using a two-DOF planar robot to track a moving target in [57]. Two types of target trajectories are performed: a synthetic and a real target trajectory. For synthetic target tracking a computer generates an elliptical target path in the image

Pseudo-code: Dynamic Gauss-Newton Algorithm with Partitioned Broyden's Method (DGN-PBM)

Given: $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$; $(\hat{f}_t)_0 \in \mathbb{R}^{m \times 1}$; $P_0 \in \mathbb{R}^{n+1 \times n+1}$; $\lambda \in (0, 1)$

Initialize: $J_0, \theta_0, \theta_1, (\hat{f}_t)_0$, and P_0

for $k = 1, \dots$ **do**

$$\begin{aligned}\Delta f &= f_k - f_{k-1} \\ h_\theta &= \theta_k - \theta_{k-1} \\ h_t &= t_k - t_{k-1} \\ \tilde{h} &= \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix} \\ \tilde{J}_{k-1} &= \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix} \\ \tilde{J}_k &= \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1} \\ \tilde{P}_k &= \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right) \\ \theta_{k+1} &= \theta_k - \left(\hat{J}_k^T \hat{J}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)\end{aligned}$$

end for

Figure 3.4: A pseudo-code for the DGN-PBM method

plane, while for a real target tracking the target motion is generated by rotating a arm, attached with a 10 mm white disc at its end, in a circular arc.

Synthetic Target

The results show that the steady-state tracking can be achieved with a variety of target speeds ranging from $\frac{0.05}{h_t}$ to $\frac{0.25}{h_t}$ where h_t is a sampling period. However its performance varies with target speeds. The tracking errors exponentially increase as target speed increases but they are bounded within a relatively small range and the EE still follows the target motion. Further, increased noise in target motion increases

the tracking errors but convergence or stability is not affected in this range of target speeds.

The performance comparison between the DBM-RLS and the calibrated visual servoing method is also studied for the same target motion. While the calibrated method presents some overshoot, the DBM-RLS method produces somewhat better tracking. Moreover, the DBM-RLS method yields a more robust algorithm when the camera or the robot model is reconfigured. In this case, the calibrated method is completely degraded and fails to converge after the camera is moved. The DBM-RLS performance, on the other hand, is not affected by the reconfiguration and presents no loss of tracking accuracy.

Actual Target

With the same setup and algorithm used in the synthetic target case, the robot EE is controlled to follow a 10 mm white disc attached at the end of a arm rotating in a circular arc. The DBM-RLS method effectively provides a stable and convergent tracking for a moving target, though its performance is decreased due to presence of noise in image processing and the imprecise target motion. In addition, effects of varying the forgetting factor λ is briefly investigated. An optimal value of λ evidently exists (the detail of this matter is discussed in Section 3.4).

3.3.2.2 The DGN-PBM Algorithm

In [59] the simulation results of the DGN-PBM method using a six DOF robot with an eye-in-hand camera system are presented. Different controllers used to calculate the updated θ_{k+1} and methods used to approximate the time-dependent $(\hat{f}_t)_k$ are investigated. Among the combination of controllers and the $(\hat{f}_t)_k$ approximation, the algorithm presented in the pseudo-code in Figure 3.4 yields the best results by offering the least numbers of trials when the system lost track of the target and the control becomes unstable.

Experimental results implementing the DGN-PBM method into an eye-in-hand, planar two-DOF robot was presented in [58]. The DGN-PBM method illustrates stable convergent tracking behavior even though the orientation of the camera or the kinematic model of the robot is significantly changed.

3.3.2.3 Conclusion

From simulation and experimental results these methods robustly provide stable and convergent tracking with a variety of robot DOF, target motions, and speeds. Moreover, these uncalibrated visual servoing schemes significantly improve tolerance to robot and camera configuration changes. Hence, they are independent from analytical models, calibration, and parameter tuning and can be efficiently applied to any type of manipulator and camera systems. However, these algorithms present a number of disadvantages that still needed to be addressed. Section 3.4 discusses some of these challenges.

3.4 Limitations

Three major challenges existing in the DBM-RLS algorithms either with or without partitioning for Broyden’s method are discussed in this study:

1. Large initial error for a dynamic target tracking
2. Selection of an optimal forgetting factor λ
3. A singular or ill-conditioned Hessian matrix $F_{\theta\theta} = J_k^T J_k + S_k$

3.4.1 Large Initial Error

Even though the above algorithms exhibit robust uncalibrated visual servoing schemes in both eye-to-hand and eye-in-hand systems, they are limited to only the zero- or small-residual cases where the Gauss-Newton method is applicable. The initial robot

configuration θ_0 is assumed to be in the neighborhood of the target acquisition configuration θ^* . Convergence is only guaranteed if $F_{\theta\theta}(\theta^*)$ is positive definite, but generally it is not. Hao et al. [30] note that the performance of the algorithms substantially depends on a proper initial estimation of the Jacobian matrix. If the initial choice of Jacobian differs too greatly from the actual Jacobian, the robot may move erratically for a few iterations.

With the small-residual condition, the Gauss-Newton method can be expected to perform well if any of the following conditions hold [49]:

1. The error f is small
2. The function is linear or close to linear

Unfortunately, these assumptions are overly restrictive situations. If the initial error, or a subsequent error, is large then $S = \frac{\partial J^T}{\partial \theta} f$ in (3.12) becomes substantial and an estimation of S is necessary. For the visual servoing problem, the residual error becomes large if the target is significantly far from the initial configuration of the robot or if the target is located outside of the robot workspace. Figure 3.5 shows an initial large-error simulation result using a six DOF robot with an eye-in-hand camera similar to one presented in [59] utilizing the DGN-PBM method (excluding the S approximation). This example demonstrates that the transient tracking trajectory may be uncontrollable before the robot converges to the steady-state tracking if the S approximation is not properly included. Since the Gauss-Newton method becomes less effective on large-residual problems, methods that attempt to improve this problem is investigated in Chapter 5.

3.4.2 Selection of Optimal Forgetting Factor

While a constant value of the forgetting factor λ is utilized in the DBM-RLS and the DGN-PBM algorithms, the optimal performance of these schemes is dependent on its selection [57]. Simulation results show that a lower value of λ improves convergence

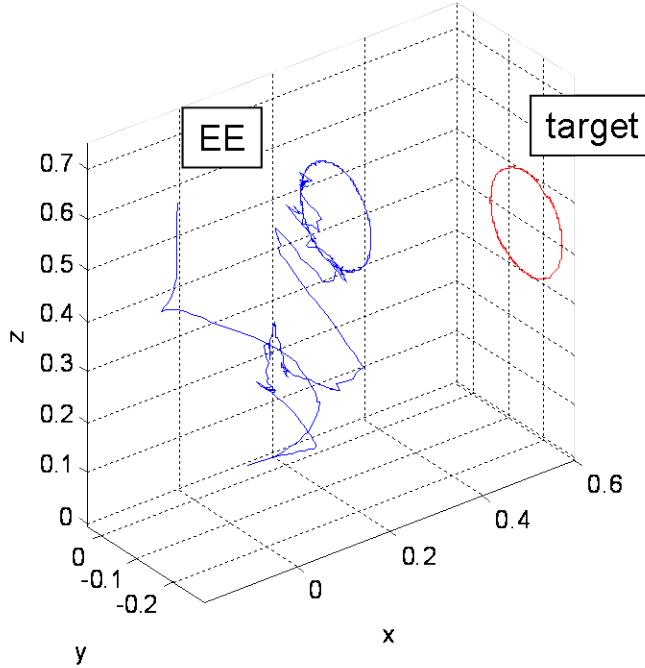


Figure 3.5: A simulation result of a six DOF robot with an eye-in-hand system using the DGN-PBM method when an initial error is significant shown in the task space view.

time but deteriorates steady-state tracking performance. In contrast, a higher λ induces difficulties in acquiring the target, yet steady-state tracking ability is enhanced. A switching scheme that utilizes a lower λ initially then changes to a higher value when the image error norm is below a certain criteria demonstrates some improvement in both transient and steady-state tracking. Simulation results using the same system setup as in the large initial error case show that an inappropriate choice of λ can cause uncontrollable motion before the robot converges to steady-state tracking as shown in Figure 3.6. In some complex trajectory tracking this problem leads to failure in tracking. As a result, an adaptive forgetting factor λ that may lead to a more robust dynamic quasi-Newton method is discussed in Chapter 6.

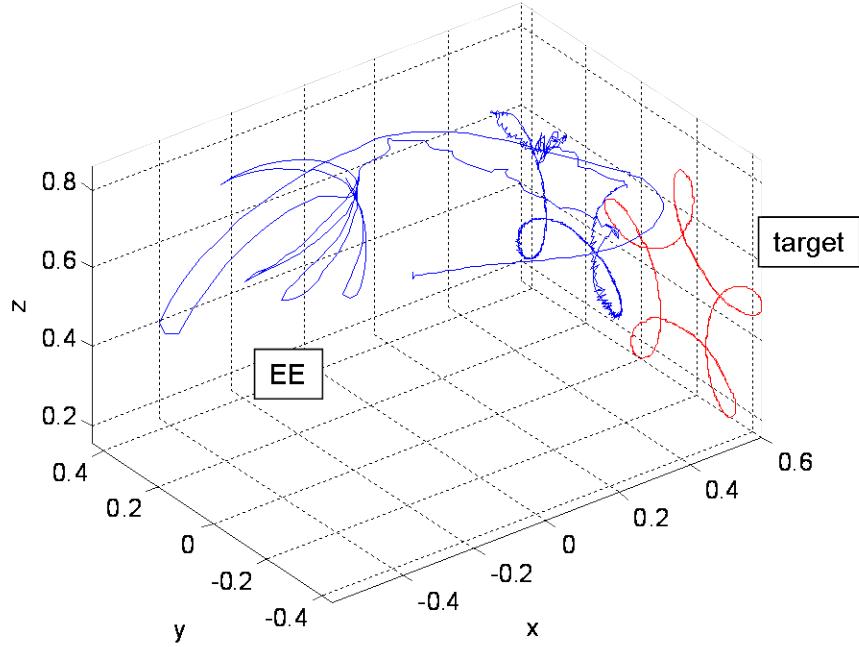


Figure 3.6: A simulation result of an eye-in-hand, six DOF robot using the DGN-PBM method with a constant λ shown in the task space view.

3.4.3 Ill-conditioned Hessian Matrix

Nonlinear optimization problems sometimes yield an ill-conditioned or, in the worst case, a singular approximate Hessian matrix $\hat{H}_k (= \hat{F}_{\theta\theta})$. Since the quasi-Newton method (3.12) requires the inverse of the Hessian matrix, the ill-conditioning usually leads to numerical problems that results in slow or no convergence. A scheme such as the *Levenberg-Marquardt algorithm* [42] that modifies the Hessian matrix to ensure positive definiteness may be implemented to overcome this deficiency. In addition, methods are available in [19] guarantee a positive definite approximate Hessian, even if the actual Hessian is not positive definite. Though this introduces nonphysical artifacts, it may improve effective near singularities. A proper strategy to cope with this limitation is investigated in Chapter 6.

3.5 Summary

The brief reviews of the DBM-RLS algorithms either with or without partitioning for Broyden's method (the DBM-RLS and the DGN-PBM algorithms) developed by Piepmeyer et al. [57, 59, 60] are presented in this chapter. These methods are shown to be effective for moving target tracking with the eye-to-hand and eye-in-hand camera configurations. Depending on a camera system setup, generally the error vector $f(\theta, t)$ between the EE image feature y and the desired target image features y^* as seen on the image plane can be measured,

$$f(\theta, t) = y(\theta) - y^*(t) \quad [3.2]$$

A visual servoing problem is cast as a nonlinear least squares optimization problem where the objective function $F(\theta, t)$ to be minimized is the squared error $f^T(\theta, t) f(\theta, t)$,

$$F(\theta, t) = \frac{1}{2} f^T(\theta, t) f(\theta, t) \quad [3.3]$$

A solution θ^* that minimizes (3.3) can be calculated using the Gauss-Newton method,

$$\theta_{k+1} = \theta_k - (J_k^T J_k)^{-1} J_k^T \left(f_k + \frac{\partial f_k}{\partial t} h_t \right) \quad [3.13]$$

in which it is assumed that the initial Robot configuration θ_1 is in the neighborhood of the desired robot configuration θ^* . Since a priori knowledge of the robot kinematic model and the camera model are assumably unknown, i.e., an analytical model to calculate J_k is unavailable, the Jacobian J_k can be recursively estimated using a dynamic Broyden's method with the recursive least square method as

$$\hat{J}_k = \hat{J}_{k-1} + \frac{\left(\Delta f - \hat{J}_{k-1} h_\theta - \frac{\partial f_k(t)}{\partial t} h_t \right) h_\theta^T}{h_\theta^T h_\theta} \quad [3.14]$$

Substituting an approximation of \hat{J}_k into (3.13) yields the dynamic quasi Gauss-Newton method,

$$\theta_{k+1} = \theta_k - (\hat{J}_k^T \hat{J}_k)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k}{\partial t} h_t \right) \quad [3.15]$$

A significant improvement of the DBM-RLS algorithms over the previous work in [34, 37] is due to the dynamic time-dependent term $\frac{\partial f_k(t)}{\partial t} h_t$ in equations (3.14) and (3.15).

From simulation and experimental results the DBM-RLS algorithms robustly provide stable and convergent tracking with a variety of robot DOF, target motions, and speeds. Furthermore, they improve tolerance to robot and camera configuration changes. However, there exists a number of challenges using these novel uncalibrated visual servoing algorithms:

1. Large initial error for a dynamic target tracking
2. Selection of an optimal forgetting factor λ
3. A singular or ill-conditioned Hessian matrix $F_{\theta\theta} = J_k^T J_k + S_k$

These issues are thoroughly investigated and discussed in Chapter 4, Chapter 5, and Chapter 6.

CHAPTER IV

MODIFIED METHODS FOR THE LARGE-RESIDUAL VISUAL SERVOING PROBLEM

One of the major disadvantages of the DBM-RLS and the DGN-PBM algorithms is that they are limited to only the zero- or small-residual cases since the algorithms exploit the quasi-Gauss-Newton method in which the residual S is neglected. This chapter introduces the theoretical fundamentals of the nonlinear least-squares visual control problem for the large-residual case in which the initial robot configuration θ_0 is not in the neighborhood of the target acquisition configuration θ^* . In this case, S becomes substantial and the inclusion of the residual S can significantly improve the overall performance of the algorithms. Due to difficulties in the analytical computation of the residual S , various methods are used to approximate the residual S .

This chapter starts with Section 4.1 where a brief summary of the related terminologies and fundamentals for solving large-residual visual servoing problems are reviewed. Various algorithms used to solve large-residual cases for optimization are discussed in Section 4.2. Section 4.3 presents previous attempts for solving large-residual cases in uncalibrated visual servoing applications. Due to the fact that only static target tracking has appeared in literature, various algorithms are proposed for solving moving target tracking for large-residual cases and is presented in Section 4.4. In this section a novel algorithm called the *modified BFGS* (MBFGS) method is developed. Then the convergence analysis of the MBFGS algorithm and its convergence rate are discussed in Section 4.5 and Section 4.6 respectively. Section 4.7 discusses a hybrid method known as the *switching modified BFGS-dynamic Broyden*

or *switching MBFGS-DB* algorithm which is a combination of the MBFGS algorithm and the DGN-PBM method. This novel hybrid attains fast convergence with a lowered computational cost for solving large-residual visual servoing problems because it only employs the MBFGS algorithm when the error is greater than a criteria and switches to the DGN-PBM method otherwise. Lastly the summary of this chapter is presented in Section 4.8.

4.1 Introduction

In Chapter 3 the visual servoing problem is cast as nonlinear least-squares optimization problem seeking the solution of

$$\text{Minimize} \quad F : \mathbb{R}^m \rightarrow \mathbb{R} \quad (4.1)$$

in which F is an objective function defined as

$$F(\theta, t) = \frac{1}{2} f^T(\theta, t) f(\theta, t) \quad (4.2)$$

where $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$ is an image plane error vector in pixels between the robot EE feature vector y and the target feature vector y^* .

A solution of the minimization problem (4.2) can be found by the dynamic Newton's method as

$$\theta_{k+1} = \theta_k - (F_{\theta\theta})^{-1}(F_\theta + F_{\theta t} h_t) \quad (4.3)$$

where $F_{\theta\theta}$ is also known as the Hessian matrix H_k

Expanding (4.3) in terms of $f(\theta, t)$ gives

$$F_\theta = J_k^T f_k \quad (4.4a)$$

$$F_{\theta\theta} = J_k^T J_k + S_k \quad (4.4b)$$

$$F_{\theta t} = J_k^T \frac{\partial f_k}{\partial t} \quad (4.4c)$$

$$S_k = \frac{\partial J_k^T}{\partial \theta} f_k \quad (4.4d)$$

$$J_k = \frac{\partial f_k}{\partial \theta} \quad (4.4e)$$

Equation (4.3) becomes,

$$\theta_{k+1} = \theta_k - (J_k^T J_k + S_k)^{-1} J_k^T (f_k + \frac{\partial f_k}{\partial t} h_t) \quad (4.5)$$

where S_k is known as the residual and the Hessian matrix H_k is

$$H_k = J_k^T J_k + S_k \quad (4.6)$$

Substitution into (4.5) yields,

$$\theta_{k+1} = \theta_k - H_k^{-1} J_k^T (f_k + \frac{\partial f_k}{\partial t} h_t) \quad (4.7)$$

This is known as the *full dynamic Newton method* due to the inclusion of the residual S_k in the Hessian H_k .

The algorithms presented in Chapter 3 implement a quasi-Gauss-Newton method in which the residual S_k is neglected and (4.6) reduces to $H_k = J_k^T J_k$. These algorithms robustly provide stable and convergent tracking if the initial robot configuration θ_0 is close to the target acquisition configuration θ^* , i.e., the zero- or small-residual case. If the residual S_k becomes significant, the nonlinear least-squares problem (4.1) is called the *large-residual* problem. Even though the LMA can generally be used to solve for a solution of the large-residual problem, the rate of convergence can be unacceptably slow or sometimes fail to converge [67].

In order to improve this problem various algorithms are introduced to approximate the Hessian H_k and can be categorized into two classes:

1. Approximation of the whole Hessian H_k giving \hat{H}_k
2. Approximation of only the residual S_k giving \hat{S}_k

4.2 *Approximation of the Hessian for Large-Residual Problems Background*

The original algorithms developed for the large-residual problems are introduced for general nonlinear optimization using Newton's method or a quasi-Newton method

that do not include the dynamic time-dependent term $\frac{\partial f_k}{\partial t} h_t$ as in (4.7). For such a case (4.7) becomes

$$\theta_{k+1} = \theta_k - H_k^{-1} J_k^T f_k \quad (4.8)$$

The algorithms reviewed in this section are developed using Newton's method (4.8) for the large-residual problems in general nonlinear least squares optimization. To improve uncalibrated visual servoing algorithms (the DBM-RLS and the DGN-PBM algorithms in Chapter 3) a modified BFGS method for approximating the residual S_k is developed in Section 4.4. In that case the developed algorithm employs the dynamic Newton's method (4.7) to efficiently deal with the error vector f that is a function of two independent variables, robot joint angles θ and time t .

4.2.1 Approximation of the Whole Hessian

The approximation of the Hessian H_k (4.6) can be done using secant techniques similar to those in Section 2.2.2 in Chapter 2. In this context, the Hessian H_k is approximated as \hat{H} such that it satisfies

$$\hat{H}_k h_{k-1} = g_{k-1} \quad (4.9)$$

where

$$h_{k-1} = \theta_k - \theta_{k-1} \quad (4.10)$$

$$\begin{aligned} g_{k-1} &= \nabla F_k - \nabla F_{k-1} \\ &= J_k^T f_k - J_{k-1}^T f_{k-1} \end{aligned} \quad (4.11)$$

Then \hat{H}_k can be updated from \hat{H}_{k-1} using rank 1 and rank 2 Hessian updates. For example, the rank 1 update [67] is

$$\hat{H}_k = \hat{H}_{k-1} + \frac{(g_{k-1} - \hat{H}_{k-1} h_{k-1}) (g_{k-1} - \hat{H}_{k-1} h_{k-1})^T}{(g_{k-1} - \hat{H}_{k-1} h_{k-1})^T h_{k-1}} \quad (4.12)$$

The two well-known rank 2 Hessian approximations are the BFGS method [19],

$$\hat{H}_k = \hat{H}_{k-1} + \frac{g_{k-1}g_{k-1}^T}{g_{k-1}^T h_{k-1}} - \frac{\hat{H}_{k-1}h_{k-1}h_{k-1}^T\hat{H}_{k-1}}{h_{k-1}^T\hat{H}_{k-1}h_{k-1}} \quad (4.13)$$

and the DFP method [19],

$$\begin{aligned} \hat{H}_k &= \hat{H}_{k-1} + \frac{\left(g_{k-1} - \hat{H}_{k-1}h_{k-1}\right)g_{k-1}^T + g_{k-1}\left(g_{k-1} - \hat{H}_{k-1}h_{k-1}\right)^T}{g_{k-1}^T h_{k-1}} \\ &\quad - \frac{\left(g_{k-1} - \hat{H}_{k-1}h_{k-1}\right)^T h_{k-1}g_{k-1}g_{k-1}^T}{\left(g_{k-1}^T h_{k-1}\right)^2} \end{aligned} \quad (4.14)$$

The approximated Hessian \hat{H}_k is used in the quasi-Newton method to calculate θ_{k+1} ,

$$\hat{H}_k h_k = -J_k^T f_k \quad (4.15)$$

where $h_k = \theta_{k+1} - \theta_k$.

Nazareth [48] introduces a hybrid method that combines a quasi-Newton method and the Gauss-Newton method so that the resultant search direction is a weighted average between the quasi-Newton and the Gauss-Newton directions and is controlled via a weighting parameter ϕ as

$$\left[\phi_k J_k^T J_k + (1 - \phi_k) \hat{H}_k\right] h_k = -J_k^T f_k \quad (4.16)$$

where \hat{H}_k is an approximation of H_k , and $0 \leq \phi_k \leq 1$ is adaptively chosen according to how well the Gauss-Newton model can be trusted. To examine the Gauss-Newton method performance [48] compares the actual reduction in the function F_{k-1} value with the predicted value from a Gauss-Newton model. This is implemented with the LMA so (4.16) becomes

$$\left[\phi_k J_k^T J_k + (1 - \phi_k) \hat{H}_k + \mu_k I\right] h_k = -J_k^T f_k \quad (4.17)$$

A strategy for selecting μ_k is similar to choosing ϕ_k . The Hessian \hat{H}_k is assumed to be positive semi-definite and Davidon's optimally conditioned algorithm [13] is used

for approximating the Hessian \hat{H}_k in [48]. As k increases and ϕ_k gets close to being zero, this method yields superlinear convergence (a property of the Gauss-Newton method). If the initial H_0 is approximated as $J_1^T J_1$ with $\mu_1 = 0$, then for linear problems the algorithm converges in one step.

4.2.2 Approximation of the Residual S

One disadvantage of approximating the whole Hessian \hat{H}_k is that J_k is already available either analytically or from approximation using finite differences so $J_k^T J_k$, which is often the dominant portion of \hat{H}_k in (4.6), is already known. In order to minimize computing costs a number of studies focus on algorithms that only approximate the residual S_k . Since S_k is an $n \times n$ symmetric matrix, similar to the Hessian matrix H_k , an approximation of S_k can be done using secant techniques similar to the Hessian \hat{H}_k approximation as in Section 4.2.1. As a result, \hat{H}_k is used to distinguish this type of the Hessian approximation from the approximation of \hat{H}_k in Section 4.2.1. For the remainder of this development, \hat{H}_k is referred to as an approximation of the whole Hessian H_k , while \hat{H}_k is referred to as an approximation of H_k in which only the residual S_k is approximated,

$$\hat{H}_k = J_k^T J_k + \hat{S}_k \quad (4.18)$$

where \hat{S}_k is an approximation of the residual S_k .

Although in this study only the approximation of Jacobian \hat{J}_k is available, for simplicity of notation presented in this chapter J_k is used to represent general cases.

Methods that adapt the Hessian approximation techniques for estimating S_k including the Broyden-Dennis (BD) method [15], the Betts (B) method [3], and the early version of NL2SOL [16] ([49] referred this method as the DGW method) are discussed in this section.

Recall the secant equation of the Hessian approximation as

$$H_k h_{k-1} = g_{k-1} \quad [4.9]$$

where

$$h_{k-1} = \theta_k - \theta_{k-1} \quad [4.10]$$

and

$$g_{k-1} = J_k^T f_k - J_{k-1}^T f_{k-1} \quad [4.11]$$

Substituting H_k from (4.6) and g_{k-1} from (4.11) into (4.9) gives

$$(J_k^T J_k + S_k) h_{k-1} = J_k^T f_k - J_{k-1}^T f_{k-1} \quad (4.19)$$

or more concisely,

$$S_k h_{k-1} = \gamma_{BD,k} \quad (4.20)$$

where

$$\gamma_{BD,k} = J_k^T f_k - J_{k-1}^T f_{k-1} - J_k^T J_k h_{k-1} \quad (4.21)$$

Equation (4.20) is the secant equation which is similar to (4.9) for the Hessian approximation. As a result, the residual S_k approximation can be done by simply substituting \hat{S} instead of \hat{H} in the formulas such as in (4.12)-(4.14). Differences between the BD method, the B method, and NL2SOL is how $\gamma_{BD,k}$ in (4.20) is defined and what approach is applied to approximate S_k . The form of $\gamma_{BD,k}$ in (4.21) is known as the *Broyden-Dennis method* or the BD method [15]. This approach applies Powell's symmetric rank 2 update (or the PSB method, see Chapter 2) for \hat{S}_k approximation [61].

A slight modification of $\gamma_{BD,k}$ was proposed by Betts [3] that leads to

$$\gamma_{B,k} = J_k^T f_k - J_{k-1}^T f_{k-1} - J_{k-1}^T J_{k-1} h_{k-1} \quad (4.22)$$

in which Betts uses Davidon's symmetric rank 1 to update \hat{S}_k [12].

Lastly Dennis et al. [16] introduced

$$\gamma_{DGW,k} = J_k^T f_k - J_{k-1}^T f_k - J_{k-1}^T J_{k-1} h_{k-1} \quad (4.23)$$

where \hat{S}_k is updated using the Davidon-Fletcher-Powell (DFP) method. Equation (4.23) is used in their nonlinear least-squares code *NL2SOL* of which the later version is presented in [17, 20]. The details of this algorithm are discussed in Section 4.3.2.

Dennis et al. [17] tested the various choices for γ_k ,

- The Broyden-Dennis: $\gamma_{BD,k} = J_k^T f_k - J_{k-1}^T f_{k-1} - J_k^T J_k h_{k-1}$
- The Betts: $\gamma_{B,k} = J_k^T f_k - J_{k-1}^T f_{k-1} - J_{k-1}^T J_{k-1} h_{k-1}$
- The Dennis-Gay-Welsch (NL2SOL): $\gamma_{DGW,k} = J_k^T f_k - J_{k-1}^T f_k - J_{k-1}^T J_{k-1} h_{k-1}$

and concluded that the choice used in NL2SOL gives the best results.

Nazareth [49] tested the B method, the BD method, the early version of NL2SOL [16], a full quasi-Newton method (DQN), the LMA, and his own hybrid algorithm (N) (4.17). All methods were implemented with a trust region algorithm so differences between these methods are only confined to the different ways in approximating the Hessian \hat{H}_k . He tested six zero-residual and five large-residual problems and measured the numbers of function and Jacobian evaluations for efficiency comparison between these methods.

Nazareth concluded that algorithms B, BD, and NL2SOL perform well in the large-residual problems, but the DGW method is the clear winner. For the zero-residual cases the LMA yields the most efficient method except for one function that the hybrid algorithm (N) did better than the LMA.

Seber and Wild [67] summarize their observations from Nazareth's results into 3 themes:

1. For zero-residual problems the Hessian approximation using a Gauss-Newton method with the LMA implementation seems to be best

2. For large-residual problems the DGW method appears to be best
3. Better overall performance of these algorithms may be improved through some forms of hybridization

The other interesting observation from Nazareth's result is that for a zero-residual problem where the solution of a function is badly scaled, NL2SOL (the DGW method) requires less than half the numbers of function and Jacobian evaluations compared to the LMA or the N method. Although this observation was not mentioned in either [49] or [67], it may lead to a potential improvement in the case of an ill-conditioned or badly-scaled Hessian approximation which is further discussed in Chapter 6.

4.2.3 The Gill-Murray Method

The Gill-Murray (GM) method [26] qualitatively differs from the above methods. This method is motivated from the fact that any vector direction p can be written in a form $p = p_1 + p_2$ where p_1 lies in the range space of J_k and p_2 lies in the null space of J_k^T , where $\text{rank}(J_k) = q < n$. Similarly, the Newton step h_k can be expressed as $h_k = h_{1,k} + h_{2,k}$ where $h_{1,k}$ is in the range space of $J_k^T J_k$ with dimension q and $h_{2,k}$ is in the null space of $J_k^T J_k$ with dimension $n - q$. As a result, it is believed that the Gauss-Newton method is deficient because it only includes the component in the range space of $J_k^T J_k$ but excludes the component in the null space of $J_k^T J_k$.

In [26] the GM method calculates h_k as a sum of a set of q eigenvectors T_1 corresponding to the first q largest dominant eigenvalues of $J_k^T J_k$ and a set of eigenvalues for its complementary space T_2 with dimension $n - q$ corresponding to the lesser eigenvalues of $J_k^T J_k$. The parameter q is called the *grade* and is needed to be selected to properly determine the size of the dominant and the complementary sets of T_1 and T_2 . The Newton step becomes

$$h_{C,k} = T_1 \tilde{c}_1 + T_2 \tilde{c}_2 \quad (4.24)$$

which is known as the *corrected* Gauss-Newton step [67]. In this case $T_1\tilde{c}_1$ is $h_{1,k}$ and $T_2\tilde{c}_2$ is $h_{2,k}$ where \tilde{c}_1 and \tilde{c}_2 are vectors that have dimension q and $n - q$ respectively. Three alternatives are detailed in [26] for solving (4.24):

1. Explicit second derivatives
2. Finite-difference approximation
3. Quasi-Newton approximation

The corrected Gauss-Newton step is the sum of a *Gauss-Newton step* which is spanned by the q -dimensional eigenvectors in T_1 corresponding to the q largest eigenvalues of $J_k^T J_k$ and a *Newton step* which is spanned by the $n - q$ eigenvectors in T_2 corresponding to the smallest $n - q$ eigenvalues of $J_k^T J_k$. If J_k has full rank n , then the corrected step $h_{C,k}$ (4.24) is the Gauss-Newton step. On the contrary, if the current corrected step $h_{C,k}$ yields poor progression, then more eigenvalues are moved into the lesser eigenvalue set by decreasing q . Thus, the corrected Newton step $h_{C,k}$ behaves more like or becomes the Newton step in the case that $q = 0$. As a result, the GM method offers an interpolation between the Gauss-Newton step ($q = n$) and the Newton step ($q = 0$) [67].

Nazareth [49] did not include the GM method in his study so no comparative performance between the DWG method (or NL2SOL) and the GM method has been studied. However, it is recommended in [67] that a combination between NL2SOL and the GM method should be empirically developed.

4.3 Review of Large-Residual Visual Servoing Problems

For visual servoing problems, the residual error S_k is crucial when the target is significantly far from the initial configuration of the robot or the target is located outside the robot workspace. However, only a few studies have been done for large-residual cases in visual servoing problems. This section discusses two main studies that have

been done by Kim et al. [40, 38, 39] and Fu et al. [25]. For simplicity the original terminologies and notations are adapted to be consistent with those defined in Section 4.1.

4.3.1 Kim et al. Studies

Kim and Lee [40] propose an uncalibrated visual servoing algorithm using the full Newton's method and the secant approximation to calculate robot joint angles for the large-residual case. The objective function in this context is slightly different from (4.2) because it is only used for static target tracking with a stationary camera system. Hence, the objective function $F(\theta)$ is

$$F(\theta) = \frac{1}{2} f^T(\theta) f(\theta) \quad (4.25)$$

where

$$f(\theta) = y(\theta) - y^* \quad (4.26)$$

and y^* is constant. Similar to (4.2) $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$ is an error vector between a robot EE feature vector y and a static target image feature vector y^* in unit pixels as seen on the image plane except that now the objective function F is only a function of robot joint angles θ . As a result, the full Newton's method for this application does not use the dynamic time-dependent term $\frac{\partial f_k}{\partial t} h_t$ and the next robot joint angles can be found using (4.8) as

$$\theta_{k+1} = \theta_k - H_k^{-1} J_k^T f_k \quad [4.8]$$

Kim and Lee [40] applied a secant technique to the residual term S_k as

$$S_k h_{k-1} = (J_k - J_{k-1})^T f_k \quad (4.27)$$

where $h_{k-1} = \theta_k - \theta_{k-1}$. The approximation of S_k is given in [40] as

$$S_k = \frac{(J_k - J_{k-1})^T f_k h_{k-1}^T}{h_{k-1}^T h_{k-1}} \quad (4.28)$$

Then the calculated S_k is used in (4.8) to solve for the successive robot joint angles θ_{k+1} .

The method in (4.28) using a secant technique for the residual S_k update is implemented in conjunction with a Jacobian approximation similar to the one presented in the DBM-RLS algorithm [57]. Since this algorithm is developed for static target tracking the Jacobian approximation does not include the time-varying term $\frac{\partial f_k}{\partial t} h_t$,

$$\hat{J}_k = \hat{J}_{k-1} + (\lambda + h_{k-1}^T P_{k-1} h_{k-1})^{-1} (\Delta f - \hat{J}_{k-1} h_{k-1}) h_{k-1}^T P_{k-1} \quad (4.29)$$

$$P_k = \frac{1}{\lambda} \left(P_{k-1} - (\lambda + h_{k-1}^T P_{k-1} h_{k-1})^{-1} (P_{k-1} h_{k-1} h_{k-1}^T P_{k-1}) \right) \quad (4.30)$$

Both simulation and experimental results are performed in [39] to validate the performance of the algorithm using the S_k update in (4.28) with a non-recursive Jacobian update for tracking a moving target with the eye-to-hand configuration. Experiments are performed to validate the efficiency of this algorithm against a model-based visual servoing algorithm. This algorithm reduces the average error norm in comparison to the model-based visual servoing.

4.3.2 Fu et al. Studies

Fu et al. [25] also use a full Newton's method with a Jacobian approximation for solving large-residual visual servoing problems. They develop an uncalibrated visual servoing algorithm for static target tracking in an eye-to-hand configuration similar to Kim and Lee's work [40, 38, 39]. In this context, the objective function $F(\theta)$ and the error vector $f(\theta)$ are the same as in (4.25) and in (4.26) respectively.

The full Newton's method (4.8) is also employed for determining θ_{k+1} ,

$$\theta_{k+1} = \theta_k - H_k^{-1} J_k^T f_k \quad [4.8]$$

where $H_k = J_k^T J_k + S_k$.

In this work the recursive Jacobian approximation (4.29) and (4.30) are also utilized for the Jacobian estimation to achieve model-free static target tracking.

The three major contributions presented in [25] that differs from Kim and Lee in [40, 38, 39] are:

1. A different method is used to approximate the residual S_k
2. A trust region method is implemented to improve the algorithm stability
3. Different camera configurations are studied to improve the algorithm performance

1. The approximation of the residual S_k

In order to approximate the residual S_k a secant technique used in NL2SOL [17, 20] is employed. First the residual S is re-introduced in a different form as

$$S = \sum_{i=1}^n f_i \nabla^2 f_i$$

where $\nabla^2 f_i$ is a third order tensor. Dennis et al. [17, 20] introduce a secant approximation to the residual S_{k+1} as

$$\begin{aligned} S_{k+1} h_k &= \sum f_i(\theta_{k+1}) \nabla^2 f_i(\theta_{k+1}) h_k \\ &\cong \sum f_i(\theta_{k+1}) (\nabla f_i(\theta_{k+1}) - \nabla f_i(\theta_k)) \end{aligned}$$

Recall that $\nabla f_i(\theta_{k+1}) = J_{k+1}^T$ so

$$S_{k+1} h_k \cong J_{k+1}^T f_{k+1} - J_k^T f_{k+1} \equiv z_k^* \quad (4.31)$$

where $h_k = \theta_{k+1} - \theta_k$.

To update S_{k+1} from S_k and satisfy (4.31), Dennis et al. [17, 20] employ an algorithm that slightly differs from the DFP method (a very well-known method for Hessian approximation),

$$\hat{S}_{k+1} = \hat{S}_k + \frac{(z_k^* - \hat{S}_k h_k) g_k^T + g_k (z_k^* - \hat{S}_k h_k)^T}{g_k^T h_k} - \frac{h_k^T (z_k^* - \hat{S}_k h_k) g_k g_k^T}{(g_k^T h_k)^2} \quad (4.32)$$

where $g_k = \nabla F_{k+1} - \nabla F_k = J_{k+1}^T f_{k+1} - J_k^T f_k$. The difference between g_k and z_k^* is

$$\begin{aligned} g_k - z_k^* &= (J_{k+1}^T f_{k+1} - J_k^T f_k) - (J_{k+1}^T f_{k+1} - J_k^T f_{k+1}) \\ &= J_k^T (f_{k+1} - f_k) \end{aligned}$$

2. Trust region method

Fu et al. [25] embed a trust region method presented in [23] into their algorithm to guarantee global convergence. In this case the quadratic model q_k is used as a suborder approximation of the objective function $F(\theta)$,

$$q_{k+1} = \frac{1}{2} f_k^T f_k + (J_k^T f_k)^T (\theta - \theta_k) + \frac{1}{2} (\theta - \theta_k)^T (J_k^T J_k + S_k) (\theta - \theta_k) \quad (4.33)$$

The least-squares solution of the objective function $F(\theta)$ is now obtained by solving a subproblem of the following constrained equation,

$$\min q_k = \frac{1}{2} f_k^T f_k + (J_k^T f_k)^T (\theta - \theta_k) + \frac{1}{2} (\theta - \theta_k)^T (J_k^T J_k + S_k) (\theta - \theta_k) \quad (4.34)$$

such that

$$\|\theta - \theta_k\| \leq \delta_k$$

where δ_k determines the size of the trust region.

To solve (4.34) they employ the LMA yielding the modified Newton method,

$$\theta_{k+1} = \theta_k - (J_k^T J_k + S_k + \alpha_k D_k^2)^{-1} J_k^T f_k \quad (4.35)$$

where α_k is the Levenberg-Marquardt parameter or the damping parameter and D_k is a diagonal matrix. Both α_k and D_k are updated using Moré's approach [47].

The size of the trust region δ_k is calculated according to how well the model q_k approximates the objective function F_k . To evaluate the performance of the current model q_k , the ratio r between the actual reduction of the function and the predicted value obtained from the model q_k is computed. The actual reduction of the objective function is

$$\Delta F = F_{k-1} - F_k \quad (4.36)$$

while, the predicted reduction is

$$\Delta q = F_k - q_{k+1} \quad (4.37)$$

and the ratio r is

$$r = \frac{\Delta F}{\Delta q} \quad (4.38)$$

The value of r determines whether the trust region size δ_k needs to be adjusted. The closer r is to unity, the better the approximation of F_k using the current model q_k .

Fu et al. follows Fletcher's conditions [23] for updating δ_k ,

$$\delta_{k+1} = \begin{cases} 0.5\|D_k h_k\|_2 & \text{if } r \leq 0.25 \\ 2\|D_k h_k\|_2 & \text{if } r \geq 0.75 \\ \|D_k h_k\|_2 & \text{if otherwise} \end{cases}$$

3. Effects of camera configuration in improving precision tracking

Two stationary cameras are used in [25] and camera arrangements are studied to improve precision in robot motion control. Two cases of camera configurations are discussed: 1) the two cameras views (optical axes) are nearly parallel and, 2) the two cameras are arranged such that the optical axes are perpendicular to each other. When the cameras are arranged as in case 1), both cameras may become insensitive to the target motion parallel to the optical axes. In contrast, the second case significantly improves upon this problem.

A summary of the required steps implementing the Fu et al. algorithm [25] is shown in the pseudo-code in Figure 4.1.

Simulations using a spatial RRR robot to track a static target in an eye-to-hand configuration are performed to validate the efficiency of the algorithm in comparison with the Gauss Newton method in which a Jacobian estimation is done using (4.29) and (4.30). Each algorithm is used to calculate θ_{k+1} for both small- and large-residual cases. From these results, their algorithm outperforms the latter algorithm in both

Pseudo-code: Summary of the algorithm in [25] for the large-residual visual servoing problem

Given: $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$, $P_0 \in \mathbb{R}^{n \times n}$, $\lambda \in (0, 1)$

Initialize: \hat{J}_0 , θ_0 , θ_1 , , y_0 , y^* , $S_0 = 0$, and P_0

for $k = 0, \dots$ **do**

Calculate H_k

$$H_k = J_k^T J_k + S_k$$

Compute θ_{k+1}

$$\theta_{k+1} = \theta_k - (J_k^T J_k + S_k + \alpha_k D_k^2)^{-1} J_k^T f_k$$

Update Jacobian J_k

$$\Delta f = f_k - f_{k-1}, \quad h_{k-1} = \theta_k - \theta_{k-1}$$

$$\hat{J}_k = \hat{J}_{k-1} + (\lambda + h_{k-1}^T P_{k-1} h_{k-1})^{-1} (\Delta f - \hat{J}_{k-1} h_{k-1}) h_{k-1}^T P_{k-1}$$

$$P_k = \frac{1}{\lambda} \left(P_{k-1} - (\lambda + h_{k-1}^T P_{k-1} h_{k-1})^{-1} (P_{k-1} h_{k-1} h_{k-1}^T P_{k-1}) \right)$$

Compute the ratio r

$$r = \frac{\Delta F}{\Delta q}$$

Update trust region size δ_k

$$\delta_{k+1} = \begin{cases} 0.5 \|D_k h_k\|_2 & \text{if } r \leq 0.25 \\ 2 \|D_k h_k\|_2 & \text{if } r \geq 0.75 \\ \|D_k h_k\|_2 & \text{if otherwise} \end{cases}$$

Update the residual S_k

$$\hat{S}_{k+1} = \hat{S}_k + \frac{(z_k^* - \hat{S}_k h_k) g_k^T + g_k (z_k^* - \hat{S}_k h_k)^T}{g_k^T h_k} - \frac{h_k^T (z_k^* - \hat{S}_k h_k) g_k g_k^T}{(g_k^T h_k)^2}$$

where

$$\begin{aligned} z_k^* &= \hat{J}_{k+1}^T f_{k+1} - \hat{J}_k^T f_{k+1} \\ g_k &= \hat{J}_{k+1}^T f_{k+1} - \hat{J}_k^T f_k \end{aligned}$$

end for

Figure 4.1: Pseudo-code for the algorithm in [25]

cases. For the large-residual case, the algorithm presented by Fu et al. quickly converges to the desired robot configuration θ^* while the other method fails to converge. In addition, the average error is substantially reduced when the two cameras are arranged perpendicularly to each other.

4.4 Modified Hessian Approximations for the Large-Residual Visual Servoing Problem

Both previous studies are mainly developed for static target tracking in an eye-to-hand configuration. An algorithm for moving target tracking in an eye-in-hand configuration for the large-residual cases does not seem to have appeared in the literature. Although Kim et al. [39] briefly presented moving target tracking in an eye-to-hand configuration the theoretical development was not given. As a result, this section is devoted to a development of a novel algorithm for solving the large-residual visual servoing problems without a priori requirement of a kinematic robot model or a camera calibration.

This development is originally inspired by the work of Fu et al. [25]. An attempt to implement Fu's algorithm for a moving target tracking has been made. For simplicity, the eye-to-hand camera configurations with a RRR robot presented in [25] are initially used to simulate both static and moving target tracking. For static target tracking similar simulation results presented in [25] are obtained. However, for the moving target tracking case the performance of this algorithm deteriorates and often diverges. It is worth mentioning that Fu's algorithm embeds three different issues, i.e., estimating S_k , a trust region method, and the LMA, of which imperfect implementation of any issue may affect the overall performance of the algorithm. In addition, the residual S_k approximation in [25] follows NL2SOL [17, 20] which is a complex FORTRAN code containing 2360 lines excluding comments. Furthermore, the algorithm for updating the damping parameter α_k and the diagonal matrix D_k might yield a size of the trust region δ_k that satisfies (4.34) but is still too large

for small-approximation Jacobian based control in robotic applications. Therefore, a novel algorithm that offers improved stability for moving target tracking with less complexity and calculation costs is developed in this section.

Since the ideas of implementing a trust-region method and the LMA in [25] demonstrates a potential to improve the efficiency and stability of the algorithm, the LMA similar to those used in [25] are also parts of this novel algorithm and they are discussed in more detail in Chapter 6.

In this section three distinct algorithms are proposed for the Hessian H_k approximation:

1. Approximation of the whole Hessian H_k using the *DBFGS* update
2. Approximation of the residual S_k
 - (a) Using the BFGS update
 - (b) Using the *modified BFGS* update

4.4.1 Approximation of the whole Hessian H_k using the *DBFGS* update

For the most general case of visual servoing problems the objective function $F(\theta, t)$ is defined as in (4.2) since it applies to the special cases, i.e., an eye-to-hand camera configuration, with small modifications. From

$$F(\theta, t) = \frac{1}{2} f^T(\theta, t) f(\theta, t) \quad [4.2]$$

and expanding (4.2) in a Taylor series about (θ, t) gives

$$F(\theta + h_\theta, t + h_t) = F(\theta, t) + F_\theta h_\theta + F_t h_t + O(h_\theta^2) \quad (4.39)$$

where F_θ and F_t are partial derivatives of F with respect to θ and t while h_θ and h_t are increments of θ and t respectively. At a given sampling period h_t , the function F

is minimized by solving

$$\begin{aligned} 0 &= \frac{\partial F(\theta + h_\theta, t + h_t)}{\partial \theta} \\ 0 &= F_\theta + F_{\theta\theta}h_\theta + F_{\theta t}h_t + O(h_\theta^2) \end{aligned} \quad (4.40)$$

Dropping the higher order term $O(h_\theta^2)$ yields

$$0 = F_\theta + F_{\theta\theta}h_\theta + F_{\theta t}h_t \quad (4.41)$$

Rearranging (4.41) gives

$$\begin{aligned} F_{\theta\theta}h_\theta &= -(F_\theta + F_{\theta t}h_t) \\ &= -\frac{\partial}{\partial \theta}(F + F_t h_t) \\ &= -\nabla(F + F_t h_t) \end{aligned} \quad (4.42)$$

Recall that $F_{\theta\theta} \equiv H_k$ and the secant equation for approximating the Hessian H_{k+1} is analogous to (4.9),

$$H_{k+1}h_k = g_k^* \quad (4.43)$$

where $h_k = \theta_{k+1} - \theta_k$. However, in this case from (4.42) g_k^* is defined differently as

$$g_k^* = \nabla(F + F_t h_t)_{k+1} - \nabla(F + F_t h_t)_k \quad (4.44)$$

Expanding (4.44) in terms of $f(\theta, t)$ in (4.4) yields

$$g_k^* = \left[J_{k+1}^T(f_{k+1} + \frac{\partial f_{k+1}}{\partial t}h_t) \right] - \left[J_k^T(f_k + \frac{\partial f_k}{\partial t}h_t) \right] \quad (4.45)$$

It should be emphasized that (4.45) differs from g_{k-1} (4.11) in Section 4.3.2 due to the inclusion of the time-dependent term $J_{k+1}^T \frac{\partial f_{k+1}}{\partial t}h_t - J_k^T \frac{\partial f_k}{\partial t}h_t$. For simpler referencing the secant equation (4.43) is referred to as the *dynamic secant equation*.

Since the Hessian H_k is symmetric and often positive symmetric, an algorithm that preserves symmetric positiveness of H_k such as the BFGS method is preferred.

As a result, the derivation of the Hessian H_k update that satisfies the dynamic secant equation can be done analogously to the BFGS derivation. The following derivation of the H_{k+1} approximation follows the steps presented in [19].

From the Lemma 9.2.1 in [19] H_{k+1} is symmetric and positive definite if and only if

$$H_{k+1} = M_{k+1}M_{k+1}^T \quad \text{for some nonsingular } M_{k+1} \in \mathbb{R}^{n \times n} \quad (4.46)$$

such that

$$M_{k+1}M_{k+1}^T h_k = g_k^* \quad \text{for given } h_k, g_k^* \in \mathbb{R}^n \text{ and } h_k \neq 0 \quad (4.47)$$

Equation (4.47) can be rewritten as

$$M_{k+1}v_k = g_k^* \quad (4.48)$$

$$M_{k+1}^T h_k = v_k \quad (4.49)$$

The following procedures are suggested in [19] for solving (4.47),

1. M_{k+1} is selected such that $M_{k+1}v_k = g_k^*$. In this step M_{k+1} is a function of v_k and g_k^*
2. v_k is selected to satisfy $M_{k+1}^T h_k = v_k$

Using the Cholesky factorization on H_k gives

$$H_k = K_k K_k^T \quad (4.50)$$

In order to properly choose M_{k+1} for step 1, Broyden's method is used to approximate M_{k+1} with the minimal changes from K_k , i.e., minimizing the Frobenius norm of the difference between the two matrices,

$$M_{k+1} = K_k + \frac{(g_k^* - K_k v_k)v_k^T}{v_k^T v_k} \quad (4.51)$$

Substituting (4.51) into $v_k = M_{k+1}^T h_k$ in step 2 gives,

$$v_k = K_k^T h_k + v_k \frac{(g_k^* - K_k v_k)^T h_k}{v_k^T v_k} \quad (4.52)$$

Rearranging (4.52) yields

$$v_k \left(1 - \frac{(g_k^* - K_k v_k)^T h_k}{v_k^T v_k} \right) = K_k^T h_k \quad (4.53)$$

Let

$$v_k = \alpha K_k^T h_k \quad (4.54)$$

and then substituting (4.54) into (4.53), multiplying through by the denominator $v_k^T v_k$, and rearranging the equation gives

$$\alpha K_k^T h_k (\alpha^2 h_k^T K_k K_k^T h_k - (g_k^* - K_k v_k)^T h_k) = \alpha^2 h_k^T K_k K_k^T h_k K_k^T h_k$$

where

$$\begin{aligned} \alpha^2 &= \frac{g_k^{*T} h_k}{h_k^T K_k K_k^T h_k} \\ &= \frac{g_k^{*T} h_k}{h_k^T H_k h_k} \end{aligned} \quad (4.55)$$

Substituting α into (4.54) gives an expression for v_k

$$v_k = \left(\frac{g_k^{*T} h_k}{h_k^T H_k h_k} \right)^{1/2} K_k^T h_k \quad (4.56)$$

Next substituting (4.51) into (4.46) gives,

$$\begin{aligned} H_{k+1} &= \left[K_k + \frac{(g_k^* - K_k v_k) v_k^T}{v_k^T v_k} \right] \left[K_k^T + \frac{v_k (g_k^* - K_k v_k)^T}{v_k^T v_k} \right] \\ &= K_k K_k^T + \frac{(g_k^* - K_k v_k) v_k^T K_k^T}{v_k^T v_k} \\ &\quad + \frac{K_k v_k (g_k^* - K_k v_k)^T}{v_k^T v_k} \\ &\quad + \frac{(g_k^* - K_k v_k) v_k^T v_k (g_k^* - K_k v_k)^T}{(v_k^T v_k)^2} \\ &= K_k K_k^T + \frac{g_k^* v_k^T K_k^T - K_k v_k v_k^T K_k^T}{v_k^T v_k} \\ &\quad + \frac{K_k v_k g_k^{*T} - K_k v_k v_k^T K_k^T}{v_k^T v_k} \\ &\quad + \frac{g_k^* g_k^{*T} - g_k^* v_k^T K_k^T - K_k v_k g_k^{*T} + K_k v_k v_k^T K_k^T}{v_k^T v_k} \\ H_{k+1} &= K_k K_k^T + \frac{g_k^* g_k^{*T}}{v_k^T v_k} - \frac{K_k v_k v_k^T K_k^T}{v_k^T v_k} \end{aligned} \quad (4.57)$$

Substituting (4.56) into $v_k^T v_k$ yields

$$\begin{aligned} v_k^T v_k &= \left(\frac{g_k^{*T} h_k}{h_k^T H_k h_k} \right) h_k^T K_k K_k^T h_k \\ &= \left(\frac{g_k^{*T} h_k}{h_k^T H_k h_k} \right) h_k^T H_k h_k \\ &= g_k^{*T} h_k \end{aligned} \quad (4.58)$$

and then substituting (4.56) and (4.58) into (4.57) gives

$$\begin{aligned} H_{k+1} &= K_k K_k^T + \frac{g_k^* g_k^{*T}}{g_k^{*T} h_k} \\ &\quad - \left(\frac{1}{g_k^{*T} h_k} \right) \cdot K_k \left[\left(\frac{g_k^{*T} h_k}{h_k^T H_k h_k} \right)^{1/2} K_k^T h_k \right] \left[h_k^T K_k \left(\frac{g_k^{*T} h_k}{h_k^T H_k h_k} \right)^{1/2} \right] K_k^T \\ &= K_k K_k^T + \frac{g_k^* g_k^{*T}}{g_k^{*T} h_k} \\ &\quad - \left(\frac{1}{g_k^{*T} h_k} \right) \cdot \left(\frac{g_k^{*T} h_k}{h_k^T H_k h_k} \right) [K_k K_k^T h_k h_k^T K_k K_k^T] \end{aligned}$$

Finally, using $H_k = K_k K_k^T$ yields

$$\begin{aligned} H_{k+1} &= H_k + \frac{g_k^* g_k^{*T}}{g_k^{*T} h_k} - \left(\frac{g_k^{*T} h_k}{h_k^T H_k h_k} \right) \left(\frac{H_k h_k h_k^T H_k}{g_k^{*T} h_k} \right) \\ &= H_k + \frac{g_k^* g_k^{*T}}{g_k^{*T} h_k} - \frac{H_k h_k h_k^T H_k}{h_k^T H_k h_k} \end{aligned} \quad (4.59)$$

Although (4.59) appears to be in the same form of the well-known BFGS update [19], in this case g_k^* is defined differently by (4.45) due to the inclusion of the time-dependent terms. Consequently, (4.59) is referred as the *dynamic BFGS (DBFGS)* update. In summary the DBFGS update is

$$H_{k+1} = H_k + \frac{g_k^* g_k^{*T}}{g_k^{*T} h_k} - \frac{H_k h_k h_k^T H_k}{h_k^T H_k h_k} \quad [4.59]$$

where

$$g_k^* = J_{k+1}^T \left(f_{k+1} + \frac{\partial f_{k+1}}{\partial t} h_t \right) - J_k^T \left(f_k + \frac{\partial f_k}{\partial t} h_t \right) \quad (4.60)$$

$$h_k = \theta_{k+1} - \theta_k \quad (4.61)$$

Pseudo-code: The DBFGS method

Given: $\mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$, $P_0 \in \mathbb{R}^{n \times n}$, $\lambda \in (0, 1)$

Initialize: J_0 , θ_0 , θ_1 , H_0 and P_0

for $k = 1, \dots$ **do**

Calculate: \hat{J}_k using the dynamic Broyden's method for a Jacobian estimation

$$\Delta f = f_k - f_{k-1}$$

$$h_{k-1} = \theta_k - \theta_{k-1}$$

$$h_t = t_k - t_{k-1}$$

$$\tilde{h} = \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix}$$

$$\tilde{J}_{k-1} = \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix}$$

$$\tilde{J}_k = \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1}$$

$$\tilde{P}_k = \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right)$$

Calculate: \hat{H}_k using the DBFGS update

$$g_{k-1}^* = \hat{J}_k^T \left(f_k + \frac{\partial f_k}{\partial t} h_t \right) - \hat{J}_{k-1}^T \left(f_{k-1} + \frac{\partial f_{k-1}}{\partial t} h_t \right)$$

$$\hat{H}_k = \hat{H}_{k-1} + \frac{g_{k-1}^* g_{k-1}^{* T}}{g_{k-1}^{* T} h_{k-1}} - \frac{\hat{H}_{k-1} h_{k-1} h_{k-1}^T \hat{H}_{k-1}}{h_{k-1}^T \hat{H}_{k-1} h_{k-1}}$$

Calculate: θ_{k+1} using the quasi-Newton's method

$$\theta_{k+1} = \theta_k - \left(\hat{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

end for

Figure 4.2: A pseudo-code for the DBFGS update

The DBFGS update is implemented with the dynamic Broyden's method for a Jacobian approximation similar to the DGN-PBM algorithm as shown in the pseudo-code of Figure 4.2.

The Hessian approximation H_k using the standard BFGS method,

$$H_k = H_{k-1} + \frac{g_{k-1} g_{k-1}^T}{g_{k-1}^T h_{k-1}} - \frac{H_{k-1} h_{k-1} h_{k-1}^T H_{k-1}}{h_{k-1}^T H_{k-1} h_{k-1}} \quad (4.62)$$

where g_{k-1} does not include the dynamic terms $J_k^T \frac{\partial f_k}{\partial t} h_t - J_{k-1}^T \frac{\partial f_{k-1}}{\partial t} h_t$ and is defined as

$$g_{k-1} = J_k^T f_k - J_{k-1}^T f_{k-1} \quad (4.63)$$

is used as a comparative method to the DBFGS update (4.59). For future reference the Hessian approximation (4.62) where g_{k-1} is defined as in (4.63) is referred as the *standard* BFGS update.

4.4.2 Approximation of the residual S_k using the BFGS method

In Section 4.2.2 a secant technique used for the Hessian \hat{H}_k approximation is extended to estimate the residual S_k . As presented in [49], NL2SOL [17, 20] demonstrates a stability improvement for solving the large-residual optimization problems. This method further shows a potential to provide a more rapid and robust algorithm to achieve static target tracking [25] for the large-residual visual servoing problem. As a result, two approaches inspired by the residual S_k approximation presented in NL2SOL are developed for moving target tracking, namely the BFGS method and the modified BFGS method. This section reviews an approximation of the residual S_k using the BFGS method, then Section 4.4.3 discusses the residual approximation using a novel modified BFGS method.

Dennis et al. [17, 20] introduce a secant equation for the residual S_k as

$$S_k h_{k-1} = J_k^T f_k - J_{k-1}^T f_k \equiv z_{k-1}^* \quad [4.31]$$

Instead of using the DFP-like method to approximate S_k from S_{k-1} as in [17, 20], the BFGS method, which is well-known to be more robust and stable than the DFP method, can be used,

$$\hat{S}_k = \hat{S}_{k-1} + \frac{z_{k-1}^* z_{k-1}^{*T}}{z_{k-1}^{*T} h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad (4.64)$$

where

$$z_{k-1}^* = J_k^T f_k - J_{k-1}^T f_k$$

Then \hat{S}_k from (4.64) is used to approximate \hat{H}_k as

$$\hat{H}_k = \hat{J}_k^T \hat{J}_k + \hat{S}_k \quad (4.65)$$

where \hat{J}_k can be approximated using the dynamic Broyden's method as in the DGN-PBM algorithm [59]. Since in this case the objective function $F(\theta, t)$ is now a function of both the robot joint angles θ and time t , the image error vector $f(\theta, t)$ may change due to either robot movement, target motion, or a combination of both. The inclusion of the time-varying term $\frac{\partial f_k}{\partial t} h_t$ is necessary for estimating \hat{J}_k and $\frac{\partial f_k}{\partial t}$ simultaneously.

Then the dynamic full quasi-Newton method is utilized to estimate θ_{k+1} as

$$\theta_{k+1} = \theta_k - \left(\hat{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right) \quad (4.66)$$

This algorithm is called the *dynamic full Newton method with BFGS* algorithm or DFN-BFGS. The pseudo-code summarizing the implementation of the DFN method is shown in Figure 4.3.

4.4.3 Approximation of the residual S_k using the MBFGS update

A slight modification of (4.64) has been made where $g_{k-1}^T h_{k-1}$ replaces $z_{k-1}^{*T} h_{k-1}$,

$$\hat{S}_k = \hat{S}_{k-1} + \frac{z_{k-1}^* z_{k-1}^{*T}}{g_{k-1}^T h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad (4.67)$$

and

$$z_{k-1}^* = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k$$

$$g_{k-1} = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_{k-1}$$

This is called the *modified BFGS* method or MBFGS for estimating the residual S_k .

The MBFGS method is also implemented in conjunction with a Jacobian estimation presented in the DGN-PBM algorithm [59] to approximate θ_{k+1} as in (4.66). The summary of this approach is shown in the pseudo-code in Figure 4.4.

Pseudo-code: The Dynamic Full Newton Method with BFGS Algorithm (DFN-BFGS)

Given: $\mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$, $P_0 \in \mathbb{R}^{n \times n}$, $\lambda \in (0, 1)$

Initialize: J_0 , θ_0 , θ_1 , H_0 and P_0

for $k = 1, \dots$ **do**

Calculate: \hat{J}_k using the DGN-PBM algorithm [59]

$$\begin{aligned}\Delta f &= f_k - f_{k-1} \\ h_{k-1} &= \theta_k - \theta_{k-1} \\ h_t &= t_k - t_{k-1} \\ \tilde{h} &= \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix} \\ \tilde{J}_{k-1} &= \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix} \\ \tilde{J}_k &= \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1} \\ \tilde{P}_k &= \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right)\end{aligned}$$

Update the residual \hat{S}_k

$$\hat{S}_k = \hat{S}_{k-1} + \frac{\hat{z}_{k-1}^* \hat{z}_{k-1}^{*T}}{\hat{z}_{k-1}^{*T} h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}}$$

where

$$z_{k-1}^* = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k$$

Calculate \acute{H}_k

$$\acute{H}_k = \hat{J}_k^T \hat{J}_k + \hat{S}_k$$

Calculate: θ_{k+1} using the dynamic full quasi-Newton method

$$\theta_{k+1} = \theta_k - \left(\acute{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

end for

Figure 4.3: Pseudo-code for the DFN-BFGS approach

The main difference is that the denominator $z_{k-1}^T h_{k-1}$ in (4.64) is replaced by $g_{k-1}^T h_{k-1}$ in (4.67). In order to understand the impact of these quantities recall that the Hessian H_k approximation satisfies the secant equation,

$$H_k h_{k-1} = g_{k-1} \quad [4.9]$$

where

$$h_{k-1} = \theta_k - \theta_{k-1} \quad [4.10]$$

and

$$g_{k-1} = J_k^T f_k - J_{k-1}^T f_{k-1} \quad [4.11]$$

Transposing (4.9) and multiplying through by h_k gives

$$h_{k-1}^T H_k h_{k-1} = g_{k-1}^T h_{k-1} \quad (4.68)$$

The quantity $g_{k-1}^T h_{k-1}$ is known as the *approximate curvature* [27] of the objective function F_{k-1} . The new quadratic model q_k of the function F_k can be approximated using the update \hat{H}_k satisfying (4.9),

$$q_k(\theta) = F_k + \nabla F_k^T (\theta - \theta_k) + \frac{1}{2} (\theta - \theta_k)^T H_k (\theta - \theta_k) \quad (4.69)$$

Equation (4.68) implies that the approximate curvature $g_{k-1}^T h_{k-1}$ implements the *exact* curvature of the new quadratic model q_k in the direction of h_{k-1} .

A similar derivation of $z_{k-1}^* T h_{k-1}$ in (4.64) can be done. Each \hat{S}_k update from (4.64) satisfies the secant equation,

$$S_k h_{k-1} = J_k^T f_k - J_{k-1}^T f_k \quad \equiv \quad z_{k-1}^* \quad [4.31]$$

or

$$S_k h_{k-1} = z_{k-1}^* \quad (4.70)$$

Then transposing (4.70) and multiplying through by h_{k-1} gives

$$h_{k-1}^T S_k h_{k-1} = z_{k-1}^{* T} h_{k-1} \quad (4.71)$$

Since S_k is only a component of H_k , i.e., $H_k = J_k^T J_k + S_k$, the approximate curvature $z_{k-1}^{*T} h_{k-1}$ only represents a part of the curvature information of quadratic model q_k . As a result, replacing the approximate curvature $g_{k-1}^{*T} h_{k-1}$ as in (4.67) basically enforces the curvature information of the function F_k into the residual S_k approximation. Although the exact relationship between $z_k^{*T} h_k$ and the approximate curvature $g_k^{*T} h_k$ of the function $F(\theta, t)$ has not been established, simulation results in Chapter 6 shows significant improvement using (4.67) over the method in (4.64).

Another aspect of (4.67) can be established by multiplying the second term of the right hand side of (4.64) with the unity term,

$$\left(\frac{z_{k-1}^{*T} h_{k-1}}{g_{k-1}^{*T} h_{k-1}} \cdot \frac{g_{k-1}^{*T} h_{k-1}}{z_{k-1}^{*T} h_{k-1}} \right)$$

to give

$$\hat{S}_k = \hat{S}_{k-1} + \frac{z_{k-1}^{*T} z_{k-1}^{*T}}{z_{k-1}^{*T} h_{k-1}} \left(\frac{z_{k-1}^{*T} h_{k-1}}{g_{k-1}^{*T} h_{k-1}} \cdot \frac{g_{k-1}^{*T} h_{k-1}}{z_{k-1}^{*T} h_{k-1}} \right) - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad (4.72)$$

or

$$\hat{S}_k = \hat{S}_{k-1} + \left(\frac{z_{k-1}^{*T} z_{k-1}^{*T}}{g_{k-1}^{*T} h_{k-1}} \right) \left(\frac{g_{k-1}^T h_{k-1}}{z_{k-1}^{*T} h_{k-1}} \right) - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad (4.73)$$

Substituting (4.71) and (4.68) for $z_{k-1}^{*T} h_{k-1}$ and $g_{k-1}^T h_{k-1}$ gives

$$\hat{S}_k = \hat{S}_{k-1} + \left(\frac{z_{k-1}^{*T} z_{k-1}^{*T}}{g_{k-1}^{*T} h_{k-1}} \right) \left(\frac{h_{k-1}^T H_k h_{k-1}}{h_{k-1}^T S_k h_{k-1}} \right) - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad (4.74)$$

Substituting $H_k = J_k^T J_k + \hat{S}_k$ gives

$$\hat{S}_k = \hat{S}_{k-1} + \left(\frac{z_{k-1}^{*T} z_{k-1}^{*T}}{g_{k-1}^{*T} h_{k-1}} \right) \left(\frac{h_{k-1}^T (J_k^T J_k + \hat{S}_k) h_{k-1}}{h_{k-1}^T S_k h_{k-1}} \right) - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad (4.75)$$

Comparing (4.75) with (4.67) requires that

$$\frac{h_{k-1}^T (J_k^T J_k + \hat{S}_k) h_{k-1}}{h_{k-1}^T \hat{S}_k h_{k-1}} = 1 \quad (4.76)$$

for (4.67) and (4.75) to be the same. This is the case only if

$$h_{k-1}^T \hat{S}_k h_{k-1} \gg h_{k-1}^T (J_k^T J_k) h_{k-1}$$

so that

$$\frac{h_{k-1}^T \left(J_k^T J_k + \hat{S}_k \right) h_{k-1}}{h_{k-1}^T \hat{S}_k h_{k-1}} \approx 1 \quad (4.77)$$

Therefore, for the case that $h_{k-1}^T \hat{S}_k h_{k-1}$ is relatively large compared to the term $h_{k-1}^T J_k^T J_k h_{k-1}$, the MBFGS method is the same as using the unmodified BFGS method for approximating \hat{S}_k (4.64). Otherwise, the MBFGS method generates different results from the unmodified BFGS method. Simulation results in Chapter 6 show that the approach in Figure 4.4 outperforms approaches in Figure 4.3 and in Figure 4.2 when $\|S_k\|_F$ is relatively the same as of $\|\hat{J}_k^T \hat{J}_k\|_F$ where $\|\cdot\|_F$ refers to the Frobenius norm.

To minimize computational cost this novel MBFGS method is only employed on the large-residual problem (not necessarily restricted to only the case that S_k is dominant compared to the linear term $\hat{J}_k^T \hat{J}_k$). If the residual S_k becomes relatively small so that S_k can be approximated as zero, then the quasi-Newton method becomes the quasi-Gauss-Newton method and the DGN-PBM approach is used instead. Consequently, a switching method where the algorithm presented in Figure 4.4 is applied whenever $\|f_k\|$ is greater than a criteria and the DGN-PBM approach is employed otherwise is created to efficiently calculate the next θ_{k+1} when the residual error is significant. The details of the switching method are discussed in Section 4.7.

4.5 Convergence Analysis of the MBFGS Method

In optimization problems the BFGS method is a well-known and widespread approach for approximating the Hessian matrix in a quasi-Newton method with satisfactory convergence properties. For example, Powell [62] proved a global convergence using the BFGS method with an inexact line search. Byrd, Nocedal, and Yuan [8] extended the proof of Powell [62] for convergence analysis of other algorithms in Broyden's class formula excluding the DFP method where $\kappa = 1$ in (4.78). The Broyden's class

Pseudo-code: The Modified BFGS method for the Residual Approximation (MBFGS)

Given: $\mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$, $P_0 \in \mathbb{R}^{n \times n}$, $\lambda \in (0, 1)$

Initialize: \hat{J}_0 , θ_0 , θ_1 , H_0 and P_0

for $k = 1, \dots$ **do**

Calculate: \hat{J}_k using a Jacobian estimation in the DGN-PBM algorithm [59]

$$\begin{aligned}\Delta f &= f_k - f_{k-1} \\ h_{k-1} &= \theta_k - \theta_{k-1} \\ h_t &= t_k - t_{k-1} \\ \tilde{h} &= \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix} \\ \tilde{J}_{k-1} &= \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix} \\ \tilde{J}_k &= \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1} \\ \tilde{P}_k &= \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right)\end{aligned}$$

Update the residual \hat{S}_k

$$\hat{S}_k = \hat{S}_{k-1} + \frac{\hat{z}_{k-1}^* \hat{z}_{k-1}^{*T}}{\hat{g}_{k-1}^{*T} \hat{h}_{k-1}} - \frac{\hat{S}_{k-1} \hat{h}_{k-1} \hat{h}_{k-1}^T \hat{S}_{k-1}}{\hat{h}_{k-1}^T \hat{S}_{k-1} \hat{h}_{k-1}}$$

where

$$\begin{aligned}z_{k-1}^* &= \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k \\ g_{k-1} &= \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_{k-1}\end{aligned}$$

Calculate \hat{H}_k

$$\hat{H}_k = \hat{J}_k^T \hat{J}_k + \hat{S}_k$$

Calculate: θ_{k+1} using the dynamic full quasi-Newton method

$$\theta_{k+1} = \theta_k - \left(\hat{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

end for

Figure 4.4: Pseudo-code for the MBFGS approach

formula [6] for Hessian approximation is

$$\hat{H}_{k+1} = \hat{H}_k + \frac{g_k g_k^T}{g_k^T h_k} - \frac{\hat{H}_k h_k h_k^T \hat{H}_k}{h_k^T \hat{H}_k h_k} + \kappa \left(h_k^T \hat{H}_k h_k \right) w_k w_k^T \quad (4.78)$$

where $\kappa \in [0, 1]$,

$$g_k = \nabla F_{k+1} - \nabla F_k$$

$$h_k = \theta_{k+1} - \theta_k$$

and

$$w_k = \left[\frac{g_k}{g_k^T h_k} - \frac{H_k h_k}{h_k^T H_k h_k} \right]$$

For convenience in future reference, let

$$d_k \equiv \nabla F_k \quad (= J_k^T f_k)$$

so that g_k becomes

$$g_k \equiv d_{k+1} - d_k$$

For $\kappa = 0$, (4.78) yields the BFGS method and for $\kappa = 1$, it gives the DFP method. Even though it has been shown in [21] that all members of this class give the same result with exact line searches, Byrd et al. [8] states that the performance of this class varies with inexact line searches, which are more efficient and require less computational cost compared to exact line searches. Since the MBFGS method is developed from the BFGS method, the convergence analysis of the MBFGS method heavily depends on the proof of the Broyden's class method presented in [8] where $\kappa = 0$ with inexact line searches.

The Hessian approximation from (4.78) is used in a quasi-Newton method for determining θ_{k+1} ,

$$\theta_{k+1} = \theta_k + \alpha_k s_k \quad (4.79)$$

where

$$s_k = -\hat{H}_k^{-1}d_k \quad (4.80)$$

α is a step length parameter that is selected using an inexact line search algorithm that satisfies the Wolfe Conditions [8, 50] to ensure a sufficient reduction of the objective function $F(\theta)$ as

$$F_{k+1} \leq F_k + c_1 \alpha_k d_k^T s_k \quad (4.81)$$

$$c_2 d_k^T s_k \leq d_{k+1}^T s_k \quad (4.82)$$

where c_1, c_2 are positive constants such that $0 < c_1 < \frac{1}{2}$ and $0 < c_1 < c_2 < 1$.

If the objective function $F(\theta, t)$ has a minimizer θ^* at a given time t then $\hat{H}(\theta^*)$, it is assumed to be positive definite. Dennis and Moré [18] proved that if the step length α_k is always chosen as 1, it satisfies conditions (4.81) and (4.82), and

$$\sum_{k=0}^{\infty} \|\theta_k - \theta^*\| < \infty \quad (4.83)$$

so θ_k converges superlinearly to θ^* .

Other representative studies related to the BFGS and Broyden class approaches are presented in [28, 66, 72, 74].

The goal of solving an optimization problem in this analysis is to find θ_{k+1} that minimizes the objective function $F(\theta, t)$ at any given instant t so t is held constant and is dropped from the convergence analysis for simplicity. Since Byrd et al. [8] have already established the convergence analysis for the Broyden's class formula used in conjunction with a line search algorithm, the MBFGS convergence analysis mainly follows the same process. The major difference is due to the fact that in [8] the BFGS method (4.78) is used to approximate the whole Hessian matrix \hat{H}_k in (4.80), which is subsequently used in the quasi-Newton method to calculate θ_{k+1} as in (4.79). In contrast, the MBFGS method is used to approximate the residual S_k which is only a

component of the Hessian \hat{H}_k , i.e., $\hat{H}_k = \hat{J}_k^T \hat{J}_k + \hat{S}_k$. Consequently, the resulting S_k update using the MBFGS method is indirectly used to calculate θ_{k+1} .

For the remainder of the convergence analysis, the Jacobian J_k is assumed to be known and is positive definite. A proper method dealing with a case when the Hessian is ill-conditioned or becomes singular is discussed in Chapter 6.

4.5.1 Convergence Analysis of the BFGS method

This section reviews the convergence analysis of the Broyden's class formula for approximating the Hessian \hat{H}_k presented in [8] that is implemented with an inexact line search for determining a solution of unconstrained optimization using a quasi-Newton method. It shows that if the objective function is assumed to be uniformly convex, the sequence of θ_k from (4.79)-(4.80) using the Hessian approximation \hat{H}_k from (4.78) with $\kappa \in [0, 1)$ converges to a solution linearly. The study in [8] further implies the results of Dennis and Moré [18] and of Griewank and Toint [28] for superlinear convergence if the step length $\alpha_k = 1$ is always chosen when it is permissible. However, this convergence analysis implies that θ_k is sufficiently close to the minimizer θ^* and the sequence of θ_k remains in the neighborhood of the solution. In [8] it is also shown that the iterates generated by (4.79)-(4.80) gives superlinear convergence even if the different values of $\kappa \in [0, 1)$ are applied in (4.78) as long as $\cos(\vartheta_k)$, the angle between the gradient vector d_k and the Newton step h_k , is bounded away from zero. This concept is in fact the key attribute applied to the MBFGS algorithm for proving its convergence in Section 4.5.2.

Since a similar analysis with the same line search, the same assumptions, and some of the lemmas developed in [8] can be extended to the MBFGS algorithm, it is appropriate to review these assumptions, lemmas, and theorems herein as fundamental knowledge for the extended convergence study of the MBFGS algorithm, which is developed in Section 4.5.2. Since only the BFGS algorithm is relevant, the review of

the Broyden's class formula is only focused on the BFGS method i.e., $\kappa = 0$ in (4.78).

The following two assumptions hold throughout the remainder of the development. Note that Assumption 1 is originally presented as a preliminary rather than an assumption in [8]. However, for simplicity of the extended analysis development (the MBFGS algorithm), it is suitable to present these conditions as Assumption 1.

Assumption 1. *The objective function $F(\theta)$ is assumed to be twice continuously differentiable and is uniformly convex on the level set D where $D = \{x \in \mathbb{R}^m : F(x) \leq F(\theta_1)\}$. Let $H(\theta)$ be the true value of the Hessian matrix of $F(\theta)$ and let \bar{H} is defined as*

$$\bar{H} = \int_0^1 H(\theta_k + \tau h_k) d\tau$$

\bar{H} is an average value of the Hessian $H(\theta)$ over the interval between θ_k and θ_{k+1} . Also let θ^* be the unique minimizer of $F(\theta)$ in D and the true Hessian at that point is $H(\theta^*)$. The approximated Hessian \hat{H}_k is assumed to be positive definite and consequently the inequality

$$g_k^T h_k > 0 \quad (4.84)$$

holds for all values of k .

Assumption 2 (Byrd et al. [8]). *The level set D is convex and m_1 and m_2 are positive constants such that*

$$m_1 \|q\|^2 \leq q^T H(\theta) q \leq m_2 \|q\|^2 \quad (4.85)$$

for all $q \in \mathbb{R}^m$ and all $\theta \in D$.

The average Hessian \bar{H} satisfies the following secant equation

$$g_k = \bar{H} h_k \quad (4.86)$$

Substituting h_k for q and $g_k^T = h_k^T \bar{H}$ for $q^T H(\theta)$ in (4.85) yields

$$m_1 \|h_k\|^2 \leq g_k^T h_k \leq m_2 \|h_k\|^2 \quad (4.87)$$

From the definition of g_k ,

$$g_k = d_{k+1} - d_k \quad (4.88)$$

Transposing and multiplying (4.88) by h_k gives

$$g_k^T h_k = d_{k+1}^T h_k - d_k^T h_k \quad (4.89)$$

From the condition in (4.82), the approximate curvature $g_k^T h_k$ is constrained by

$$-(1 - c_2)d_k^T h_k \leq g_k^T h_k = d_{k+1}^T h_k - d_k^T h_k \quad (4.90)$$

The quantity $d_k^T h_k$ directly relates to the angle ϑ_k between the steepest descent direction $-d_k$ and the step h_k ,

$$-d_k^T h_k = \|d_k\| \|h_k\| \cos \vartheta_k \quad (4.91)$$

The angle ϑ_k plays a crucial role determining the length of the step h_k that results in the reduction of the function $F(\theta)$ at each iteration and is presented in a number of studies such as [62, 74]. Indeed, the angle ϑ_k is the key variable used for convergence proof of the Broyden's class formula in [8]. In order to understand the effect of $\cos(\vartheta_k)$ on the sequence of θ_k obtained from (4.79)-(4.80) using the Hessian approximation \hat{H}_k from (4.78), it is necessary to review the following lemmas from [8].

From (4.87), (4.90), and (4.91) the lower and the upper bounds of $\|h_k\|$ can be established. The range of $\|h_k\|$ and the upper bound of $F_{k+1} - F^*$ are concluded in Lemma 2.1 in [8] which is herein presented as Lemma 1 and Lemma 2 respectively.

Lemma 1 (Lemma 2.1 [8]). (*see proof on p. 97*)

Consider the resultant iteration, $\theta_{k+1} = \theta_k + \alpha_k s_k$, using (4.79), where α_k satisfies the Wolfe conditions (4.81) and (4.82). If Assumption 1 and Assumption 2 hold, then

$$c_3 \|d_k\| \cos \vartheta_k \leq \|h_k\| \leq c_4 \|d_k\| \cos \vartheta_k \quad (4.92)$$

where c_3 and c_4 are positive constants such that $c_3 = \frac{(1 - c_2)}{m_2}$ and $c_4 = \frac{2(1 - c_1)}{m_1}$.

The upper bound of $F_{k+1} - F^*$ is

Lemma 2 (Byrd et al. [8]). (*see proof on p. 99*)

$$F_{k+1} - F^* \leq [1 - c_1 m_1 c_3 \cos^2 \vartheta_k] (F_k - F^*) \quad (4.93)$$

Equation (4.93) requires that $\cos \vartheta_k$ needs to be bounded away from zero so that θ_k converges to θ^* . Byrd et al. [8] proved this statement by using the trace of the Hessian \hat{H}_k approximation in (4.78),

$$\text{Tr}(\hat{H}_{k+1}) \leq \text{Tr}(\hat{H}_k) - \frac{\|\hat{H}_k h_k\|^2}{h_k^T \hat{H}_k h_k} + \frac{\|g_k\|^2}{g_k^T h_k} \quad (4.94)$$

where $\text{Tr}(\hat{H}_{k+1})$ is the trace of \hat{H}_{k+1} .

It is shown that each term in (4.94) is bounded (see Lemma 3.1 in [8]),

Lemma 3 (Byrd et al. [8]).

$$\frac{\|g_k\|^2}{g_k^T h_k} \leq m_2 \quad (4.95)$$

$$-\frac{\|\hat{H}_k h_k\|^2}{h_k^T \hat{H}_k h_k} \leq -\frac{\alpha_k}{c_4 \cos^2 \vartheta_k} \quad (4.96)$$

The proof of each inequality is presented in [8].

Substituting the inequalities (4.95) and (4.96) in (4.94) gives

$$\text{Tr}(\hat{H}_{k+1}) \leq \text{Tr}(\hat{H}_k) + m_2 - \frac{\alpha_k}{c_4 \cos^2 \vartheta_k} \quad (4.97)$$

Because the approximated \hat{H}_k is assumed to be positive definite, so is its trace $\text{Tr}(\hat{H}_k)$ [75]. In (4.97) the third term on the right-hand side reduces the trace and this term is proportional to $\frac{\alpha_k}{\cos^2 \vartheta_k}$.

In fact, [8] shows that

$$0 < \frac{\alpha_k}{c_4 \text{Tr}(\hat{H}_k)} \leq \cos \vartheta_k \quad (4.98)$$

Thus $\cos \vartheta_k$ is small when (i) α_k is small or (ii) $\text{Tr}(\hat{H}_k)$ is large. Two cases are considered in [8]. The first case assumes that the step length α_k is bounded away from zero and that $\cos \vartheta_k$ is arbitrarily small so the third term of the right-hand

side in (4.97) becomes dominant and $\text{Tr}(\hat{H}_{k+1})$ is reduced. However, from (4.98) if $\cos \vartheta_{k+1}$ becomes arbitrarily small, the bound of $\text{Tr}(\hat{H}_{k+1})$ becomes large (since α_{k+1} is assumed to be bounded away from zero). Thus the calculated $\text{Tr}(\hat{H}_{k+1})$ is self-conflicting that limits its value to impossibly become too large. Therefore, the tendency of $\cos(\vartheta_k)$ becoming close to zero is self limiting if α_k is assumably bounded away from zero.

The second case assumes that the step length α_k tends to zero. In this case consider Lemma 2.2 in [8] which establishes the relationship between α_k and $\frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2}$,

Lemma 4 (Byrd et al. [8]).

$$c_3 \frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2} \leq \alpha_k \leq c_4 \frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2} \quad (4.99)$$

When α_k is close to zero, $\frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2}$ becomes very small and the trace equation (4.97) is no longer useful [8]. The relation between the quantity $\frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2}$ and $\text{Tr}(\hat{H}_k)$ can be established as

$$\frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2} \leq \frac{\|\hat{H}_k h_k\| \|h_k\|}{\|h_k\|^2}$$

cancelling $\|h_k\|$ in both the numerator and denominator on the right-hand side gives

$$\frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2} \leq \frac{\|\hat{H}_k h_k\|}{\|h_k\|} \quad (4.100)$$

where $\frac{\|\hat{H}_k h_k\|}{\|h_k\|}$ holds in the following inequality [8],

$$\frac{\|\hat{H}_k h_k\|}{\|h_k\|} \leq \text{Tr}(\hat{H}_k) \quad (4.101)$$

Note that $\|\cdot\|$ of a vector is referred to the Euclidean norm where as $\|\cdot\|_F$ of a matrix is referred to the Frobenius norm of a matrix throughout the remainder of the analysis.

The derivation of (4.101) is presented in the proof of Lemma 10 on p. 107.

Combining (4.100) and (4.101) gives

$$\frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2} \leq \text{Tr}(\hat{H}_k) \quad (4.102)$$

and it can be concluded that if $\frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2}$ goes to zero, which is due to small eigenvalues of \hat{H}_k , $\text{Tr}(\hat{H}_k)$ becomes very small and so does its determinant. The determinant of \hat{H}_k is expressed in [56] as

$$\det(\hat{H}_{k+1}) \geq \det(\hat{H}_k) \frac{g_k^T h_k}{h_k^T \hat{H}_k h_k} \quad (4.103)$$

Equations (4.102) and (4.103) are used to show that $\text{Tr}(\hat{H}_{k+1}) \leq c_0^k$ where c_0 is a positive constant [8]. Then the relation between the trace and the determinant of \hat{H}_k is developed in [8] as

$$\prod_{i=1}^k \frac{1 - c_2}{\alpha_i} \leq \frac{1}{\det(\hat{H}_1)} \left[\frac{\text{Tr}(\hat{H}_{k+1})}{n} \right]^n$$

or

$$\prod_{i=1}^k \frac{1 - c_2}{\alpha_i} \leq \frac{1}{\det(\hat{H}_1)n^n} (c_0^n)^k \quad (4.104)$$

where n is the number of eigenvalues of \hat{H} . From (4.104) α_k is in fact bounded away from zero (see Lemma 3.2 in [8] for details) and is presented herein as

Lemma 5 (Byrd et al. [8]). *If $\kappa \in [0, 1]$, there exists a constant $c > 0$ such that*

$$\prod_{i=1}^k \alpha_i \geq c^k \quad (4.105)$$

for all $k \geq 1$

Lemma 5 contradicts to the assumption of the second case whereas the step length α_k is assumed to be arbitrarily small, thus this case is not possible. Due to the result from the first case in which the tendency of $\cos(\vartheta_k)$ to zero is self-limiting $\text{Tr}(\hat{H}_k)$ from becoming negative. Byrd et al. [8] conclude that “all the updates in the restricted Broyden class have a strong self correcting property with respect to the determinant.”

Then Byrd et al. [8] introduce Theorem 3.1 that is presented as

Theorem 4.5.1 (Byrd et al. [8]). *Assume that the function $F(\theta)$ satisfies Assumption 1 and Assumption 2 and let θ_1 be a starting point, then for any positive definite \hat{H}_1 , the sequence of θ_k , generated by (4.78)-(4.80) with $\kappa \in [0, 1)$ and line search satisfying (4.81)-(4.82), converges to θ^* .*

To prove this theorem, recall the trace inequality as

$$0 < Tr(\hat{H}_{k+1}) \leq Tr(\hat{H}_k) + m_2 - \frac{\alpha_k}{c_4 \cos^2 \vartheta_k} \quad [4.97]$$

Since it is proved that α_k is bounded from below in Lemma 5, if there are too many iterates such that $\{\cos \vartheta_k\}$ becomes arbitrarily small then $Tr(\hat{H}_{k+1})$ can become negative, which violates the fact that \hat{H}_{k+1} is positive definite. As a result, Theorem 4.5.1 is proved by contradiction in which it seeks to prove that (4.97) is satisfied even if $\{\cos \vartheta_k\} \rightarrow 0$. It is shown that $Tr(\hat{H}_{k+1})$ is negative for sufficiently large k when $\{\cos \vartheta_k\} \rightarrow 0$, thus it is impossible to satisfy (4.97). As a result, $\cos \vartheta_k$ is bounded from below. Then combining this result with Lemma 2 , Byrd et al. conclude that the iterates θ_k converge to the solution.

4.5.2 Convergence Analysis of the MBFGS method

The Hessian approximation in the case of the MBFGS method is

$$\hat{H}_k = J_k^T J_k + \hat{S}_k \quad (4.106)$$

in which the MBFGS method is used to update \hat{S}_k as

$$\hat{S}_{k+1} = \hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} - \frac{\hat{S}_k h_k h_k^T \hat{S}_k}{h_k^T \hat{S}_k h_k} \quad (4.107)$$

where

$$h_k = \theta_{k+1} - \theta_k$$

$$z_k^* = J_{k+1}^T f_{k+1} - J_k^T f_{k+1}$$

$$g_k = J_{k+1}^T f_{k+1} - J_k^T f_k$$

$$\equiv d_{k+1} - d_k$$

In this section the Jacobian J_k is again assumed to be known and is positive definite. For simplicity of referencing, the *BFGS-QN* algorithm is referred to the quasi-Newton method in which the BFGS method presented in Byrd et al. [8] where $\kappa = 0$ is used to approximate \hat{H}_k in calculating of θ_{k+1} . The *MBFGS-QN* algorithm is referred to as the quasi-Newton method in which the MBFGS method is used to approximate \hat{H}_k in calculating of θ_{k+1} . Both approaches are assumed to satisfy the Wolfe conditions (4.81)-(4.82) to generate the sequence θ_{k+1} as

$$\theta_{k+1} = \theta_k + \alpha_k \dot{s}_k \quad (4.108)$$

or

$$h_k = \alpha_k \dot{s}_k \quad (4.109)$$

where

$$\dot{s}_k = -\hat{H}_k^{-1} d_k \quad (4.110)$$

Since \dot{s}_k is approximated from \hat{H}_k , which is different from the s_k calculation using \hat{H}_k in the previous sections, the notation \dot{s}_k is introduced to distinguish between the two approaches. Note that, $d_k = \nabla F_k = J_k^T f_k$ is the same for both methods. Furthermore, Assumption 1 and Assumption 2 are still valid and are also assumed in this section (both assumptions are hold in regardless of how the Hessian matrix is approximated). In addition, the following assumptions are applied in the MBFGS method.

Assumption 3. *The objective function $F(\theta)$ is assumed to be twice continuously differentiable and is uniformly convex on the level set D where $D = \{x \in \mathbb{R}^m : F(x) \leq F(\theta_1)\}$. Let $S(\theta)$ be the true value of $\nabla^2 F(\theta)$ and let \bar{S} is an average value of $\nabla^2 F(\theta)$ over the interval between θ_k and θ_{k+1} defined as*

$$\bar{S} = \int_0^1 S(\theta_k + \tau h_k) d\tau$$

For this section it is assumed that the approximated residual \hat{S}_k is positive definite and consequentially the inequality

$$z_k^{*T} h_k > 0 \quad (4.111)$$

holds for all values of k .

Assumption 4. The level set D is convex and w_1 and w_2 are positive constants such that

$$w_1\|\dot{q}\|^2 \leq \dot{q}^T S(\theta)\dot{q} \leq w_2\|\dot{q}\|^2 \quad (4.112)$$

for all $\dot{q} \in \mathbb{R}^m$ and all $\theta \in D$.

The average residual \bar{S} satisfies the following secant equation

$$z_k^* = \bar{S}h_k \quad (4.113)$$

Substituting h_k for q and $z_k^{*T} = h_k^T \bar{S}$ for $q^T S(\theta)$ in (4.112) yields

$$w_1\|h_k\|^2 \leq z_k^{*T} h_k \leq w_2\|h_k\|^2 \quad (4.114)$$

Since \hat{S}_k , which is a component of the Hessian \acute{H}_k , is calculated from the MBFGS approach, the relationship between z^* and $\cos \vartheta_k$ and between s_k and h_k as in (4.91) in Section 4.5.1 can be established. Transposing both sides of (4.113) and multiplying through by h_k yields

$$h_k^T \bar{S}h_k = z_k^{*T} h_k \quad (4.115)$$

$$= \|z_k^*\| \|h_k\| \cos \gamma_k \quad (4.116)$$

where γ_k is the angle between z_k^* and h_k . Recall

$$-d_k^T h_k = \|d_k\| \|h_k\| \cos \vartheta_k \quad [4.91]$$

and

$$h_k = -\alpha_k \acute{H}_k^{-1} d_k \quad [4.109]$$

which is rewritten as

$$\dot{H}_k h_k = -\alpha_k d_k \quad (4.117)$$

Transposing, and multiplying through by h_k yields

$$h_k^T \dot{H}_k h_k = -\alpha_k d_k^T h_k \quad (4.118)$$

To obtain the relation between $\cos \vartheta_k$ and $\cos \gamma_k$, substituting (4.106) for \dot{H}_k and (4.91) for $d_k^T h_k$ gives

$$h_k^T \left(J_k^T J_k + \hat{S}_k \right) h_k = \alpha_k \|d_k\| \|h_k\| \cos \vartheta_k \quad (4.119)$$

If $\hat{S}_k \cong \bar{S}$ is assumed, then substituting (4.116) into (4.119) gives

$$\|J_k h_k\|^2 + \|z_k^*\| \|h_k\| \cos \gamma_k = \alpha_k \|d_k\| \|h_k\| \cos \vartheta_k \quad (4.120)$$

For Hessian approximations \hat{H}_k and \dot{H}_k both d_k and $\cos \vartheta_k$ retain their original definitions so (4.92) and (4.93) in Lemma 1 and Lemma 2 are still valid in the MBFGS approach. Then the convergence analysis of the MBFGS method can be done analogously to the BFGS algorithm. To prove that the sequence θ_k generated by the MBFGS algorithm converges to θ^* when both Wolfe conditions are satisfied, it is necessary to prove that $\cos \vartheta_k$ is bounded away from zero so Lemma 2 is satisfied.

Using the trace equation analogous to that in [8] gives

$$Tr(\dot{H}_{k+1}) = Tr(J_{k+1}^T J_{k+1}) + Tr(\hat{S}_{k+1}) \quad (4.121)$$

Because the J_{k+1} is positive definite, $Tr(J_{k+1}^T J_{k+1})$ is a positive number. Therefore only $Tr(\hat{S}_{k+1})$ can reduce $Tr(\dot{H}_{k+1})$ and for simplicity only $Tr(\hat{S}_{k+1})$ is considered. The trace equation of \hat{S}_{k+1} using the MBFGS method is

$$Tr(\hat{S}_{k+1}) = Tr(\hat{S}_k) + \frac{\|z_k^*\|^2}{g_k^T h_k} - \frac{\|\hat{S}_k h_k\|^2}{h_k^T \hat{S}_k h_k} \quad (4.122)$$

Each of the terms in (4.122) is bounded and the following lemmas hold,

Lemma 6. (see proof on p. 100)

$$\frac{\|z_k^*\|^2}{g_k^T h_k} \leq w_3 \quad (4.123)$$

where w_3 is a constant.

Lemma 7. (see proof on p. 101)

$$\alpha_k \|d_k\| - \sigma_k \|h_k\| \leq \|\hat{S}_k h_k\| \leq \alpha_k \|d_k\| + \sigma_k \|h_k\| \quad (4.124)$$

where $\sigma_k \equiv \|J_k\|_F = \sqrt{\text{Tr}(J_k^T J_k)}$.

Lemma 8. (see proof on p. 103)

$$-\frac{\|\hat{S}_k h_k\|^2}{h_k^T \hat{S}_k h_k} \leq -\left(\frac{\alpha_k}{c_4 \cos \vartheta_k} - \sigma_k\right) \quad (4.125)$$

A similar convergence analysis of the MBFGS-QN algorithm can be done analogously to the BFGS-QN algorithm by using the trace equation in (4.122). Substituting the inequalities (4.123) and (4.125) into (4.122) yields

$$\text{Tr}(\hat{S}_{k+1}) \leq \text{Tr}(\hat{S}_k) + w_3 - \left(\frac{\alpha_k}{c_4 \cos \vartheta_k} - \sigma_k\right)$$

or

$$\text{Tr}(\hat{S}_{k+1}) \leq \text{Tr}(\hat{S}_k) + w_3 + \sigma_k - \frac{\alpha_k}{c_4 \cos \vartheta_k} \quad (4.126)$$

Since \hat{S}_{k+1} is assumed to be positive definite, then $\text{Tr}(\hat{S}_{k+1})$ is positive. The reduction of $\text{Tr}(\hat{S}_{k+1})$ is due to the fourth term in the right-hand side of (4.126) that is proportional to $\frac{\alpha_k}{\cos \vartheta_k}$. This term becomes large if (i) α_k is large, or (ii) $\cos \vartheta_k$ is very small. As a result, similar reasoning and analysis as in [8] are applied in this case in which the two scenarios are also considered. The first case assumes that the step length α_k is bounded away from zero and the $\cos \vartheta_k$ is arbitrarily small, thus $\text{Tr}(\hat{S}_k)$ is reduced since $\frac{\alpha_k}{c_4 \cos \vartheta_k}$ becomes dominant. However, from

$$\frac{\alpha_k}{c_4 \text{Tr}(\hat{H}_k)} \leq \cos \vartheta_k \quad [4.98]$$

it can be implied that $\text{Tr}(\hat{H}_k)$ becomes large if $\cos \vartheta_k$ tends to be arbitrarily small. Although a large value of $\text{Tr}(\hat{H}_k)$ may be a result of a large value of $\text{Tr}(J_{k+1}^T J_{k+1})$ even if $\text{Tr}(\hat{S}_k)$ is small, this is not the case being considered here since the MBFGS-QN algorithm is only utilized when the residual S_k is significant. Thus the value of $\text{Tr}(\hat{S}_k)$ is assumed at least to be as significant as of $\text{Tr}(J_{k+1}^T J_{k+1})$. Since $\text{Tr}(\hat{S}_k)$ varies due to the value of $\cos \vartheta_k$, (4.98) implies that an arbitrarily small value of $\cos \vartheta_k$ is somewhat a result of a large value of $\text{Tr}(\hat{S}_k)$. Hence, this phenomenon demonstrates the self-correcting property of the MBFGS approach when $\cos(\vartheta_k)$ tends to go to zero if α_k is assumed to be bounded away from zero.

The second case assumes that the step length α_k tends to zero. From Lemma 4 when α_k is close to zero, $\frac{h_k^T \hat{H}_k h_k}{\|h_k\|^2}$ becomes very small and so does the determinant of \hat{H}_k . Similar to the analysis in Section 4.5.1 that the trace equation (4.97) is no longer useful, it is necessary to carry on the study using the determinant of \hat{S}_{k+1} and

Lemma 9. (*see proof on p. 104*)

$$\det(\hat{S}_{k+1}) = \det(\hat{S}_k) \frac{(z_k^{*T} h_k)^2}{(g_k^T h_k) (h_k^T \hat{S}_k h_k)} \quad (4.127)$$

Lemma 9 leads to Lemma 10 and finally Lemma 11 in which the mean of α_k is proved to be bounded away from zero.

Lemma 10. (*see proof on p. 107*) *There exists a constant $c_5 > 0$ such that*

$$\prod_{i=1}^k \left(\frac{\alpha_i}{c_4} + \sigma_i \right) \geq c_5 \quad (4.128)$$

for all $k \geq 1$.

Lemma 11. (*see proof on p. 111*) *There exists a constant $c_6 > 0$ such that*

$$\prod_{i=1}^k \alpha_i \leq c_6^k \quad (4.129)$$

for all $k \geq 1$.

Since α_i is bounded away from zero as in Lemma 11, the only factor that can reduce $Tr(\hat{S}_{k+1})$ is due to $\cos \vartheta_k$. Equation (4.126) is rewritten as,

$$0 < Tr(\hat{S}_{k+1}) \leq Tr(\hat{S}_k) + w_3 + \sigma_k + \eta_k \frac{\alpha_k}{c_4} \quad (4.130)$$

where

$$\eta_k = -\frac{1}{\cos \vartheta_k} \quad (4.131)$$

If too many steps generate $\cos \vartheta_k \approx 0$, the last term on the right-hand side of (4.130) can make $Tr(\hat{S}_{k+1})$ negative. Hence, the assumption that $Tr(\hat{S}_{k+1})$ is positive definite is violated. Thus, $\cos \vartheta_k$ is required to be bounded away from zero to satisfy (4.130), i.e., the trace of \hat{S}_{k+1} is positive. Consequently, the following theorem is presented.

Theorem 4.5.2. *For the starting iterate θ_1 for which the objective function $F(\theta)$ satisfies Assumptions 1, 2, 3, and 4, then for any positive definite \hat{H}_1 , the MBFGS method (4.106)-(4.110) that satisfies the Wolfe conditions (4.81)-(4.82) generates θ_k that converge to θ^**

Proof. This proof can be done by contradiction analogous to the BFGS-QN approach presented in the previous section. As a result, the following steps attempt to show that the trace $Tr(\hat{S}_{k+1})$ remains positive for sufficiently large k even if $\{\cos \vartheta_k\}$ becomes arbitrarily small. If $\{\cos \vartheta_k\} \rightarrow 0$, then η_k gets close to $-\infty$. As a result, there is an index K_0 such that, for all $i > K_0$ then $\eta_i < -\frac{2w_3}{c_5^{1/k}}$. Then starting from $i = K_0$, (4.130) can be expressed as

$$0 < Tr(\hat{S}_{k+1}) \leq Tr(\hat{S}_{K_0}) + w_3(k+1-K_0) + \sum_{i=K_0}^k \sigma_i + \sum_{i=K_0}^k \eta_i \frac{\alpha_i}{c_4}$$

Substituting $\eta_i < -\frac{2w_3}{c_5^{1/k}}$ gives

$$0 < Tr(\hat{S}_{k+1}) \leq Tr(\hat{S}_{K_0}) + w_3(k+1-K_0) + \sum_{i=K_0}^k \sigma_i - \frac{2w_3}{c_5^{1/k}} \sum_{i=K_0}^k \frac{\alpha_i}{c_4} \quad (4.132)$$

Using the geometric/arithmetic mean inequality of (4.128) gives

$$c_5^{1/k} \leq \frac{1}{k} \sum_{i=1}^k \left(\frac{\alpha_i}{c_4} + \sigma_i \right)$$

and then rearranging,

$$kc_5^{1/k} - \sum_{i=1}^k \sigma_i \leq \sum_{i=1}^k \frac{\alpha_i}{c_4} \quad (4.133)$$

From (4.133), when $i = K_0$ then

$$kc_5^{1/k} - \sum_{i=1}^k \sigma_i - \sum_{i=1}^{K_0-1} \frac{\alpha_i}{c_4} \leq \sum_{i=K_0}^k \frac{\alpha_i}{c_4} \quad (4.134)$$

Substituting (4.134) into (4.132) gives

$$0 < Tr(\hat{S}_{k+1}) \leq Tr(\hat{S}_{K_0}) + w_3(k+1-K_0) + \sum_{i=K_0}^k \sigma_i - \frac{2w_3}{c_5^{1/k}} \left(kc_5^{1/k} - \sum_{i=1}^k \sigma_i - \sum_{i=1}^{K_0-1} \frac{\alpha_i}{c_4} \right)$$

or

$$0 < Tr(\hat{S}_{k+1}) \leq Tr(\hat{S}_{K_0}) + w_3(1-k) - w_3K_0 + \left(1 + \frac{2w_3}{c_5^{1/k}}\right) \sum_{i=K_0}^k \sigma_i + \frac{2w_3}{c_5^{1/k}} \sum_{i=1}^{K_0-1} \frac{\alpha_i}{c_4}$$

From the second term on the right-hand side, $Tr(\hat{S}_{k+1})$ becomes negative for sufficiently large k and thus contradicts the assumption that \hat{H}_k and \hat{S}_k are positive definite. Consequently, $\cos \vartheta_k$ is bounded away from zero. From (4.93) in Lemma 2,

$$F_{k+1} - F^* \leq [1 - c_1 m_1 c_3 \cos^2 \vartheta_k] (F_k - F^*) \quad [4.93]$$

it can be concluded that the sequence θ_k generated by using the MBFGS method converges to θ^* . \square

4.5.3 Proofs of Lemmas

Proof of Lemma 1. Into (4.90) is substituted the upper bound for $g_k^T h_k$ from (4.87) and $-d_k^T h_k$ from (4.91),

$$(1 - c_2) \|d_k\| \|h_k\| \cos \vartheta_k \leq m_2 \|h_k\|^2$$

cancelling $\|h_k\|$ from both sides and rearranging gives

$$c_3\|d_k\|\cos\vartheta_k \leq \|h_k\| \quad (4.135)$$

where $c_3 \equiv \frac{(1 - c_2)}{m_2} > 0$.

The upper bound of $\|h_k\|$ can be calculated by using a Taylor series approximation of the function $F(\theta)$ about θ_k ,

$$F(\theta + h_k) = F(\theta_k) + d_k^T h_k + \frac{1}{2} h_k^T \bar{H} h_k$$

or

$$F(\theta + h_k) - F(\theta_k) = d_k^T h_k + \frac{1}{2} h_k^T \bar{H} h_k$$

Using this in the first Wolfe condition (4.81) gives,

$$d_k^T h_k + \frac{1}{2} h_k^T \bar{H} h_k \leq c_1 d_k^T h_k$$

and then introducing the inequality (4.87) yields

$$\frac{1}{2} m_1 \|h_k\|^2 \leq -(1 - c_1) d_k^T h_k$$

into which $d_k^T h_k$ is substituted from (4.91),

$$\frac{1}{2} m_1 \|h_k\|^2 \leq (1 - c_1) \|d_k\| \|h_k\| \cos\vartheta_k$$

As a result, the upper bound of $\|h_k\|$ is

$$\|h_k\| \leq \frac{2(1 - c_1)}{m_1} \|d_k\| \cos\vartheta_k$$

or

$$\|h_k\| \leq c_4 \|d_k\| \cos\vartheta_k \quad (4.136)$$

where $c_4 \equiv \frac{2(1 - c_1)}{m_1} > 0$. From (4.135) and (4.136), $\|h_k\|$ is in a range of

$$c_3 \|d_k\| \cos\vartheta_k \leq \|h_k\| \leq c_4 \|d_k\| \cos\vartheta_k$$

□

Proof of Lemma 2. The proof is a review of Lemma (4.93) from [8]. From the first Wolfe condition (4.81),

$$F_{k+1} - F_k \leq c_1 \alpha_k d_k^T s_k$$

is substituted $\alpha_k s_k = h_k$ from (4.79),

$$F_{k+1} - F_k \leq c_1 d_k^T h_k$$

Introducing (4.91) yields,

$$F_{k+1} - F_k \leq -c_1 \|d_k\| \|h_k\| \cos \vartheta_k \quad (4.137)$$

and then substituting the lower bound of $\|h_k\|$ from the inequality in (4.92) gives

$$F_{k+1} - F_k \leq -c_1 c_3 \|d_k\|^2 \cos^2 \vartheta_k \quad (4.138)$$

Since the function $F(\theta)$ is convex on the level set D, then

$$\begin{aligned} F_k - F^* &\leq d_k^T (\theta_k - \theta^*) \\ &\leq \|d_k\| \|\theta_k - \theta^*\| \end{aligned} \quad (4.139)$$

At θ^* , $\bar{H}(\theta^*)$ satisfies the secant equation (4.86),

$$g_k^* = (\bar{H})(\theta^*)(\theta_k - \theta^*) \quad (4.140)$$

Since the gradient d_k can be expressed as

$$d_k = \bar{H}(\theta_k - \theta^*) \quad (4.141)$$

then transposing and multiplying (4.141) by $(\theta_k - \theta^*)$ gives

$$d_k^T (\theta_k - \theta^*) = (\theta_k - \theta^*)^T \bar{H}(\theta_k - \theta^*) \quad (4.142)$$

Equation (4.142) also satisfies (4.85), that is

$$m_1 \|\theta_k - \theta^*\|^2 \leq (\theta_k - \theta^*)^T \bar{H}(\theta_k - \theta^*) \leq m_2 \|\theta_k - \theta^*\|^2 \quad (4.143)$$

Taking the lower bound of (4.143),

$$m_1 \|\theta_k - \theta^*\|^2 \leq (\theta_k - \theta^*)^T d_k \quad (4.144)$$

the right hand side is expanded as

$$m_1 \|\theta_k - \theta^*\|^2 \leq \|\theta_k - \theta^*\| \|d_k\|$$

Cancelling $\|\theta_k - \theta^*\|$ from both sides,

$$\|\theta_k - \theta^*\| \leq \frac{1}{m_1} \|d_k\| \quad (4.145)$$

and substituting $\|\theta_k - \theta^*\|$ from (4.145) into (4.139) yields

$$\|d_k\|^2 \leq m_1 (F_k - F^*) \quad (4.146)$$

Substituting $\|d_k\|^2$ from (4.146) and adding $F_k - F^*$ to both sides of (4.138) finally yields,

$$F_{k+1} - F^* \leq [1 - c_1 c_3 m_1 \cos^2 \vartheta_k] (F_k - F^*)$$

□

Proof of Lemma 6. The quantity $\|z_k^*\|^2$ can be described as

$$\|z_k^*\|^2 = z_k^{*T} z_k^*$$

Introducing (4.113) gives,

$$\|z_k^*\|^2 = h_k^T \bar{S} \bar{S} h_k$$

or

$$= h_k^T \bar{S}^{1/2} \bar{S} \bar{S}^{1/2} h_k \quad (4.147)$$

Denoting

$$\acute{b}_k \equiv \bar{S}^{1/2} h_k \quad (4.148)$$

where $\bar{S}^{1/2}\bar{S}^{1/2} = \bar{S}_k$. Then substituting (4.148) into (4.147) gives

$$\|z_k^*\|^2 = \hat{b}_k^T \bar{S} \hat{b}_k$$

Since $\hat{b}_k^T \bar{S} \hat{b}_k$ satisfies (4.112), it also follows (4.114). Thus the upper bound of $\frac{\|z_k^*\|^2}{g_k^T h_k}$ can be calculated by substituting the upper bound of $z_k^{*T} h_k$ from (4.114) and the lower bound of $g_k^T h_k$ from (4.87) as,

$$\begin{aligned} \frac{\|z_k^*\|^2}{g_k^T h_k} &\leq \frac{w_2 \|h_k\|^2}{m_1 \|h_k\|^2} \\ &\leq w_3 \end{aligned}$$

where $w_3 = \frac{w_2}{m_1}$. □

Proof of Lemma 7. Into

$$\hat{H}_k h_k = -\alpha_k d_k \quad [4.117]$$

substitute $\hat{H}_k = J_k^T J_k + \hat{S}_k$ to give,

$$\hat{S}_k h_k = -(\alpha_k d_k + J_k^T J_k h_k) \quad (4.149)$$

and then

$$\|\hat{S}_k h_k\| = \|\alpha_k d_k + J_k^T J_k h_k\|$$

Using the *triangle inequality* for the right-hand side gives

$$\|\hat{S}_k h_k\| \leq \alpha_k \|d_k\| + \|J_k^T J_k h_k\| \quad (4.150)$$

The lower bound of $\|\hat{S}_k h_k\|$ can be calculated from

$$\hat{H}_k = J_k^T J_k + \hat{S}_k \quad (4.151)$$

Multiplying by h_k on both sides and rearranging the equation gives

$$\hat{S}_k h_k = \hat{H}_k h_k - (J_k^T J_k) h_k \quad (4.152)$$

so

$$\|\hat{S}_k h_k\| = \|\dot{H}_k h_k - (J_k^T J_k) h_k\| \quad (4.153)$$

According to the *reverse triangle inequality* the right-hand side of (4.153) is

$$\left| \|\dot{H}_k h_k\| - \|(J_k^T J_k) h_k\| \right| \leq \|\dot{H}_k h_k - (J_k^T J_k) h_k\| \quad (4.154)$$

Since $(J_k^T J_k) h_k \leq \dot{H}_k h_k$, then

$$\|\dot{H}_k h_k\| - \|(J_k^T J_k) h_k\| \leq \|\dot{H}_k h_k - (J_k^T J_k) h_k\| \quad (4.155)$$

Substituting (4.155) into (4.153) gives

$$\|\dot{H}_k h_k\| - \|(J_k^T J_k) h_k\| \leq \|\hat{S}_k h_k\| \quad (4.156)$$

Replacing $\dot{H}_k h_k$ by $-\alpha_k d_k$ in (4.117) yields

$$\|-\alpha_k d_k\| - \|(J_k^T J_k) h_k\| \leq \|\hat{S}_k h_k\|$$

or

$$\alpha_k \|d_k\| - \|(J_k^T J_k) h_k\| \leq \|\hat{S}_k h_k\| \quad (4.157)$$

since $\alpha_k > 0$. Combining (4.150) and (4.157) gives,

$$\alpha_k \|d_k\| - \|(J_k^T J_k) h_k\| \leq \|\hat{S}_k h_k\| \leq \alpha_k \|d_k\| + \|J_k^T J_k h_k\| \quad (4.158)$$

Since J_k is positive definite then

$$\|J_k\|_F = \sqrt{\text{Tr}(J_k^T J_k)} \equiv \sigma_k^{1/2} \quad (4.159)$$

where $\sigma_k > 0$. Hence

$$\|J_k^T J_k\|_F = \text{Tr}(J_k^T J_k) = \sigma_k \quad (4.160)$$

for all k . $\|J_k^T J_k h_k\|$ satisfies the following inequality.

$$\begin{aligned} \|(J_k^T J_k) h_k\| &\leq \|J_k^T J_k\|_F \|h_k\| \\ &\leq \sigma_k \|h_k\| \end{aligned} \quad (4.161)$$

Substituting (4.161) into (4.158) yields,

$$\alpha_k \|d_k\| - \sigma_k \|h_k\| \leq \|\hat{S}_k h_k\| \leq \alpha_k \|d_k\| + \sigma_k \|h_k\|$$

□

Proof of Lemma 8. Since \hat{S}_k is assumed to be positive, $h_k^T \hat{S}_k h_k > 0$, then applying the matrix norms inequality gives

$$h_k^T \hat{S}_k h_k \leq \|\hat{S}_k h_k\| \|h_k\| \quad (4.162)$$

As a result, substituting (4.162) into the denominator of $\frac{\|\hat{S}_k h_k\|^2}{h_k^T \hat{S}_k h_k}$ yields

$$\frac{\|\hat{S}_k h_k\|^2}{\|\hat{S}_k h_k\| \|h_k\|} \leq \frac{\|\hat{S}_k h_k\|^2}{h_k^T \hat{S}_k h_k} \quad (4.163)$$

Cancelling $\|\hat{S}_k h_k\|$ in the numerator and the denominator on the right-hand side of (4.163) gives

$$\frac{\|\hat{S}_k h_k\|}{\|h_k\|} \leq \frac{\|\hat{S}_k h_k\|^2}{h_k^T \hat{S}_k h_k} \quad (4.164)$$

The lower bound of $\|\hat{S}_k h_k\|$ in (4.124) is equivalent to

$$\frac{\alpha_k \|d_k\| - \sigma_k \|h_k\|}{\|h_k\|} \leq \frac{\|\hat{S}_k h_k\|}{\|h_k\|} \quad (4.165)$$

From the inequality (4.92), $\|d_k\|$ can be expressed as

$$\frac{\|h_k\|}{c_4 \cos \vartheta_k} \leq \|d_k\| \quad (4.166)$$

Substituting (4.166) into (4.165) yields

$$\frac{\alpha_k \frac{\|h_k\|}{c_4 \cos \vartheta_k} - \sigma_k \|h_k\|}{\|h_k\|} \leq \frac{\|\hat{S}_k h_k\|}{\|h_k\|}$$

Factoring out $\|h_k\|$ and canceling this term in the numerator and the denominator gives

$$\frac{\alpha_k}{c_4 \cos \vartheta_k} - \sigma_k \leq \frac{\|\hat{S}_k h_k\|}{\|h_k\|} \quad (4.167)$$

From (4.164),

$$\frac{\alpha_k}{c_4 \cos \vartheta_k} - \sigma_k \leq \frac{\|\hat{S}_k h_k\|^2}{h_k^T \hat{S}_k h_k}$$

□

Proof of Lemma 9. The determinant of \hat{S}_{k+1} is

$$\det(\hat{S}_{k+1}) = \det\left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} - \frac{\hat{S}_k h_k h_k^T \hat{S}_k}{h_k^T \hat{S}_k h_k}\right) \quad (4.168)$$

From Sylvester's determinant theorem [31], it is stated that for any $n \times m$ matrix A , $m \times n$ matrix B , and invertible $n \times n$ matrix X ,

$$\det(X + AB) = \det(X) \det(I + BX^{-1}A) \quad (4.169)$$

where I is the $n \times n$ identity matrix. It is also true that for any column vector u and row vector v^T (each with n components),

$$\det(X + uv^T) = \det(X) (1 + v^T X^{-1} u) \quad (4.170)$$

As a result, (4.168) is rearranged into the form

$$\det(\hat{S}_{k+1}) = \det\left[\left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k}\right) - \frac{\hat{S}_k h_k h_k^T \hat{S}_k}{h_k^T \hat{S}_k h_k}\right] \quad (4.171)$$

Analogous to (4.170)

$$X \equiv \left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k}\right) \quad (4.172)$$

$$u \equiv \frac{\hat{S}_k h_k}{(h_k^T \hat{S}_k h_k)^{1/2}} \quad (4.173)$$

$$v^T \equiv \frac{(\hat{S}_k h_k)^T}{(h_k^T \hat{S}_k h_k)^{1/2}} \quad (4.174)$$

So applying Sylvester's determinant theorem (4.170) gives

$$\begin{aligned} \det(\hat{S}_{k+1}) &= \det \left[\left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right) - \frac{\hat{S}_k h_k h_k^T \hat{S}_k}{h_k^T \hat{S}_k h_k} \right] \\ &= \det \left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right) \left[1 - \frac{\left(\hat{S}_k h_k \right)^T}{\left(h_k^T \hat{S}_k h_k \right)^{1/2}} \left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right)^{-1} \frac{\hat{S}_k h_k}{\left(h_k^T \hat{S}_k h_k \right)^{1/2}} \right] \end{aligned} \quad (4.175)$$

The first parentheses on the right-hand side of (4.175) is simplified by applying Sylvester's determinant theorem (4.170) as

$$\det \left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right) = \det(\hat{S}_k) \left(1 + \frac{z_k^{*T} \hat{S}_k^{-1} z_k^*}{g_k^T h_k} \right) \quad (4.176)$$

Next the quantities inside the square bracket are simplified. In order to do so, $\left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right)^{-1}$ is required. This can be calculated using the *Sherman-Morrison formula* for calculating the sum of an invertible matrix X and uv^T (which is called the *dyadic product*) where u is a column vector and v^T is a row vector and $1+v^T X^{-1} u \neq 0$.

The Sherman-Morrison formula [2, 68] states that

$$(X + uv^T)^{-1} = X^{-1} - \frac{X^{-1}uv^T X^{-1}}{1 + v^T X^{-1} u} \quad (4.177)$$

Since $\left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right)^{-1}$ is in a form analogous to $(X + uv^T)^{-1}$ where in this case

$$\begin{aligned} X &\equiv \hat{S}_k \\ u &\equiv \frac{z_k^*}{(g_k^T h_k)^{1/2}} \\ v^T &\equiv \frac{z_k^{*T}}{(g_k^T h_k)^{1/2}} \end{aligned}$$

then applying the Sherman-Morrison formula gives

$$\left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right)^{-1} = \hat{S}_k^{-1} - \frac{\hat{S}_k^{-1} z_k^* z_k^{*T} \hat{S}_k^{-1}}{(g_k^T h_k)^{1/2} \left(1 + \frac{z_k^{*T} \hat{S}_k^{-1} z_k^*}{(g_k^T h_k)^{1/2} (g_k^T h_k)^{1/2}} \right) (g_k^T h_k)^{1/2}}$$

which reduces to

$$\left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right)^{-1} = \hat{S}_k^{-1} - \frac{\hat{S}_k^{-1} z_k^* z_k^{*T} \hat{S}_k^{-1}}{\left(g_k^T h_k + z_k^{*T} \hat{S}_k^{-1} z_k^* \right)} \quad (4.178)$$

To simplify the square bracket in (4.175) substituting $\left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right)^{-1}$ from (4.178) in the square bracket gives

$$\begin{aligned} & \left[1 - \frac{\left(\hat{S}_k h_k \right)^T}{\left(h_k^T \hat{S}_k h_k \right)^{1/2}} \left(\hat{S}_k + \frac{z_k^* z_k^{*T}}{g_k^T h_k} \right)^{-1} \frac{\hat{S}_k h_k}{\left(h_k^T \hat{S}_k h_k \right)^{1/2}} \right] \\ &= 1 - \frac{\left(\hat{S}_k h_k \right)^T}{\left(h_k^T \hat{S}_k h_k \right)^{1/2}} \left\{ \hat{S}_k^{-1} - \frac{\hat{S}_k^{-1} z_k^* z_k^{*T} \hat{S}_k^{-1}}{\left(g_k^T h_k + z_k^{*T} \hat{S}_k^{-1} z_k^* \right)} \right\} \frac{\hat{S}_k h_k}{\left(h_k^T \hat{S}_k h_k \right)^{1/2}} \\ &= 1 - \left(\frac{h_k^T \hat{S}_k \hat{S}_k^{-1} \hat{S}_k h_k}{h_k^T \hat{S}_k h_k} \right) + \frac{h_k^T \hat{S}_k \hat{S}_k^{-1} z_k^* z_k^{*T} \hat{S}_k^{-1} \hat{S}_k h_k}{\left(g_k^T h_k + z_k^{*T} \hat{S}_k^{-1} z_k^* \right) \left(h_k^T \hat{S}_k h_k \right)} \\ &= 1 - \left(\frac{h_k^T \hat{S}_k h_k}{h_k^T \hat{S}_k h_k} \right) + \frac{h_k^T z_k^* z_k^{*T} h_k}{\left(g_k^T h_k + z_k^{*T} \hat{S}_k^{-1} z_k^* \right) \left(h_k^T \hat{S}_k h_k \right)} \\ &= \frac{h_k^T z_k^* z_k^{*T} h_k}{\left(g_k^T h_k + z_k^{*T} \hat{S}_k^{-1} z_k^* \right) \left(h_k^T \hat{S}_k h_k \right)} \end{aligned} \quad (4.179)$$

Substituting (4.176) and (4.179) into (4.175) gives

$$\det(\hat{S}_{k+1}) = \det(\hat{S}_k) \left(1 + \frac{z_k^{*T} \hat{S}_k^{-1} z_k^*}{g_k^T h_k} \right) \frac{h_k^T z_k^* z_k^{*T} h_k}{\left(g_k^T h_k + z_k^{*T} \hat{S}_k^{-1} z_k^* \right) \left(h_k^T \hat{S}_k h_k \right)}$$

Expanding the second parentheses in the right-hand side of the equation gives,

$$\begin{aligned} \det(\hat{S}_{k+1}) &= \det(\hat{S}_k) \left(\frac{g_k^T h_k + z_k^{*T} \hat{S}_k^{-1} z_k^*}{g_k^T h_k} \right) \frac{h_k^T z_k^* z_k^{*T} h_k}{\left(g_k^T h_k + z_k^{*T} \hat{S}_k^{-1} z_k^* \right) \left(h_k^T \hat{S}_k h_k \right)} \\ &= \det(\hat{S}_k) \frac{h_k^T z_k^* z_k^{*T} h_k}{\left(g_k^T h_k \right) \left(h_k^T \hat{S}_k h_k \right)} \\ &= \det(\hat{S}_k) \frac{\left(z_k^* h_k \right)^2}{\left(g_k^T h_k \right) \left(h_k^T \hat{S}_k h_k \right)} \end{aligned}$$

which establishes (4.127). \square

Proof of Lemma 10. First it is proved that $\text{Tr}(\hat{S}_{k+1})$ is bounded. From

$$\frac{\alpha_k}{c_4 \cos \vartheta_k} - \sigma_k \leq \frac{\|\hat{S}_k h_k\|}{\|h_k\|} \quad [4.167]$$

using the matrix norm inequality, the left-hand side of (4.167) is

$$\begin{aligned} \frac{\|\hat{S}_k h_k\|}{\|h_k\|} &\leq \frac{\|\hat{S}_k\|_F \|h_k\|}{\|h_k\|} \\ &\leq \|\hat{S}_k\|_F \end{aligned} \quad (4.180)$$

In fact $\|\hat{S}_k\|_F$ in (4.180) can be related to $\text{Tr}(\hat{S}_k)$. The trace norm of a matrix X (in this case $X \in \mathbb{R}^{n \times n}$) is defined as

$$\|X\|_{tr} \equiv \sum_{i=1}^n |\Lambda_i| \quad (4.181)$$

where $|\Lambda_i|$ is the absolute value of the i^{th} eigenvalue of X . In [65] it is proved that

$$\|X\|_F \leq \|X\|_{tr} \leq \sqrt{\text{rank}(X)} \|X\|_F \quad (4.182)$$

where $\|\cdot\|_F$ is the Frobenius norm of a matrix. Since \hat{S}_k is assumed to be positive definite, all its eigenvalues are positive. Thus $\|\hat{S}_k\|_{tr}$ is simply $\text{Tr}(\hat{S}_k)$. As a result, (4.180) becomes

$$\frac{\|\hat{S}_k h_k\|}{\|h_k\|} \leq \|\hat{S}_k\|_F \leq \text{Tr}(\hat{S}_k) \quad (4.183)$$

The substitution of (4.183) into (4.167) yields

$$\frac{\alpha_k}{c_4 \cos \vartheta_k} - \sigma_k \leq \text{Tr}(\hat{S}_k)$$

or,

$$\frac{\alpha_k}{c_4 \cos \vartheta_k} \leq \text{Tr}(\hat{S}_k) + \sigma_k \quad (4.184)$$

From the trace inequality (4.126),

$$\text{Tr}(\hat{S}_{k+1}) \leq \text{Tr}(\hat{S}_k) + w_3 + \sigma_k - \frac{\alpha_k}{c_4 \cos \vartheta_k} \quad [4.126]$$

and substituting in (4.184) gives

$$Tr(\hat{S}_{k+1}) \leq Tr(\hat{S}_k) + w_3 + \sigma_k - (Tr(\hat{S}_k) + \sigma_k)$$

which is just

$$Tr(\hat{S}_{k+1}) \leq w_3 \quad [4.185]$$

To prove Lemma 10 it is necessary to find the lower bound of $\det(\hat{S}_{k+1})$. From Lemma 9,

$$\det(\hat{S}_{k+1}) = \det(\hat{S}_k) \frac{(z_k^{*T} h_k)^2}{(g_k^T h_k) (h_k^T \hat{S}_k h_k)} \quad [4.127]$$

The lower bound of $\det(\hat{S}_{k+1})$ is established by taking the minimum value of $z_k^{*T} h_k$ and the maximum values of $g_k^T h_k$ and $h_k^T \hat{S}_k h_k$. The lower bound of $z_k^{*T} h_k$ is

$$w_1 \|h_k\|^2 \leq z_k^{*T} h_k \quad [4.114]$$

and the upper bound of $g_k^T h_k$ from (4.87) gives

$$g_k^T h_k \leq m_2 \|h_k\|^2 \quad [4.87]$$

The upper bound of $h_k^T \hat{S}_k h_k$ can be obtained by first recalling

$$h_k^T \hat{S}_k h_k \leq \|\hat{S}_k h_k\| \|h_k\| \quad [4.162]$$

where the upper bound of $\|\hat{S}_k h_k\|$ is

$$\|\hat{S}_k h_k\| \leq \alpha_k \|d_k\| + \sigma_k \|h_k\| \quad [4.124]$$

Substituting the upper bound of $\|\hat{S}_k h_k\|$ into (4.162) yields

$$h_k^T \hat{S}_k h_k \leq (\alpha_k \|d_k\| + \sigma_k \|h_k\|) \|h_k\| \quad (4.186)$$

Since $\|d_k\|$ can be expressed as

$$\frac{\|h_k\|}{c_4 \cos \vartheta_k} \leq \|d_k\| \quad [4.166]$$

Substituting (4.166) (4.186) gives

$$h_k^T \hat{S}_k h_k \leq \left(\alpha_k \frac{\|h_k\|}{c_4 \cos \vartheta_k} + \sigma_k \|h_k\| \right) \|h_k\|$$

or

$$h_k^T \hat{S}_k h_k \leq \alpha_k \frac{\|h_k\|^2}{c_4 \cos \vartheta_k} + \sigma_k \|h_k\|^2 \quad (4.187)$$

Finally substituting (4.87), (4.114), and (4.187) into (4.127) yields

$$\det(\hat{S}_k) \frac{w_1^2 \|h_k\|^4}{m_2 \|h_k\|^2 \left(\frac{\alpha_k \|h_k\|^2}{c_4} + \sigma_k \|h_k\|^2 \right)} \leq \det(\hat{S}_{k+1})$$

which reduces to

$$\det(\hat{S}_k) \frac{m_3}{\left(\frac{\alpha_k}{c_4} + \sigma_k \right)} \leq \det(\hat{S}_{k+1}) \quad (4.188)$$

where $m_3 = \frac{w_1^2}{m_2}$.

Using (4.188) to calculate $\det(\hat{S}_2)$ from $\det(\hat{S}_1)$ gives,

$$\det(\hat{S}_1) \frac{m_3}{\left(\frac{\alpha_1}{c_4} + \sigma_1 \right)} \leq \det(\hat{S}_2)$$

A similar calculation can be done for $\det(\hat{S}_3)$,

$$\det(\hat{S}_2) \frac{m_3}{\left(\frac{\alpha_2}{c_4} + \sigma_2 \right)} \leq \det(\hat{S}_3)$$

Substituting $\det(\hat{S}_2)$ to the above equation gives

$$\det(\hat{S}_1) \left[\frac{m_3}{\left(\frac{\alpha_1}{c_4} + \sigma_1 \right)} \right] \left[\frac{m_3}{\left(\frac{\alpha_2}{c_4} + \sigma_2 \right)} \right] \leq \det(\hat{S}_3)$$

So for the k iteration,

$$\det(\hat{S}_1) \prod_{i=1}^k \left(\frac{m_3}{\frac{\alpha_i}{c_4} + \sigma_i} \right) \leq \det(\hat{S}_{k+1}) \quad (4.189)$$

The geometric/arithmetic mean inequality [73] states that for n nonnegative numbers x_1, x_2, \dots, x_n

$$\sqrt[n]{x_1 \cdot x_2 \cdots x_n} \leq \frac{x_1 + x_2 + \cdots + x_n}{n}$$

Since a determinant of a matrix is the product of all its eigenvalues, using the geometric/arithmetic mean inequality yields

$$\det(\hat{S}_{k+1}) \leq \left[\frac{\text{Tr}(\hat{S}_{k+1})}{n} \right]^n$$

Substituting the inequality (4.185) for $\text{Tr}(\hat{S}_{k+1})$ yields,

$$\det(\hat{S}_{k+1}) \leq \left[\frac{w_3}{n} \right]^n \quad (4.190)$$

Substituting (4.189) into (4.190) yields

$$\det(\hat{S}_1) \prod_{i=1}^k \left(\frac{m_3}{\frac{\alpha_i}{c_4} + \sigma_i} \right) \leq \left[\frac{w_3}{n} \right]^n$$

The quantity m_3 in the left-hand side can be factored out,

$$(m_3)^k \det(\hat{S}_1) \prod_{i=1}^k \left(\frac{1}{\frac{\alpha_i}{c_4} + \sigma_i} \right) \leq \frac{w_3^n}{n^n}$$

so rearranging gives

$$\prod_{i=1}^k \left(\frac{1}{\frac{\alpha_i}{c_4} + \sigma_i} \right) \leq \frac{w_3^n}{n^n m_3^k \det(\hat{S}_1)} \quad (4.191)$$

Because the left hand side of (4.191) is lesser or equal to a constant, then its reciprocal implies that there is a positive constant c_5 such that

$$c_5 \leq \prod_{i=1}^k \left(\frac{\alpha_i}{c_4} + \sigma_i \right)$$

for all $k \geq 1$ since α_i , c_4 , and σ_i are positive. \square

Proof of Lemma 11. Using the matrix norm inequality, $\frac{\|\acute{H}_k h_k\|}{\|h_k\|}$ is described as

$$\begin{aligned}\frac{\|\acute{H}_k h_k\|}{\|h_k\|} &\leq \frac{\|\acute{H}_k\|_F \|h_k\|}{\|h_k\|} \\ &\leq \|\acute{H}_k\|_F\end{aligned}\quad (4.192)$$

From (4.182) $\|\acute{H}_k\|_F$ in (4.192) relates to $Tr(\acute{H}_k)$ as

$$\|\acute{H}\|_F \leq \|\acute{H}\|_{tr} \leq \sqrt{\text{rank}(\acute{H})} \|\acute{H}\|_F \quad (4.193)$$

In analogy to (4.183), (4.192) relates to $Tr(\acute{H}_k)$ as

$$\frac{\|\acute{H}_k h_k\|}{\|h_k\|} \leq \|\acute{H}_k\| \leq Tr(\acute{H}_k) \quad (4.194)$$

From Lemma 4 the upper bound of α_k is

$$\alpha_k \leq c_4 \frac{h_k^T \acute{H}_k h_k}{\|h_k\|^2} \quad [4.99]$$

then applying the matrix norm inequality gives

$$\begin{aligned}\alpha_k &\leq c_4 \frac{\|\acute{H}_k h_k\| \|h_k\|}{\|h_k\|^2} \\ &\leq c_4 \frac{\|\acute{H}_k h_k\|}{\|h_k\|}\end{aligned}$$

Substituting (4.194) yields

$$\alpha_k \leq c_4 Tr(\acute{H}_k) \equiv c_6 \quad (4.195)$$

where c_6 is a constant. Thus the sum of α_i is

$$\sum_{i=1}^k \alpha_i \leq \sum_{i=1}^k c_6 = kc_6$$

and rearranging gives

$$\sum_{i=1}^k \frac{\alpha_i}{k} \leq c_6 \quad (4.196)$$

Applying the geometric/arithmetic mean inequality to the left side of (4.196),

$$\prod_{i=1}^k \alpha_i \leq \left(\sum_{i=1}^k \frac{\alpha_i}{k} \right)^k \leq c_6^k$$

or

$$\prod_{i=1}^k \alpha_i \leq c_6^k$$

□

4.6 Rate of Convergence of the MBFGS method

In this section the linear convergence analysis analogous to the treatment presented in [8] for the linear convergence of the MBFGS method is presented. Byrd et al. [8] applies Theorem 6.4 of Dennis and Moré [16] to conclude superlinear convergence of the BFGS-QN approach if admissible $\alpha_k = 1$ is employed for all sufficiently large k . Due to the fact that the MBFGS-QN method yields similar linear convergence to the BFGS-QN approach, it is reasonable to expect that the MBFGS-QN method analogously offers superlinear convergence for all sufficiently large k if admissible $\alpha_k = 1$ is applied.

The linear convergence analysis of the MBFGS-QN method can be induced from (4.93) in Lemma 2 as

Lemma 12. *There exists a constant $0 \leq c_8 < 1$ such that*

$$F_{k+1} - F^* \leq c_8^k (F_1 - F^*) \quad (4.197)$$

for all sufficiently large k .

Proof. Recall

$$0 < Tr(\hat{S}_{k+1}) \leq Tr(\hat{S}_k) + w_3 + \sigma_k - \frac{\alpha_k}{c_4 \cos \vartheta_k} \quad [4.126]$$

Starting from $Tr(\hat{S}_1)$, (4.126) can be expressed as

$$0 < Tr(\hat{S}_{k+1}) \leq Tr(\hat{S}_1) + w_3 k + \sum_{i=1}^k \sigma_i - \frac{1}{c_4} \sum_{i=1}^k \frac{\alpha_i}{\cos \vartheta_i}$$

As a result

$$\sum_{i=1}^k \frac{\alpha_i}{\cos \vartheta_i} \leq c_7 k \quad (4.198)$$

From the geometric/arithmetic mean equality, (4.198) can be rewritten as

$$\prod_{i=1}^k \frac{\alpha_i}{\cos \vartheta_i} \leq c_7^k \quad (4.199)$$

From Lemma 11, the upper bound of $\prod_{i=1}^k \alpha_i$ is

$$\prod_{i=1}^k \alpha_i \leq c_6^k \quad [4.129]$$

Then substituting (4.129) into (4.199) yields

$$\left(\frac{c_6}{c_7} \right)^k \leq \prod_{i=1}^k \cos \vartheta_i \quad (4.200)$$

Starting from the objective function F_1 , (4.93) is expressed as

$$F_{k+1} - F^* \leq \prod_{i=1}^k (1 - c_1 m_1 c_3 \cos^2 \vartheta_i) (F_1 - F^*) \quad (4.201)$$

Applying the geometric/arithmetic mean inequality gives

$$F_{k+1} - F^* \leq \left[\frac{1}{k} \sum_{i=1}^k (1 - c_1 m_1 c_3 \cos^2 \vartheta_i) \right]^k (F_1 - F^*)$$

Using the geometric/arithmetic mean inequality again,

$$F_{k+1} - F^* \leq \left[1 - c_1 m_1 c_3 \left(\prod_{i=1}^k \cos^2 \vartheta_i \right)^{1/k} \right]^k (F_1 - F^*) \quad (4.202)$$

Substituting in (4.200) yields

$$F_{k+1} - F^* \leq c_8^k (F_1 - F^*)$$

where

$$c_8 = \left[1 - c_1 m_1 c_3 \left(\frac{c_6}{c_7} \right)^{1/k} \right]$$

□

Since the linear convergence analysis of the MBFGS-QN algorithm yields results analogous to the BFGS-QN approach [8], it can be reasonably hypothesized that the same reasoning and proof for the superlinear convergence presented in [8, 16, 28] can be directly applied to the MBFGS-QN method.

In [8] Broyden's class formula is proved to yield superlinear convergence if the following additional assumption is applied,

Assumption 5. *The Hessian matrix H is Hölder continuous at θ^* in which there exists nonnegative real constants p, L such that*

$$\|H(\theta) - H(\theta^*)\| \leq L\|\theta - \theta^*\|^p \quad (4.203)$$

for all θ in a neighborhood of θ^*

Hence, [8] proved in their Theorem 4.1 that if the BFGS-QN algorithm, including Broyden's class formula when $\kappa \in [0, 1]$, satisfies (4.81)-(4.82) and employs $\alpha_k = 1$ whenever it is permissible, i.e., satisfying (4.81)-(4.82) when Assumption 1,2 and (4.203) hold, it produces the sequence θ_k that q -superlinearly converges to θ^* . The analysis in [8] further implies that

$$\lim_{k \rightarrow \infty} \frac{\|(\hat{H}_k - H^*)h_k\|}{\|h_k\|} = 0 \quad (4.204)$$

which is the proposition proved by Griewank and Toint [28]. Byrd et al. [8] used Theorem 6.4 of Dennis and Moré [16] to conclude that $\alpha_k = 1$ "is admissible for all sufficiently large k and that the rate of convergence is superlinear."

Because the convergence analysis of the MBFGS-QN algorithm is analogously derived and yields results similar to the BFGS-QN method, it is reasonably assumed that the MBFGS-QN algorithm, which satisfies Assumptions 1, 2, 3, 4, and 5 as well as the Wolfe conditions (4.81)-(4.82), is expected to generate the sequence θ_k that superlinearly converges to θ^* . Therefore, it is rational to hypothesize that the proposition (4.204) proved by Griewank and Toint [28] can be directly implied and

the conclusion of Byrd et al. [8] using Theorem 6.4 of Dennis and Moré [16] can be analogously applied to the MBFGS-QN algorithm. Consequently, it is hypothesized that if the MBFGS-QN algorithm produces the sequence θ_k of which $\alpha_k = 1$ is employed for sufficiently large k , the rate of convergence is reasonably expected to be superlinear. Since the proof of the superlinear convergence is too involved, it is not included in this analysis. However, the details of aforementioned theorems can be found in [16, 28].

4.7 *Switching MBFGS-DB Algorithm*

Although the developed algorithms are motivated to improve the performance of a quasi-Gauss-Newton method on the large residual case, as the procedure proceeds the residual S should ideally converge to zero or at least become relatively small for which the quasi-Gauss-Newton method should perform well. Consequently, it is not always necessary to include \hat{S}_k into the Hessian \hat{H}_k . Due to the fact that all of the proposed methods (the approaches presented in Sections 4.4.1-4.4.3, NL2SOL in [17, 20], and the algorithm proposed in [25]) recursively update \hat{S}_k from \hat{S}_{k-1} , the approximation of \hat{S}_k may not go to zero even if f_k and z_{k-1}^* become zero. For this reason NL2SOL [17, 20] sometimes applies the Gauss-Newton method for small final residuals while using their novel algorithm for approximating \hat{S}_k when the residual is large since a quasi-Gauss-Newton method tends to work well for the zero- or small-residual case [17, 19]. It is suggested in [17, 20] to multiply a *scaling factor* ς to \hat{S}_{k-1} before each update so that the approximated \hat{S}_k accurately accommodates to the small- or zero-residual case. This scaling method is inspired by direct modification of the *self-scaling* technique presented in [51] so that the approximation of \hat{S}_k is updated from $\varsigma\hat{S}_{k-1}$ rather than \hat{S}_{k-1} . The scaling factor ς is calculated as

$$\varsigma = \min \left\{ \frac{z_{k-1}^{*T} h_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}}, 1 \right\} \quad (4.205)$$

A similar idea is introduced by Nazareth [48] as reviewed in Section 4.2.1 in which a

hybrid weighted average between the quasi-Newton and the Gauss-Newton directions is proposed through controlling a weighting parameter ϕ as

$$\left[\phi_k J_k^T J_k + (1 - \phi_k) \hat{H}_k \right] h_k = -J_k^T f_k \quad [4.16]$$

where $0 \leq \phi_k \leq 1$ is adaptively chosen according to how well the Gauss-Newton model can be trusted.

Furthermore, Nazareth [49] also recommended using a similar hybrid as in (4.16) for the DGW Hessian approximation (the earlier development of NL2SOL) as

$$J_k^T J_k + (1 - \phi_k) S_k \quad (4.206)$$

where the update of S_k satisfies the following secant equation

$$\begin{aligned} S_k h_{k-1} &= z_{k-1}^* \\ z_{k-1}^* &= J_k^T f_k - J_{k-1}^T f_k \end{aligned}$$

Although the details of this idea are not presented in [49], it is assumed that the selection of ϕ_k is the same as mentioned in (4.16).

To have the improved capability of the proposed algorithms in the large-residual case yet attaining fast convergence of the quasi-Gauss Newton method when the residual becomes relatively small, a similar hybrid method to those presented in [48, 49] is developed. This development is motivated by observations made during the comparative simulations between the DGN-PBM algorithm in Chapter 3 and the algorithms in Section 4.4.1-4.4.3. It is observed that the proposed algorithms, especially the MBFGS algorithm, yield significant improvement over the DGN-PBM algorithm by offering faster convergence if the initial error is substantial. Furthermore, the trajectory of the EE motion moving from its initial configuration to reach the steady-state tracking configuration is more direct compared to the quasi-Gauss-Newton method (the DGN-PBM algorithm). However, the RMS tracking error during the steady state is significant. If the initial error between the EE initial configuration and the target

is small, the DGN-PBM algorithm generates steps that rapidly converge to the target with higher reliability and outperforms the MBFGS method, which in the worse case diverges. Moreover, during steady-state tracking the DGN-PBM algorithm always performs better than the MBFGS algorithm.

Instability occurs in simulations when utilizing the MBFGS algorithm for the zero- or small-residual problem may be explained through the fact that the approximation of \hat{S}_k is included into the Hessian even though the actual residual term is zero or become small. Because the update residual \hat{S}_k is recursively calculated from \hat{S}_{k-1} , it may not go to zero even if f_k becomes zero. Furthermore, since the uncalibrated visual servoing application assumes no knowledge about the robot and camera models, the approximation of the residual \hat{S} is done using the approximated Jacobian \hat{J} which is obtained from the dynamic Broyden estimator. Hence the estimated Jacobian does not represent the actual Jacobian matrix but in some sense only a partial Jacobian since it is only able to update the Jacobian in the “direction” of the trajectory. Thus the updated residual \hat{S} only partially represents the actual residual S term. As a result, redundant inclusion of the residual term may significantly deteriorate tracking performance in the small image error norm case.

These observed results agree with Dennis et al. [17] noting that the Gauss-Newton method performs better than the inclusion of the residual approximation into the Hessian due to inadequate capability of the \hat{S}_k update to converge to zero for the zero- or small-residual problems. Thus the approximation of the residual \hat{S}_k should only be employed only when necessary, i.e., for the large residual case where the \hat{S}_k is significant. As a result, a hybrid between the DGN-PBM algorithm and a residual approximation algorithm (for example, the MBFGS method is used to approximate \hat{S}_k and is integrated into the full-Newton method) is developed. Then the approximated \hat{H}_k becomes

$$\hat{H}_{s,k} = \hat{J}_k^T \hat{J}_k + \varphi_k \hat{S}_k \quad (4.207)$$

where φ_k is the *switching parameter* and is either zero or one. This method is referred as the *switching method*. Unlike (4.16) and (4.206) in which $0 \leq \phi_k \leq 1$ or S_k can be scaled as proposed in [17, 19], the switching algorithm either includes or excludes \hat{S}_k .

A proper switching criterion for determining φ_k is crucial to the performance of these algorithms. An inappropriate value can lead to tracking failure. For example, if the criterion is too small, more steps are calculated using the MBFGS algorithm though the residual is small at the time. On the other hand, if the criterion is too great, inadequate steps from the MBFGS algorithm may be generated so slower or even unsuccessful tracking may occur. Since the objective of the developed algorithms is to calculate the commanded robot joint angles θ_k for large residual tracking case, the initial image error is assumed to be maximum, the switching criterion sw_{crit} is chosen to be a specified percentage v of the initial error norm $\|f_1\|$ as

$$sw_{crit} = v \|f_1\| \quad (4.208)$$

Then the switching parameter φ_k is determined with the following condition,

```

if  $\|f_k\| < v \|f_1\|$  then
     $\varphi_k = 0$ 
else
     $\varphi_k = 1$ 
end if
```

In this study v is heuristically selected in which the effect of the switching criteria sw_{crit} is discussed and compared to other hybrid methods including the scaling method in [17, 19] and the hybrid algorithms in [49] in Chapter 6.

When the MBFGS algorithm is utilized to approximate \hat{S}_k in the switching method, this algorithm is referred as the *switching MBFGS-DB* algorithm and it is summarized as shown in the pseudo-code in Figure 4.5. In fact, the DBFGS and DFN-BFGS algorithms are similarly implemented into switching schemes as well and are referred

to as the *switching DBFGS-DB* and the *switching DFN-BFGS-DB* algorithms respectively. Tracking performance of each switching algorithm is then evaluated and discussed in Chapter 6.

4.8 Summary

Since the major disadvantage of the DBM-RLS and the DGN-PBM algorithms presented in Chapter 3 are their application to the zero- or small-residual cases, various algorithms are presented in this chapter for solving the large-residual visual servoing problem. It appears in [25] that an algorithm implementing NL2SOL [17, 20] with a trust region method and the LMA shows an improvement over the quasi-Gauss Newton method when dealing with the large-residual problem. However, this algorithm is mainly developed for static target tracking in an eye-to-hand configuration, yet an algorithm for moving target tracking in an eye-in-hand configuration for the large-residual cases does not seem to have appeared in literature. Inspired by [25], three novel algorithms are proposed in this chapter:

1. Approximation of the whole Hessian H_k using the **DBFGS** update

The DBFGS update is used to approximate H_{k-1}

$$\hat{H}_k = \hat{H}_{k-1} + \frac{g_{k-1}^* g_{k-1}^{*T}}{g_{k-1}^{*T} h_{k-1}} - \frac{\hat{H}_{k-1} h_{k-1} h_{k-1}^T \hat{H}_{k-1}}{h_{k-1}^T \hat{H}_{k-1} h_{k-1}} \quad [4.59]$$

where

$$g_{k-1}^* = \hat{J}_k^T \left(f_k + \frac{\partial f_k}{\partial t} h_t \right) - \hat{J}_{k-1}^T \left(f_{k-1} + \frac{\partial f_{k-1}}{\partial t} h_t \right)$$

$$h_{k-1} = \theta_k - \theta_{k-1}$$

Then

$$\theta_{k+1} = \theta_k - \left(\hat{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k}{\partial t} h_t \right)$$

2. Approximation of the residual S_k using the **BFGS** update

The BFGS method is used to approximate \hat{S}_k ,

$$\hat{S}_k = \hat{S}_{k-1} + \frac{z_{k-1}^* z_{k-1}^{*T}}{z_{k-1}^{*T} h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad [4.64]$$

Pseudo-code: The Switching MBFGS-DB Algorithm

Given: $\mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$, $P_0 \in \mathbb{R}^{n \times n}$, $\lambda \in (0, 1)$

Initialize: \hat{J}_0 , θ_0 , θ_1 , H_0 and P_0

for $k = 1, \dots$ **do**

Calculate: \hat{J}_k using a Jacobian estimation in the DGN-PBM algorithm [59]

$$\begin{aligned}\Delta f &= f_k - f_{k-1}; \quad h_{k-1} = \theta_k - \theta_{k-1} \\ \tilde{h} &= \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix} \\ \tilde{J}_{k-1} &= \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix} \\ \tilde{J}_k &= \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1} \\ \tilde{P}_k &= \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right)\end{aligned}$$

Calculate the switching parameter φ_k

if $\|f_k\| < v \|f_1\|$ **then**

$$\varphi_k = 0$$

else

$$\varphi_k = 1$$

Update the residual \hat{S}_k

$$\begin{aligned}\hat{S}_k &= \hat{S}_{k-1} + \frac{z_{k-1}^* z_{k-1}^{*T}}{g_{k-1}^T h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \\ z_{k-1}^* &= \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k; \quad g_{k-1} = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_{k-1}\end{aligned}$$

end if

Calculate $\acute{H}_{s,k}$

$$\acute{H}_k = \hat{J}_k^T \hat{J}_k + \varphi_k \hat{S}_k$$

Calculate: θ_{k+1} using the dynamic full quasi-Newton method

$$\theta_{k+1} = \theta_k - \left(\acute{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

end for

Figure 4.5: Pseudo-code for the switching MBFGS-DB algorithm

where

$$z_{k-1}^* = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k$$

\hat{S}_k from (4.64) is used to approximate \hat{H}_k as

$$\hat{H}_k = \hat{J}_k^T \hat{J}_k + \hat{S}_k$$

Then the dynamic full quasi-Newton method is utilized to estimate θ_{k+1} as

$$\theta_{k+1} = \theta_k - (\hat{H}_k)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

This algorithm is called the *dynamic full Newton method with BFGS* algorithm or DFN-BFGS.

3. Approximation of the residual S_k using the *modified BFGS* update

The MBFGS method is used for estimating the residual S_k ,

$$\hat{S}_k = \hat{S}_{k-1} + \frac{z_{k-1}^* z_{k-1}^{*T}}{g_{k-1}^T h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad [4.67]$$

where

$$z_{k-1}^* = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k$$

$$g_{k-1} = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_{k-1}$$

\hat{S}_k from (4.67) is used to approximate \hat{H}_k as

$$\hat{H}_k = \hat{J}_k^T \hat{J}_k + \hat{S}_k$$

Then θ_{k+1} is given by,

$$\theta_{k+1} = \theta_k - (\hat{H}_k)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

For the case that \hat{S}_k is relatively large compared to $J_k^T J_k$, the MBFGS method is the same as using the unmodified BFGS method (4.64) for approximating \hat{S}_k .

For all algorithms the Jacobian is estimated using the DGN-PBM algorithm as

$$\Delta f = f_k - f_{k-1}$$

$$h_{k-1} = \theta_k - \theta_{k-1}$$

$$h_t = t_k - t_{k-1}$$

$$\tilde{h} = \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix}$$

$$\tilde{J}_{k-1} = \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix}$$

$$\tilde{J}_k = \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1}$$

$$\tilde{P}_k = \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right)$$

Since the first and the second algorithms use the BFGS method, which provides superlinear convergence under reasonable assumptions, a convergence proof is only required to validate the MBFGS algorithm. Consequently, the convergence analysis of the MBFGS method is studied in analogy to the convergence analysis of Broyden's class formula (including the BFGS method) presented in [8]. As a result, the novel MBFGS algorithm assumably yields superlinear convergence if the specified assumptions hold. Finally the criteria for switching between the MBFGS algorithm and the DGN-PBM algorithm is discussed to optimally handle the large-residual problem effectively. The summary of the switching MBFGS-DB algorithm is shown in the pseudo-code in Figure 4.5.

CHAPTER V

DYNAMIC ADAPTIVE FORGETTING FACTOR ALGORITHM

Since the switching MBFGS-DB algorithm is developed from the the dynamic-Gauss-Newton algorithms with partitioning for Broyden's method (DGN-PBM), it inherits the same difficulties presented in the DGN-PBM algorithm including, 1) the selection of an optimal forgetting factor λ and, 2) a proper action dealing with a singular or ill-conditioned Hessian matrix approximation. This chapter discusses various approaches for adaptively selecting an appropriate forgetting factor at each iteration. For the singular Hessian problem the Levenberg-Marquardt algorithm (LMA) is investigated in Chapter 6.

The organization of this chapter is as follow. Section 5.1 briefly reviews the effect of the forgetting factor λ on the DGN-PBM algorithm presented in previous studies. Due to the fact that variable forgetting factor (VFF) algorithms are mostly developed to improve the performance of the RLS algorithm, Section 5.2 presents fundamental background for the RLS algorithm. Then various VFF algorithms that have been widely studied in RLS adaptive filtering are discussed in Section 5.3. Since the existing VFF algorithms propose complex mathematic models that require a number of parameters to be selected by the user, a novel adaptive forgetting factor approach called the *dynamic adaptive forgetting factor (DAFF)* method is developed in Section 5.4. Finally concluding remarks for this chapter are presented in Section 5.5.

5.1 Introduction

Although the dynamic Gauss-Newton algorithms, the DBM-RLS and the DGN-PBM algorithms presented in Chapter 3, offer efficient approaches to recursively estimate the Jacobian \hat{J}_k from the previous information, these algorithms utilize the recursive least-squares (RLS) algorithm in which the performance is dependent on the *exponential weighting factor* or *forgetting factor* λ [53]. The k^{th} iteration of the Jacobian \hat{J}_k is estimated as

$$\begin{aligned}\hat{J}_k &= \hat{J}_{k-1} + \frac{\left(\Delta f - \hat{J}_{k-1}h_\theta - \frac{\partial f_k(t)}{\partial t}h_t\right)h_\theta^T P_{k-1}}{\lambda + h_\theta^T P_{k-1} h_\theta} \\ P_k &= \frac{1}{\lambda} \left(P_{k-1} - \frac{P_{k-1}h_\theta h_\theta^T P_{k-1}}{\lambda + h_\theta^T P_{k-1} h_\theta} \right)\end{aligned}$$

where

$$\Delta f = f_k - f_{k-1}$$

$$h_\theta = \theta_k - \theta_{k-1}$$

and P is a full rank weighting matrix. The forgetting factor λ has the range

$$0 < \lambda \leq 1$$

The inverse of $1 - \lambda$ roughly represents the *memory* of the RLS-based algorithm,

$$n_{memory} = \frac{1}{1 - \lambda} \tag{5.1}$$

When $\lambda = 1$ the memory is infinite and \hat{J}_k is estimated by equally averaging all past information so the RLS algorithm becomes the ordinary least-squares algorithm. If $\lambda < 1$, old data is deweighted so that the calculation of \hat{J}_k relies more on recent data and forgets older data.

Although the RLS algorithm with a constant forgetting factor has fast convergence with a small mean square error (MSE) in stationary environments [41], it does not offer

the optimal performance in time-varying environments. There exists a compromise between applying small and large values of the forgetting factor λ in a RLS-based algorithm. A small value of λ results in large steady-state error but improves tracking while a greater value of λ yields slower convergence but provides lower steady-state error and good stability [53].

The effect of the forgetting factor λ on the dynamic Gauss-Newton algorithms was briefly investigated in [57]. Simulation results show that a lower λ improves convergence time but hurts the steady-state tracking performance. In contrast, a higher λ enhances steady-state tracking capability but increase difficulties in target acquisition. A switching scheme that utilizes a lower λ when the image error norm is relatively large and then switches to a higher λ when the image error norm is lower than a certain criterion provides improved performance of the DGN-PBM algorithm for both transient and steady-state tracking. However, [57] only alternates λ between two constants, i.e., low and high values of λ that need to be selected.

In an adaptive filtering context an algorithm that automatically tunes the forgetting factor λ according to the squared error is known as a *variable forgetting factor* or VFF algorithm. Various studies such as [55, 71, 41] propose different schemes for varying λ with respect to the corresponding squared error demonstrate diverse levels of improvement over a constant λ algorithm. Due to the fact that a great number of VFF algorithms are variants of the RLS algorithm, it is appropriate to briefly review the classical RLS algorithm used in an adaptive filtering context to establish a fundamental understanding of the VFF developments in the next section.

5.2 *RLS Adaptive Filters Overview*

The RLS algorithm is an iterative least-squares scheme that recursively calculates an updated estimation of a model as new data arrives. The RLS algorithm presented in this section reviews classical RLS adaptive filters fundamentals discussed in [32].

The cost function ϵ to be minimized is defined as

$$\epsilon(n) = \sum_{i=1}^n \lambda^{n-i} |e(i)|^2 \quad (5.2)$$

where n is number of observable data, $\epsilon(n)$ denotes the cost function ϵ at incremental time n , and $e(i)$ is defined as

$$e(i) = d(i) - y(i) \quad (5.3)$$

where $d(i)$ is the desired signal and $y(i)$ is the output signal obtained from a transversal filter,

$$y(i) = w^H(n)u(i) \quad (5.4)$$

where $w(n)$ is the weighting vector and $u(i)$ is the signal input vector. To be consistent with adaptive filtering literature in which the signal inputs and the weights are assumed to be complex valued [32], variables used in this section are complex number and the superscript H in (5.4) is referred to as *Hermitian transposition*. Both vectors contain the M most recent data of $w(n)$ and $u(n)$ respectively so at time i the input signal $u(i)$ is defined as

$$u(i) = [u(i), u(i-1), \dots, u(i-M+1)]^T$$

and at incremental time n the weight vector $w(n)$ is defined as

$$w(n) = [w_0(n), w_1(n), \dots, w_{M-1}(n)]^T$$

During the observation interval $1 \leq i \leq n$ the transversal filter contains a fixed length of data n . An example of a transversal filter is shown in Figure 5.1.

Substituting (5.4) into (5.3) gives

$$e(i) = d(i) - w^H(n)u(i) \quad (5.5)$$

The optimal value of $w(n)$ for which the cost function ϵ in (5.2) is minimized is defined as

$$\Phi(n)w(n) = z(n) \quad (5.6)$$

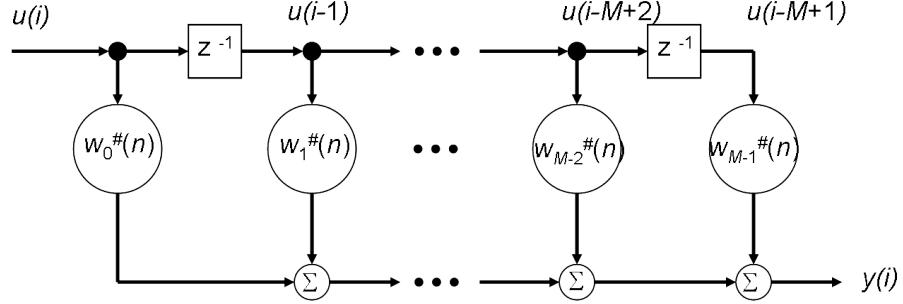


Figure 5.1: Transversal filter [32].

This equation is referred as the *normal equation* [32] where $\Phi(n)$ is the $M \times M$ correlation matrix of the signal input $u(i)$ which is defined as

$$\Phi(n) = \sum_{i=1}^n \lambda^{n-i} u(i) u^H(i) \quad (5.7)$$

and $z(n)$ is the $M \times 1$ cross-correlation vector that relates the signal inputs $u(i)$ to the desired signal $d(i)$ as

$$z(n) = \sum_{i=1}^n \lambda^{n-i} u(i) d^\#(i) \quad (5.8)$$

where $\#$ denotes complex conjugation.

From [32] the correlation matrix Φ can be recursively updated as

$$\Phi(n) = \lambda \Phi(n-1) + u(n) u^H(n) \quad (5.9)$$

where $\Phi(n-1)$ is the previous correlation matrix Φ at incremental time $n-1$. Similarly, the cross-correlation vector $z(n)$ can be updated as

$$z(n) = \lambda z(n-1) + u(n) d^\#(n) \quad (5.10)$$

In order to solve (5.6) for $w(n)$ it is required to invert Φ . The correlation matrix Φ is assumed to be positive definite and its inverse is defined as

$$P(n) \equiv \Phi^{-1}(n) \quad (5.11)$$

$$= \lambda^{-1} P(n-1) - \lambda^{-1} k(n) u^H(n) P(n-1) \quad (5.12)$$

where $P(n)$ is known as the *inverse correlation matrix* [32] and $k(n)$ is the *gain vector* or the *Kalman gain vector* [41] that is given recursively as

$$k(n) = \frac{P(n-1)u(n)}{\lambda + u^H(n)P(n-1)u(n)} \quad (5.13)$$

Rearranging (5.13) yields

$$k(n) = \lambda^{-1}P(n-1)u(n) - \lambda^{-1}k(n)u^H(n)P(n-1)u(n)$$

or

$$\begin{aligned} k(n) &= [\lambda^{-1}P(n-1) - \lambda^{-1}k(n)u^H(n)P(n-1)]u(n) \\ &= P(n)u(n) \end{aligned} \quad (5.14)$$

Rearranging (5.6) gives

$$w(n) = P(n)z(n) \quad (5.15)$$

Substituting (5.10) for $z(n)$ into (5.15) yields

$$w(n) = \lambda P(n)z(n-1) + P(n)u(n)d^\#(n) \quad (5.16)$$

Then substituting (5.12) for only the first $P(n)$ term in the right-hand side of (5.16) gives

$$w(n) = w(n-1) - k(n)u^H(n)w(n-1) + P(n)u(n)d^\#(n) \quad (5.17)$$

Using $k(n) = P(n)u(n)$ from (5.17) and rearranging yields the recursive update of the weight vector $w(n)$ as

$$w(n) = w(n-1) + k(n)\xi^\#(n) \quad (5.18)$$

where

$$\xi(n) = d(n) - u^H(n)w^\#(n-1)$$

Pseudo-code: The RLS Algorithm

Initialize: $P(0) = \delta^{-1}I$, δ = small positive constant, and $w(0) = 0$

for each instant time $n = 1, \dots$ **do**

$$\begin{aligned} k(n) &= \frac{P(n-1)u(n)}{\lambda + u^H(n)P(n-1)u(n)} \\ \xi(n) &= d(n) - w^H(n-1)u(n) \\ w(n) &= w(n-1) + k(n)\xi^\#(n) \\ P_k &= \frac{1}{\lambda} [P(n-1) - k(n)u^H(n)P(n-1)] \\ e(n) &= d(n) - w^H(n)u(n) \end{aligned}$$

end for

Figure 5.2: A pseudo-code for the RLS algorithm [32].

or

$$\xi(n) = d(n) - w^H(n-1)u(n) \quad (5.19)$$

$\xi(n)$ is referred as the *a priori estimation error* [32] since its value is calculated from the previous least-squares estimation of the weight vector $w^H(n-1)u(n)$ at time $n-1$. Then the *a posteriori estimation error* is computed as

$$e(n) = d(n) - w^H(n)u(n) \quad (5.20)$$

The summary of the RLS algorithm is shown in Figure 5.2.

5.3 Variable Forgetting Factor (VFF) Algorithms in Adaptive Filtering

Adaptive or variable forgetting factor (VFF) schemes are developed to adaptively tune the optimal value of λ in each iteration of RLS algorithms and are widely used in adaptive filters. Due to a great number of studies for VFF algorithms only the three most often mentioned algorithms are reviewed in this section:

1. RLS Algorithm with Adaptive Memory [32]
2. Gradient-Based VFF RLS Algorithm (GVFF-RLS) [41]
3. Gauss-Newton VFF RLS Algorithm (GN-VFF-RLS) [71]

These algorithms require selection of various constants determined by the user. The effect of these parameter values on each algorithm performance is investigated in Section 6.5.1.

5.3.1 RLS Algorithm with Adaptive Memory

In [32] it is suggested to select the forgetting factor λ so that the gradient of the cost function with respect to λ is zero. In this case the cost function is defined as

$$Q(n) = \frac{1}{2}E[|\xi(n)|^2] \quad (5.21)$$

where $E[\cdot]$ is the expected value of (\cdot) and the a priori estimation error ξ_k is

$$\xi(n) = d(n) - w^H(n-1)u(n) \quad (5.19)$$

In order to find a proper value of λ that optimizes (5.21), [32] takes the partial derivative of (5.21) with respect to λ , denoted as $\nabla_\lambda(n)$, to give

$$\begin{aligned} \nabla_\lambda(n) &\equiv \frac{\partial Q(n)}{\partial \lambda} \\ &= \frac{1}{2}E\left[\frac{\partial \xi(n)}{\partial \lambda}\xi^\#(n) + \frac{\xi^\#(n)}{\partial \lambda}\xi(n)\right] \\ &= -\frac{1}{2}E[\psi^H(n-1)u(n)\xi^\#(n) + u^H(n)\psi(n-1)\xi(n)] \end{aligned} \quad (5.22)$$

where

$$\begin{aligned} \psi(n) &= \frac{\partial w(n)}{\partial \lambda} \\ &= [I - k(n)u^H(n)]\psi(n-1) + S(n)u(n)\xi^\#(n) \end{aligned} \quad (5.23)$$

and $S(n)$ is the partial derivative of $P(n)$ with respect to λ and is defined as

$$\begin{aligned} S(n) &= \frac{\partial P(n)}{\partial \lambda} \\ &= \frac{1}{\lambda} [I - k(n)u^H(n)] S(n-1) [I - u(n)k^H(n)] + \frac{1}{\lambda} k(n)k^H(n) - \frac{1}{\lambda} P(n) \end{aligned} \quad (5.24)$$

Then the adaptive exponential forgetting factor $\lambda(n)$ can be recursively computed as

$$\lambda(n) = \lambda(n-1) - \eta_1 \frac{\partial Q(n)}{\partial \lambda} \quad (5.25)$$

$$= \{\lambda(n-1) + \eta_1 \operatorname{Re} [\psi^H(n-1)u(n)\xi^\#(n)]\}_{\lambda_-}^{\lambda_+} \quad (5.26)$$

where η_1 is a small positive parameter and is referred to as the *learning-rate* parameter in [32, 29]. However, the detail definition of this parameter is not presented in either reference. The forgetting factor $\lambda(n)$ is truncated with λ_+ and λ_- indicating the allowable range of the forgetting factor $\lambda(n)$. The maximum forgetting factor λ_+ could be close to unity while the minimum forgetting factor λ_- may be obtained from experiment. This VFF method is referred as the *RLS algorithm with adaptive memory* in [32] and its summary is presented in the pseudo code in Figure 5.3.

5.3.2 Gradient-Based VFF RLS Algorithm (GVFF-RLS)

In [70, 41] the *gradient-based VFF RLS algorithm* or GVFF-RLS is presented. The forgetting factor $\lambda(n)$ is computed based on the gradient of the MSE instead of the gradient of squared error as in (5.22). This study presents an improved mean square error analyses where the gradient of the new MSE equation is formulated so that its value becomes positive if the error is large and is negative if the error reaches steady state. The forgetting factor $\lambda(n)$ is then recursively computed by minimizing the gradient of the dynamic MSE equation resulting in improved performance of fast tracking with small MSE for a variety of signal-to-noise ratios (SNRs).

Pseudo-code: The RLS Algorithm with Adaptive Memory

Initialize: $P(0)$, $w(0)$, $\lambda(0)$, $S(0)$, $\psi(0)$, and η_1 is a small positive constant

for each instant time $n = 1, \dots$ **do**

$$\begin{aligned} k(n) &= \frac{P(n-1)u(n)}{\lambda(n-1) + u^H(n)P(n-1)u(n)} \\ \xi(n) &= d(n) - w^H(n-1)u(n) \\ w(n) &= w(n-1) + k(n)\xi^\#(n) \\ P(n) &= \frac{1}{\lambda(n-1)} [P(n-1) - k(n)u^H(n)P(n-1)] \\ \lambda(n) &= \left\{ \lambda(n-1) + \eta_1 \operatorname{Re} [\psi^H(n-1)u(n)\xi^\#(n)] \right\}_{\lambda_-}^{\lambda_+} \\ S(n) &= \frac{1}{\lambda(n)} [I - k(n)u^H(n)] S(n-1) [I - u(n)k^H(n)] \\ &\quad + \frac{1}{\lambda(n)} k(n)k^H(n) - \frac{1}{\lambda(n)} P(n) \\ \psi(n) &= I - k(n)u^H(n)\psi(n-1) + S(n)u(n)\xi^\#(n) \end{aligned}$$

end for

Figure 5.3: A pseudo-code for the RLS algorithm with adaptive memory [32].

The RLS algorithm used in [41] is referred as the *time-variable error weighting RLS* or TWRLS algorithm in which the cost function is defined as

$$\sum_{i=0}^n \lambda^{n-i}(n) [d(i) - w^T(n)u(i)]^2 \quad (5.27)$$

The update of the weight vector $w(n)$ and the error signal $e(n)$ are

$$w(n+1) = w(n) + k(n)e(n) \quad (5.28)$$

$$e(n) = d(n) - w^T(n)u(n) \quad (5.29)$$

And the desired signal $d(n)$ is described as

$$d(n) = w_0^T(n)u(n) + \chi(n) \quad (5.30)$$

where w_0 is the desired weight vector and χ is a Gaussian measurement noise with zero mean and variance σ_χ^2 . In [41] the desired weight vector $w(n)$ is assumably time-varying and is perturbed by a random vector $g(n)$. The update of the consecutive desired weight vector $w_0(n+1)$ is

$$w_0(n+1) = w_0(n) + g(n) \quad (5.31)$$

where $g(n)$ is assumed to be independent Gaussian vector with zero mean and its covariance matrix is denoted as G . This study also assumes that $u(n), \chi(n), g(n)$ are mutually independent.

The correlation matrix $\Phi(n)$ is defined as

$$\Phi(n) = \sum_{i=0}^n \lambda^{n-i} u(i) u^T(i) + \lambda^n(n) \nu I \quad (5.32)$$

where ν is referred as the *initial value* in [41] with no further explanation. However, similar formulation (5.32) appears in [32] where ν is a positive real number called the *regularization parameter*, which is added to avoid singularity of $\Phi(n)$. The Kalman gain $k(n)$ and the inverse of the correlation matrix $P(n)$ are similarly formulated as in the RLS algorithm with adaptive memory [32] (Figure 5.3),

$$\begin{aligned} k(n) &= \frac{P(n-1)u(n)}{\lambda(n-1) + u^H(n)P(n-1)u(n)} \\ P(n) &= \frac{1}{\lambda(n-1)} [P(n-1) - k(n)u^H(n)P(n-1)] \end{aligned}$$

The novelty of this algorithm is attributed to an improved MSE analysis in which the MSE $\sigma_e^2(n)$ is expressed as a sum of the variance of the measurement noise σ_χ^2 and the mean square term $\sigma_{ex}^2(n)$, known as the *excess mean square error* or EMSE,

$$\begin{aligned} \sigma_e^2(n) &= E [e^2(n)] \\ &= \sigma_\chi^2 + \sigma_{ex}^2(n) \end{aligned} \quad (5.33)$$

where σ_{ex}^2 is the MSE caused by the weight errors $w(n) - w_0(n)$. The equation of the MSE $\sigma_e^2(n+1)$ is

$$\sigma_e^2(n+1) = \alpha_n \sigma_e^2(n) + h_n \sigma_e^2(n+1) + Tr\{R_{uu}G\} \quad (5.34)$$

where $TR\{\cdot\}$ is the matrix trace operation and R_{uu} is the correlation matrix of the data vector $u(n)$. The coefficients α_n and h_n are defined as

$$\alpha_n = 1 - \frac{2[(M+1)\tilde{\rho}_n + \rho_n^2] - (M+2)\rho_n}{\rho_n[(M+1)\tilde{\rho}_n + \rho_n^2]} \quad (5.35)$$

$$h_n = \frac{2}{\rho_n} - \frac{2}{[(M+1)\tilde{\rho}_n + \rho_n^2]} \quad (5.36)$$

where

$$\rho_n = 1 + \lambda(n-1)\rho_{n-1}$$

$$\tilde{\rho}_n = 1 + \lambda^2(n-1)\tilde{\rho}_{n-1}$$

Using the dynamic equation of the MSE (5.34) the GVFF algorithm is developed to optimally adjust $\lambda(n)$ so that the gradient of the MSE $\sigma_e^2(n)$ with respect to the forgetting factor λ is minimized. The gradient of $\sigma_e^2(n)$ with respect to λ is

$$\frac{\partial \sigma_e^2(n+1)}{\partial \lambda} = \alpha_n \frac{\partial \sigma_e^2(n)}{\partial \lambda} + \frac{\partial \alpha_n}{\partial \lambda} \sigma_e^2(n) + \frac{\partial h_n}{\partial \lambda} \sigma_\chi^2 \quad (5.37)$$

where $\frac{\partial \alpha_n}{\partial \lambda}$ and $\frac{\partial h_n}{\partial \lambda}$ are defined as

$$\frac{\partial \alpha_n}{\partial \lambda} = \frac{2}{\rho_n^2} \frac{\partial \rho_n}{\partial \lambda} - \frac{M+2}{[(M+1)\tilde{\rho}_n + \rho_n^2]^2} \times \left[(M+1) \frac{\partial \tilde{\rho}_n}{\partial \lambda} + 2\rho_n \frac{\partial \rho_n}{\partial \lambda} \right] \quad (5.38)$$

$$\frac{\partial h_n}{\partial \lambda} = -\frac{2}{\rho_n^2} \frac{\partial \rho_n}{\partial \lambda} - \frac{2}{[(M+1)\tilde{\rho}_n + \rho_n^2]^2} \times \left[(M+1) \frac{\partial \tilde{\rho}_n}{\partial \lambda} + 2\rho_n \frac{\partial \rho_n}{\partial \lambda} \right] \quad (5.39)$$

where

$$\begin{aligned} \frac{\partial \rho_n}{\partial \lambda} &= \rho_{n-1} \\ \frac{\partial \tilde{\rho}_n}{\partial \lambda} &= 2\lambda(n-1)\tilde{\rho}_{n-1} \end{aligned}$$

In practice, the MSE $\sigma_e^2(n)$ can be estimated from $e^2(n)$ [41] as

$$\sigma_e^2(n) = \beta \sigma_e^2(n-1) + (1-\beta)e^2(n) \quad (5.40)$$

where β is a positive constant selected by the user. The variance of the measurement noise σ_χ^2 in (5.37) is calculated from $\sigma_e^2(n)$ as

$$\sigma_\chi^2(n) = \kappa \sigma_\chi^2(n-1) + (1-\kappa)e^2(n) \quad (5.41)$$

where $\kappa > \beta$ is also a positive constant selected by the user. Using (5.37) an update of forgetting factor $\lambda(n)$ is proposed in [41] as

$$\lambda(n) = \left[\lambda(n-1) - \frac{\mu}{1 - \lambda(n-1)} \frac{\partial \sigma_e^2(n+1)}{\partial \lambda} \right]_{2\lambda^\ominus}^{\lambda^\oplus} \quad (5.42)$$

where λ^\oplus and λ^\ominus are the upper and the lower bounds of forgetting factor λ . The upper bound λ^\oplus is analytically selected so that the GVFF-RLS algorithm achieves a minimum steady-state excess mean square error (EMSE). To ensure stability the minimum value of λ is required to be greater than $2\lambda^\ominus$ where λ^\ominus is calculated as

$$\lambda^\ominus = \frac{[(M-2)\rho_{n-1} + \sqrt{D}]}{4[(M+1)\tilde{\rho}_{n-1} + \rho_{n-1}^2]} \quad (5.43)$$

and

$$D = (M-2)^2 \rho_{n-1}^2 - 8(M+2)[(M+1)\tilde{\rho}_{n-1} + \rho_{n-1}^2]$$

where the scalar M is the M most recent data of $w(n)$. The pseudo-code shown in Figure 5.4 is the summary of the GVFF-RLS algorithm.

To evaluate the efficiency of the GVFF-RLS algorithm [41] performs simulations using the GVFF-RLS algorithm in comparison to other VFF-RLS approaches such as those presented in [69, 54] in identification of a switching system. Simulation results show that the GVFF-RLS algorithm yields smaller steady-state EMSE and offers better tracking capability for different SNRs. However, the GVFF-RLS algorithm seems to be somewhat sensitive to β , κ , and μ which are experimentally obtained in these studies.

5.3.3 Gauss-Newton VFF RLS Algorithm (GN-VFF-RLS)

The other often mentioned VFF algorithm in the literature was developed by Song et al. [71] and is known as the *Gauss-Newton variable forgetting factor recursive least-squares* or GN-VFF-RLS algorithm. This method is motivated by a desire to improve the RLS algorithm with the adaptive memory reviewed in Section 5.3.1

Pseudo-code: The GVFF-RLS Algorithm

Initialize: $P(0)$, $w(0)$, $\lambda(0)$, ρ_0 , $\tilde{\rho}_0$, and $\mu, \beta, \kappa \in \mathbb{R}$ where $\kappa > \beta$

for each instant time $n = 1, \dots$ **do**

$$\begin{aligned}
d(n) &= w_0^T(n)u(n) + \chi(n) \\
e(n) &= d(n) - w^T(n)u(n) \\
k(n) &= \frac{P(n-1)u(n)}{\lambda(n-1) + u^H(n)P(n-1)u(n)} \\
w(n+1) &= w(n) + k(n)e(n) \\
P(n) &= \frac{1}{\lambda(n-1)} [P(n-1) - k(n)u^H(n)P(n-1)] \\
\lambda(n) &= \left[\lambda(n-1) - \left(\frac{\mu}{1-\lambda(n-1)} \right) \left(\frac{\partial \sigma_e^2(n)}{\partial \lambda} \right) \right]_{2\lambda^\ominus}^{\lambda^\oplus} \\
\frac{\partial \sigma_e^2(n+1)}{\partial \lambda} &= \alpha_n \frac{\partial \sigma_e^2(n)}{\partial \lambda} + \frac{\partial \alpha_n}{\partial \lambda} \sigma_e^2(n) + \frac{\partial h_n}{\partial \lambda} \sigma_\chi^2 \\
\rho_n &= 1 + \lambda(n-1)\rho_{n-1} \quad ; \quad \frac{\partial \rho_n}{\partial \lambda} = \rho_{n-1} \\
\tilde{\rho}_n &= 1 + \lambda^2(n-1)\tilde{\rho}_{n-1} \quad ; \quad \frac{\partial \tilde{\rho}_n}{\partial \lambda} = 2\lambda(n-1)\tilde{\rho}_{n-1} \\
\alpha_n &= 1 - \frac{2[(M+1)\tilde{\rho}_n + \rho_n^2] - (M+2)\rho_n}{\rho_n[(M+1)\tilde{\rho}_n + \rho_n^2]} \\
h_n &= \frac{2}{\rho_n} - \frac{2}{[(M+1)\tilde{\rho}_n + \rho_n^2]} \\
\frac{\partial \alpha_n}{\partial \lambda} &= \frac{2}{\rho_n^2} \frac{\partial \rho_n}{\partial \lambda} - \frac{M+2}{[(M+1)\tilde{\rho}_n + \rho_n^2]^2} \times \left[(M+1) \frac{\partial \tilde{\rho}_n}{\partial \lambda} + 2\rho_n \frac{\partial \rho_n}{\partial \lambda} \right] \\
\frac{\partial h_n}{\partial \lambda} &= -\frac{2}{\rho_n^2} \frac{\partial \rho_n}{\partial \lambda} - \frac{2}{[(M+1)\tilde{\rho}_n + \rho_n^2]^2} \times \left[(M+1) \frac{\partial \tilde{\rho}_n}{\partial \lambda} + 2\rho_n \frac{\partial \rho_n}{\partial \lambda} \right] \\
\sigma_e^2(n) &= \beta \sigma_e^2(n-1) + (1-\beta)e^2(n) \\
\sigma_\chi^2(n) &= \kappa \sigma_\chi^2(n-1) + (1-\kappa)e^2(n)
\end{aligned}$$

Calculating the lower bound of the forgetting factor λ^\ominus

$$\begin{aligned}
\lambda^\ominus &= \frac{[(M-2)\rho_{n-1} + \sqrt{D}]}{4[(M+1)\tilde{\rho}_{n-1} + \rho_{n-1}^2]} \\
D &= (M-2)^2 \rho_{n-1}^2 - 8(M+2)[(M+1)\tilde{\rho}_{n-1} + \rho_{n-1}^2]
\end{aligned}$$

end for

Figure 5.4: A pseudo-code for the GVFF-RLS algorithm [41].
136

which only works well in the slow time varying environments. The authors proposed to adaptively compute the forgetting factor $\lambda(n)$ using a Gauss-Newton-liked method so that the calculation of $\lambda(n)$ is obtained by solving the second partial derivative of the cost function with respect to λ when it equated to zero. Simulation results in [71] shows improvement of the GN-VFF-RLS algorithm over the RLS algorithm with adaptive memory [32] and the fixed forgetting factor algorithms by offering smaller mean square deviation (MSD) with a wide range of SNRs for autoregressive (AR) parameter estimation.

In [71] the cost function $Q_{GN}(n)$ is defined as

$$Q_{GN}(n) = \frac{1}{2} E [|e(n)|^2] \quad (5.44)$$

The desired signal $d(n)$ is described as

$$d(n) = w^H(n)u(n) + \varphi(n) \quad (5.45)$$

where $\varphi(n)$ is a stationary white noise process with zero mean and variance σ_φ . The error between the desired signal $d(n)$ and its estimation is

$$e(n) = d(n) - w^H(n)u(n) \quad (5.46)$$

The Kalman gain $k(n)$ and the inverse of correlation matrix $P(n)$ are the same as in the RLS algorithm with adaptive memory [32] (Figure 5.3),

$$\begin{aligned} k(n) &= \frac{P(n-1)u(n)}{\lambda(n-1) + u^H(n)P(n-1)u(n)} \\ P(n) &= \frac{1}{\lambda(n-1)} [P(n-1) - k(n)u^H(n)P(n-1)] \end{aligned}$$

and the weight vector $w(n)$ is recursively updated as

$$w(n) = w(n-1) + k(n)e^\#(n) \quad (5.47)$$

There are two main differences between the GN-VFF-RLS algorithm and the RLS algorithm with adaptive memory in Section 5.3.1: i) the cost function is defined using

the a posteriori estimation error $e(n)$ instead of the a priori estimation error $\xi(n)$; ii) the forgetting factor λ is calculated by minimizing the second partial derivative of (5.44) with respect to λ , denoted as $\nabla'_\lambda(n)$, instead of the gradient of the cost function $Q(n)$ with respect to λ ($\nabla_\lambda(n)$) as in (5.22). In [71] the RLS algorithm with adaptive memory [32] is referred to as the *steepest descent VFF-RLS algorithm* while the proposed method is called the *Gauss-Newton approach* due to the analogous derivation of the method to Gauss-Newton method in optimization. The $\nabla'_\lambda(n)$ term is expressed in [71] as

$$\begin{aligned}\nabla'_\lambda(n) &\equiv \frac{\partial^2 Q_{GN}(n)}{\partial \lambda^2} \\ &= (1 - \eta_2) \nabla'_\lambda(n-1) + \eta_2 \psi^H(n-1) u(n) u^H(n) \psi(n-1)\end{aligned}\quad (5.48)$$

Although a small positive number η_2 in this context is referred as the *convergence rate*, its definition is the same as η_1 in (5.26). The forgetting factor is recursively computed as

$$\lambda(n) = \lambda(n-1) + \eta_2 \frac{Re [\psi^H(n-1) u(n) e^\#(n)]}{\nabla'_\lambda(n)} \quad (5.49)$$

where $\psi(n)$ is similar to (5.23),

$$\begin{aligned}\psi(n) &= \frac{\partial w(n)}{\partial \lambda} \\ &= [I - k(n) u^H(n)] \psi(n-1) + S(n) u(n) e^\#(n)\end{aligned}\quad (5.50)$$

and $S(n)$ is the same as (5.24)

$$S(n) = \frac{1}{\lambda} [I - k(n) u^H(n)] S(n-1) [I - u(n) k^H(n)] + \frac{1}{\lambda} k(n) k^H(n) - \frac{1}{\lambda} P(n) \quad [5.24]$$

To evaluate the estimation accuracy the time varying autoregressive (AR) parameter estimation is simulated using different SNR ranges with a fixed frequency. The performance of the exponentially windowed recursive least squares (EW-RLS) algorithm with an optimal fixed forgetting factor, the RLS algorithm with adaptive memory [32], and the GN-VFF-RLS algorithm are compared in [71]. The GN-VFF-RLS algorithm results in the smallest MSD for a wide range of SNRs. In tracking

Pseudo-code: The GN-VFF-RLS Algorithm

Initialize: $P(0)$, $w(0)$, $\lambda(0)$, $S(0)$, $\psi(0)$, and η_2 is a small positive constant

for each instant time $n = 1, \dots$ **do**

$$\begin{aligned}
d(n) &= w^H(n)u(n) + \varphi(n) \\
e(n) &= d(n) - w^H(n)u(n) \\
k(n) &= \frac{P(n-1)u(n)}{\lambda(n-1) + u^H(n)P(n-1)u(n)} \\
w(n) &= w(n-1) + k(n)e^\#(n) \\
P(n) &= \frac{1}{\lambda(n-1)} [P(n-1) - k(n)u^H(n)P(n-1)] \\
\lambda(n) &= \lambda(n-1) + \eta_2 \frac{Re [\psi^H(n-1)u(n)e^\#(n)]}{\nabla'_\lambda(n)} \\
\nabla'_\lambda(n) &= (1 - \eta_2) \nabla'_\lambda(n-1) + \eta_2 \psi^H(n-1)u(n)u^H(n)\psi(n-1) \\
S(n) &= \frac{1}{\lambda(n)} [I - k(n)u^H(n)] S(n-1) [I - u(n)k^H(n)] \\
&\quad + \frac{1}{\lambda(n)} k(n)k^H(n) - \frac{1}{\lambda(n)} P(n) \\
\psi(n) &= I - k(n)u^H(n)\psi(n-1) + S(n)u(n)e^\#(n)
\end{aligned}$$

end for

Figure 5.5: A pseudo-code for the GN-VFF-RLS algorithm [71].

capability testing in which the AR parameter estimation is tested at a fixed SNR with varying frequency, the GN-VFF-RLS algorithm also outperforms the other algorithms to provide smallest MSD. The summary of the GN-VFF-RLS algorithm is shown in the pseudo-code in Figure 5.5.

5.4 *Dynamic Adaptive Forgetting Factor (DAFF) Algorithms for Visual Guide Control*

Due to the complexities of the formulations of the existing VFF algorithms where their performances rely on a number of constant selections such as η_1 , κ , and β , a

novel VFF scheme called the *dynamic adaptive forgetting factor (DAFF) method* is developed in this section.

5.4.1 Previous Work on a VFF Algorithm for Uncalibrated Visual Servoing

In literature there are few studies of uncalibrated visual servoing using a VFF algorithm. An interesting work implementing the RLS algorithm with adaptive memory into a model-free uncalibrated visual servoing algorithm is presented in [29]. This method combines a classical adaptive least squares (ARLS) algorithm similar to the RLS algorithm with adaptive memory [32] in Section 5.3.1 into the DGN-PBM algorithm [59] and is referred as an *uncalibrated visual servoing using adaptive recursive least squares (VS-ARLS)* algorithm. Since the DGN-PBM algorithm is the RLS algorithm, there exists parameter equivalencies between the two algorithms [29] and is presented herein as Table 5.1.

Table 5.1: The RLS and the DGN-PBM Parameter Equivalencies [29].

RLS Variable	DGN-PBM Variable
$d(n)$	$\Delta f = f_k - f_{k-1}$
$w^H(n)$	$\tilde{J}_k = \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix}$
$u(n)$	$\tilde{h} = \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix}$
$\xi(n)$	$\Delta f - \tilde{J}_{k-1}\tilde{h}$

The VS-ARLS algorithm can be derived by merely substituting the DGN-PBM parameters into the ARLS algorithm for forgetting factor calculation and then integrated into the original DGN-PBM algorithm. One important note is that in the

original adaptive filtering context such as one presented in [32] $d(n)$ and $\xi(n)$ are scalars whereas $w(n)$ is a vector. However, for the DGN-PBM algorithm Δf and $\Delta f - \tilde{J}_{k-1} \tilde{h}$ are vectors and \tilde{J}_k is a matrix. The summary of the VS-ARLS algorithm is shown in the pseudo-code in Figure 5.6.

To evaluate tracking performance of the VS-ARLS algorithm a 6-DOF, eye-in-hand Puma 560 robot is used in [29] for simulation. The results obtained from the VS-ARLS algorithm are compared with results from the standard RLS algorithm¹ for static and dynamic tracking. Simulation results show that the average MSE from both algorithms are nearly the same in static tracking but the VS-ARLS algorithm offers a lower average MSE than the standard RLS algorithm in the dynamic target tracking. It is mentioned in [29] that the uncalibrated visual servoing is a highly nonlinear system that is vulnerable to system modeling and noises. So there exists a trade-off between using more information to improve noise resistance and minimizing the image error. Consequently, [29] recommends to explicitly construct a mathematical model relating the error and noise to the forgetting factor calculation to improve performance of the VS-ARLS algorithm.

5.4.2 DAFF Method

Although various VFF algorithms show improved results compared to a fixed forgetting factor, the mathematical formulas are complicated and the performances of these algorithms are dependent on one or more variables. For example, the adaptive memory scheme [32], the VS-ARLS [29], and the GN-VFF-RLS [71] algorithm performances rely on the learning rate η_1 and η_2 , while the GVFF-RLS [41] depends on μ , β , and κ . Due to the complexities of the formulations, the effects of these variables to these existing VFF algorithms are difficult to determine and have not been thoroughly presented. As a result, a simpler approach of adaptively calculating the

¹It is not clear what exactly is the *standard RLS* algorithm used in [29], but it is most likely referred to the DGN-PBM algorithm with a fixed forgetting factor λ .

Pseudo-code: The VS-ARLS Algorithm

Given: $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$; $(\hat{f}_t)_0 \in \mathbb{R}^{m \times 1}$; $P_0 \in \mathbb{R}^{n+1 \times n+1}$; $\lambda \in (0, 1)$

Initialize: $J_0, \theta_0, \theta_1, (\hat{f}_t)_0, P_0, S_0, \psi_0$, and η_1 is a small positive constant

for $k = 1, \dots$ **do**

$$\begin{aligned}\Delta f &= f_k - f_{k-1} \\ h_\theta &= \theta_k - \theta_{k-1} \\ h_t &= t_k - t_{k-1} \\ \tilde{h} &= \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix} \\ \tilde{J}_{k-1} &= \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix}\end{aligned}$$

Calculate: λ_k

$$\begin{aligned}\lambda_k &= \left\{ \lambda_{k-1} + \eta_1 \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right)^T \psi_{k-1} \tilde{h} \right\}_{\lambda_-}^{\lambda_+} \\ S_k &= \frac{1}{\lambda_k} \left[I - \tilde{k}_k \tilde{h}^T \right] S_{k-1} \left[I - \tilde{h} \tilde{k}_k^T \right] \\ &\quad + \frac{1}{\lambda_k} \tilde{k}_k \tilde{k}_k^T - \frac{1}{\lambda_k} \tilde{P}_k \\ \psi_k &= \left[I - \tilde{k}_k \tilde{h}^T \right] \psi_{k-1} + \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \left(S_k \tilde{h} \right)^T\end{aligned}$$

Calculate: \tilde{J}_k

$$\begin{aligned}\tilde{J}_k &= \tilde{J}_{k-1} + \left(\lambda_k + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1} \\ \tilde{k}_k &= \left(\lambda_k + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \right) \\ \tilde{P}_k &= \frac{1}{\lambda_k} \left(\tilde{P}_{k-1} - \tilde{k}_k \tilde{h}^T \tilde{P}_{k-1} \right)\end{aligned}$$

Calculate: θ_{k+1}

$$\theta_{k+1} = \theta_k - \left(\hat{J}_k^T \hat{J}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

end for

Figure 5.6: A pseudo-code for the VS-ARLS algorithm [29]

forgetting factor λ is investigated.

It is desirable to achieve fast convergence in transience and high accuracy in steady-state tracking. To achieve fast convergence a robot is expected to move in the most direct path from its initial configuration θ_0 to reach steady-state tracking. The value of the forgetting factor, where

$$n_{memory} = \frac{1}{1 - \lambda} \quad [5.1]$$

roughly represents the memory of the RLS-based algorithm, should be relatively small (less memory) in the transient state so that the updated forgetting factor λ invokes more dependence on the current image error, i.e., less past information is included in approximating the Jacobian \hat{J}_k and the Hessian \hat{H}_k . Moreover, at the beginning of tracking there is not enough information for uncalibrated visual servoing to quantitatively understand the tracking behaviors of the robot in relation to its current environment. So it is desired to update the Jacobian \hat{J}_k and the Hessian \hat{H}_k based on the most recent data, i.e., using less memory for calculation. In contrast, λ should be relatively high, i.e. closer to unity, once the robot reaches steady-state tracking. The higher λ value results in increasing the memory so that the current Jacobian \hat{J}_k and the Hessian \hat{H}_k are estimated by averaging down more past information.

Since the image error is typically large in the transient state, especially at the beginning of the tracking process, and becomes relatively small during the steady state, it can be hypothesized that the forgetting factor λ should be small (less memory) when the image error is large during transience, and λ should get closer to unity (more memory) when the image error becomes small in steady-state tracking. In the case that the current image error abruptly becomes substantially larger than the past few errors, which may be caused by sudden changes of the environment or the target velocity, either changes in direction, speed, or the combination of both, it is hypothesized that the past and current models used to estimate the Jacobian \hat{J}_k and the Hessian \hat{H}_k no longer represent the actual behaviors of the current system. Thus, the

calculation of \hat{J}_k and \hat{H}_k should include less memory (smaller value of the forgetting factor λ). Regardless of cause, the forgetting factor λ is somewhat inversely related to the current image error norm $\|f_k\|$.

For an uncalibrated visual servoing system, it is desired to study how λ_k responds to the image error norm $\|f_k\|$. However, the analytical calculation of adaptive λ_k with respect to $\|f_k\|$ is challenging due to the complexity of the mathematical formulation presented in Section 5.3. For good tracking, the robot is expected to quickly reach the desired target trajectory. Thus image error norm $\|f_k\|$, in analogy to a continuous function in the time domain, could be expected to exponentially decay to zero as the robot reaches steady-state tracking. This behavior is in fact similar to the response of a first-order differential dynamic system as shown in Figure 5.7a. In contrast, values of λ_k , also in analogy to a first-order system, could be expected to exponentially increase as shown in Figure 5.7b since λ_k is assumed to be in reversely related to $\|f_k\|$. Inspired by these observations, a novel adaptive forgetting factor called the *dynamic adaptive forgetting factor* or DAFF method is presented.

The idea is to heuristically select λ_k based on $\|f_k\|$ in analogy to the response of a first-order dynamic system. From the relationship between λ_k and n_{memory} in (5.1), rearranging gives

$$\lambda_k = 1 - \frac{1}{n_{memory,k}}$$

Denoting

$$\Lambda \equiv \frac{1}{n_{memory}}$$

then

$$\Lambda_k = 1 - \lambda_k \quad (5.51)$$

For a large value of $\|f_k\|$, λ_k should be small and Λ_k large. On the contrary, a small $\|f_k\|$ should result in large a λ_k and small Λ_k . Unlike λ_k , Λ_k is adjusted indirect

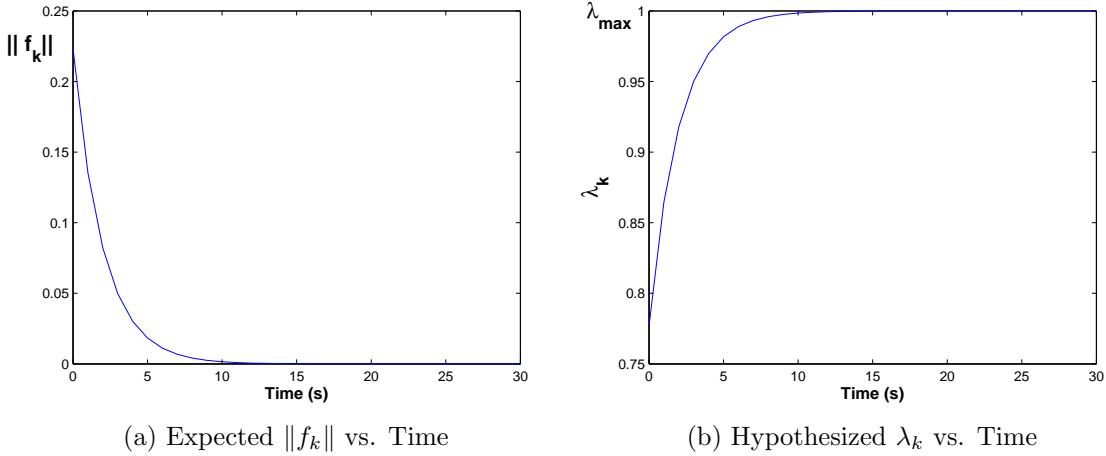


Figure 5.7: The expected $\|f_k\|$ values for ideal tracking performance and the hypothesized λ_k values in corresponding to $\|f_k\|$ with respect to time (s).

to $\|f_k\|$. Thus the changes of Λ_k value with respect to $\|f_k\|$ could be treated in an analogous way to a first order differential system. Consequently, the input function $F(s)$ is the image error norm $\|f(t)\|$ and the output function $\Lambda(s)$ is $\Lambda(t)$ in the time domain. Then the transfer function $G(s)$ in the Laplace domain is expressed as

$$G(s) = \frac{\Lambda(s)}{F(s)} = \frac{1}{\tau s + 1} \quad (5.52)$$

where τ is a time constant. In fact the input function $F(s)$ is the normalized form of the image error norm \check{f}_k where

$$\check{f}_k = \frac{\|f_k\|}{f_{max}} \quad (5.53)$$

and f_{max} is the maximum error norm of all the previous errors $f(\theta, t)$,

$$f_{max} = \max \{\|f_1\|, \|f_2\|, \dots, \|f_k\|\} \quad (5.54)$$

The normalization of the image error \check{f}_k is necessary to limit the calculated forgetting factor λ_k within the allowable range $\lambda_k \in (0, 1]$. As a large residual problem is assumed, $f_{max} = \|f_1\|$ is applied.

Due to the fact that all variables used in the switching MBFGS-DB visual servoing algorithm are available only at a fixed sampling time T , (5.52) is transformed into the

transfer function of a discrete system, $G(z)$. Using the z-transform analysis with the *trapezoid rule substitution*, also known as *Tustin's method* or the *Binary transform* [24], the transfer function of a continuous time system is approximately converted into the discrete time domain by substituting s in (5.52) by

$$s \approx \frac{2}{T} \left(\frac{z - 1}{z + 1} \right) \quad (5.55)$$

where T is the sampling time. Then substituting (5.55) into (5.52) yields

$$G(z) = \frac{\Lambda(z)}{F(z)} = \frac{T(1 + z^{-1})}{(2\tau + T) + z^{-1}(T - 2\tau)} \quad (5.56)$$

so the discretized model of the forgetting factor λ_k can be expressed as

$$\Lambda_k = \frac{T(\check{f}_k + \check{f}_{k-1}) + (2\tau - T)\Lambda_{k-1}}{2\tau + T} \quad (5.57)$$

where $2\tau > T$ to ensure that Λ_k is positive.

There exists an optimal range of the forgetting factor λ_k for which the value of the forgetting factor λ_k cannot be too small to achieve steady-state tracking, while infinite memory where $\lambda_k = 1$ can lead to instabilities and divergence for dynamic environments [32]. For this reason, the calculation of λ_k in (5.51) is modified into a form,

$$\lambda_k = \{\lambda_{max}(1 - \Lambda_k)\}_{\lambda_{min}}^{\lambda_{max}} \quad (5.58)$$

where λ_{max} and λ_{min} are the maximum and the minimum allowable values of the forgetting factor λ_k . Although it is often recommended to choose λ_{max} closer to unity, there are not many guidelines for determining the minimum value of the forgetting factor or to properly select the optimal range. The effect of the λ range on tracking performance is investigated in Section 6.5.1. A summary of the DAFF method with the DGN-PBM algorithm is shown in the pseudo-code in Figure 5.8.

Besides the selection of the forgetting factor range, the performance of the DAFF algorithm is also affected by the time constant τ . For a dynamic system analysis, the

Pseudo-code: The DAFF Method with the Switching MBFGS-DB Algorithm

Given: $\mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$, $P_0 \in \mathbb{R}^{n \times n}$, $\lambda \in (0, 1)$

Initialize: \hat{J}_0 , θ_0 , θ_1 , H_0 , P_0 , λ_{max} , λ_{min} , and τ

for $k = 1, \dots$ **do**

$$\begin{aligned}\Delta f &= f_k - f_{k-1}; & h_{k-1} &= \theta_k - \theta_{k-1} \\ \tilde{h} &= \begin{bmatrix} (\theta_k - \theta_{k-1}) & (t_k - t_{k-1}) \end{bmatrix}' \\ \tilde{J}_{k-1} &= \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix}\end{aligned}$$

Calculate: the forgetting factor λ_k

$$\begin{aligned}\Lambda_k &= \frac{T(\check{f}_k + \check{f}_{k-1}) - (T - 2\tau)\Lambda_{k-1}}{2\tau + T} \\ \lambda_k &= \{\lambda_{max}(1 - \Lambda_k)\}_{\lambda_{min}}^{\lambda_{max}} \\ \check{f}_k &= \frac{\|f_k\|}{f_{max}}\end{aligned}$$

Calculate: \hat{J}_k and \tilde{P}_k

$$\begin{aligned}\tilde{J}_k &= \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1} \\ \tilde{P}_k &= \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right)\end{aligned}$$

if $\|f_k\| < 0.1\|f_1\|$ **then**

$$\varphi_k = 0$$

else

$$\varphi_k = 1$$

Update: the residual \hat{S}_k

$$\begin{aligned}\hat{S}_k &= \hat{S}_{k-1} + \frac{z_{k-1}^* z_{k-1}^{*-T}}{g_{k-1}^T h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \\ z_{k-1}^* &= \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k; & g_{k-1} &= \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_{k-1}\end{aligned}$$

end if

Calculate $\hat{H}_{s,k}$

$$\hat{H}_{s,k} = \hat{J}_k^T \hat{J}_k + \varphi_k \hat{S}_k$$

Calculate: θ_{k+1} using the dynamic full quasi-Newton method

$$\theta_{k+1} = \theta_k - \left(\hat{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

end for

Figure 5.8: Pseudo-code for the DAFF method with the switching MBFGS-DB algorithm

time constant τ determines how quickly the system reaches steady state. Likewise, a direct interpretation of the time constant τ for the DAFF algorithm should, in theory, identify how fast λ_k gets close to steady-state, i.e., reaching λ_{max} . However, for this application, the output of interest is the time-varying image error f_k that is influenced by accuracy of the Jacobian \hat{J}_k and the Hessian \hat{H}_k approximation, which are in fact dependent on a proper value of λ_k . Therefore, the time constant τ used in the DAFF method does not directly impact the reduction of the image error f_k for this algorithm. In this study τ is heuristically selected so that the value of λ_k does not increase too slowly or too rapidly. The effects of the forgetting factor range and the time constant τ on the performance of the DAFF method are discussed in Section 6.5.1.

5.5 Summary

Since the switching MBFGS-DB algorithm utilizes the RLS method for Jacobian approximation, its performance is dependent on the forgetting factor λ . Various VFF algorithms developed for adaptive filtering are reviewed in Section 5.3. Due to mathematical formulations of existing VFF algorithms that require a number of parameter selections, a novel method called the *dynamic adaptive forgetting factor (DAFF)* method is developed in Section 5.4. The behavior of λ_k is reviewed in analogy to the transient response of a first-order dynamic system. In this case, the input function is $\Lambda_k \equiv 1 - \lambda_k$ and the output function is the normalized $\|f_k\|$. Unlike other existing VFF algorithms this method is more simple yet effectively calculates λ_k accordingly to $\|f_k\|$ with only one parameter, the time constant τ . The performance evaluations of the presented VFF algorithms are discussed in Chapter 6.

CHAPTER VI

SIMULATION

A theoretical foundation for the various switching algorithms have been presented. This chapter evaluates the novel uncalibrated visual servoing algorithms as compared to the dynamic quasi-Gauss-Newton algorithms either with or without partitioning for Broyden's method (the DBM-RLS and the DGN-PBM algorithms) with a fixed λ_k . Simulation results demonstrate the switching MBFGS-DB with the DAFF algorithm significantly improves transient and steady-state tracking for the large residual error case. Furthermore, the control scheme offers better tracking capability in the presence of measurement and processing noise for a variety of target trajectories using different robot configurations and degrees-of-freedom.

The organization of this chapter is as follows. Section 6.1 overviews the different algorithms evaluated in this study. Section 6.2 gives a summary of simulation setups with a variety of robots and target trajectories. Various switching algorithms are evaluated for the proposed residual approximations integrated with the DGN-PBM algorithm for large residual tracking problem in Section 6.3. Since the LMA algorithm is expected to improve the conditioning of the Hessian approximation, in Section 6.4 it is implemented with the switching algorithms from Section 6.3. All the switching algorithms utilize the recursive least-squares (RLS) algorithm and its performance is dependent on the forgetting factor λ . Various variable forgetting factor (VFF) algorithms reviewed in Chapter 5 are presented in Section 6.5. These are integrated into the MBFGS-DB algorithm which offers the best tracking performance compared to the other proposed switching algorithms. VFF improves tracking, especially in the presence of measurement and processing noise. In Section 6.6 the switching

MBFGS-DB and DGN-PBM algorithms with various VFF algorithms are compared with/without implementation of LMA for a cycloidal trajectory tracking using the PUMA 560 robot with an eye-in-hand camera. Section 6.7 summarized the results.

6.1 Overview

The proposed uncalibrated visual control is built upon the dynamic quasi-Gauss-Newton algorithms developed by Piepmeyer et al. [57, 59] to improve:

1. Large initial error for dynamic target tracking
2. A singular or ill-conditioned Hessian matrix $F_{\theta\theta} = J_k^T J_k + S_k$
3. Selection of an optimal forgetting factor λ

The algorithms for improving each difficulty are summarized in Table 6.1.

The DBM-RLS and the DGN-PBM algorithms [57, 59] provide stable and convergent tracking for zero- or small-residual cases where the Gauss-Newton method assumes that the residual $S_k = \frac{\partial J_k^T}{\partial \theta} f_k$ can be neglected. This assumption is true when the initial robot configuration θ_0 is in the neighborhood of the target acquisition configuration θ^* . Otherwise the residual S_k may be significant and the Gauss-Newton is less appropriate. Instead, the dynamic BFGS, the DFN-BFGS, and the MBFGS algorithms approximate the residual S_k is discussed in Chapter 4. These are used in a switching scheme so the residual S_k is only utilized when the average error norm $\|f_k\|$ is larger than a specified criterion. Otherwise the DGN-PBM algorithm is employed. The proposed schemes are referred to as the *switching DBFGS-DB*, the *switching DFN-BFGS-DB*, and the *switching MBFGS-DB* algorithms.

Two different robots and several target trajectories are used to validate these switching algorithms for large residual tracking. To study the effect of the residual \hat{S}_k approximation, a fixed forgetting factor is used without the Levenberg-Marquardt algorithm (LMA). Then the performance of each large residual algorithm is compared

Table 6.1: A summary of methods to improve the dynamic quasi-Gauss Newton algorithms.

Problem	Algorithm
Large Residual	1. Switching DBFGS-DB 2. Switching DFN-BFGS-DB 3. Switching MBFGS-DB
Singular Hessian Approximation	LMA
Variable FF	1. Fixed λ 2. VS-ARLS 3. GN-VFF-RLS 4. DAFF 5. Alternating between $[\lambda_{low}, \lambda_{high}]$

to the DBM-RLS and the DGN-PBM algorithms. Noise is not present in this set of simulations.

Then the switching algorithms are used with the LMA to improve the conditioning of the Hessian approximations. The LMA is well-known for solving nonlinear unconstrained optimizations problems if the Hessian matrix becomes ill-conditioned or singular. Though the LMA appears in various forms in the literature, such as [50, 47], the implementation of [47] is utilized. The switching algorithms with and without the LMA are compared to the DBM-RLS and the DGN-PBM algorithms using a fixed forgetting factor without added noise.

Finally various switching algorithms with and without the LMA are investigated with different VFF algorithms to improve tracking performance, especially in the presence of measurement and process noise. The robot configurations and camera configurations used to perform the tests are reviewed in the next section.

6.2 System Description

6.2.1 Robot System

Two robot systems are used, a spatial RRR robot and a 6 DOF robot as shown in Figure 6.1a. The RRR robot model is similar to the robot system described in [25] and its Denavit-Hartenberg (DH) parameters are summarized in Table 6.2. The first revolute joint is vertical and the second and third joints are horizontal. The 6 DOF robot kinematic model used for all simulations is the PUMA 560 kinematics provided in the *Robotic Toolbox* by Corke [10].

Table 6.2: The DH parameters of a RRR robot.

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0.3	θ_1
2	$\frac{\pi}{2}$	0.4	0	θ_2
3	π	0.4	0	θ_3

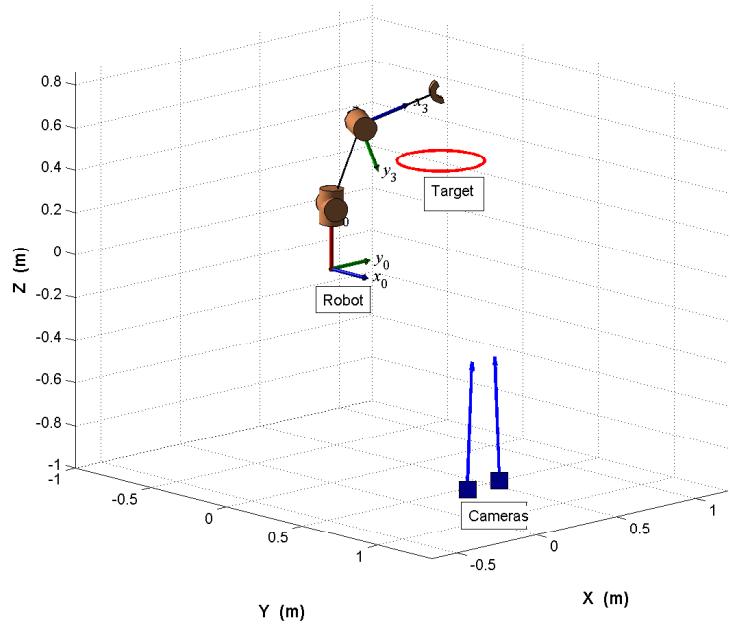
6.2.2 Camera System

All cameras used in this study have the same intrinsic parameters given in Table 6.3. The camera transformation matrix for the given intrinsic and extrinsic parameters is obtained using functions provided in the *Machine Vision Toolbox* by Corke [11].

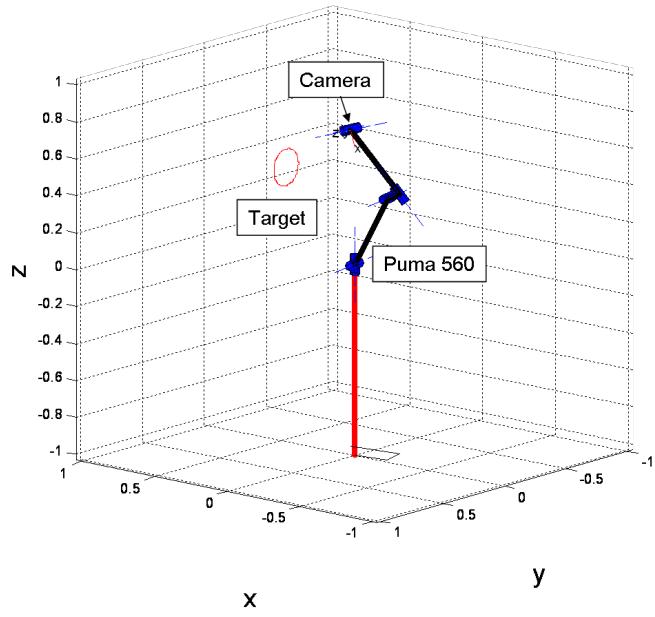
In Table 6.3 f is the focal length, $cp.px$ and $cp.py$ are the horizontal and vertical pixel pitches of the sensor, and $cp.u$ and $cp.v$ are principal point or the image center.

Different camera systems affect tracking performance and two representative camera settings are used:

1. Eye-to-hand camera setting - one or more cameras are fixed at stationary locations. Two camera arrangements are investigated.



(a)



(b)

Figure 6.1: (a)The RRR robot, (b) The PUMA 560 robot.

Table 6.3: Camera parameters.

Parameter	Value
f	0.01 (m)
$cp.px$	$\frac{1}{1.90625e - 5}$ (pixels/m)
$cp.py$	$\frac{1}{1.90625e - 5}$ (pixels/m)
$cp.u$	0 (pixels)
$cp.v$	0 (pixels)

2. Eye-in-hand camera setting - one or more cameras are attached to the EE of the robot and the optical axis is assumed to be coincident (for one camera case) or nearly parallel (for multiple camera case) with the z axis of the final frame of the manipulator.

6.2.3 Target Trajectories

Various target trajectories are used:

1. Circular trajectory - a simple translational, circular path generated on either the X-Y or the Y-Z plane
2. Square trajectory - a more difficult path due to velocity discontinuities at the corners
3. Cycloid trajectory - a complex trajectory where both direction and speed of the target are changed over its cycle
4. Helical trajectory - an out-of-a-plane trajectory

Note that these trajectories are all translational. Examples of the RRR robot with two eye-to-hand cameras and the Puma 560 robot with an eye-in-hand camera are

shown tracking circular trajectories in Figure 6.1a and Figure 6.1b respectively.

6.2.4 Assumptions

All simulations presented in this chapter are generated under the following assumptions:

1. The robot joint positions reach the commanded locations within the specified time increment.
2. The dynamics of the robot are not considered and is assumed to have no effect on the tracking performance as long as the first assumption is applied
3. The vision acquisition data at each update represents the changes in features from the most recently commanded motion.
4. For the eye-to-hand camera setting the target position y^* is only a function of time t and is independent of the robot joint angles θ .
5. The EE position y is only a function of robot joint angles θ and is independent of time t .

For almost all tests a sampling time of $h_t = 0.05$ sec (20 Hz) is used and the maximum increment of each joint is limited to $\pm 5^\circ$ to ensure reasonably small motion Jacobian-based control.

6.3 *Performance Evaluation of the Switching Algorithms for Large Residual Tracking*

As mentioned in Chapter 4 the dynamic quasi-Gauss-Newton algorithms already work well for the zero- or small-residual cases. The approximation of the residual \hat{S}_k is only recommended when the average $\|f_k\|$ is large so it is implemented into a hybrid switching algorithm. The approximation of the residual term using the *dynamic BFGS*,

the *DFN-BFGS*, and the *MBFGS* algorithms become the *switching DBFGS-DB*, the *switching DFN-BFGS-DB*, and the *switching MBFGS-DB* algorithms respectively.

In addition, the Fu algorithm in [25] (reviewed in Section 4.3.2) is compared with the switching algorithms. Although a trust region method is used with residual estimation, only the residual portion is employed for comparison. The method deteriorates when the image error becomes small so it is implemented into a switching scheme.

A summary of the residual \hat{S}_k algorithms are listed in Table 6.4. The switching schemes are summarized by the pseudo-code in Figure 6.2.

To isolate the residual \hat{S}_k approximation, the forgetting factor λ held constant and the LMA is not used. Since noise compensation requires a proper selection of the forgetting factor value, the effect of noise disturbance is deferred so no noise is added. The initial Jacobian is estimated by successively perturbing each joint by a small angle.

For all approaches the fixed forgetting factor $\lambda = 0.5$ is used despite the hypothesis made in Chapter 5 that a small value of λ should be applied if image error is large and a large λ should be employed if the image error becomes small. For this reason, a mean value $\lambda = 0.5$ seems to be a reasonable choice. A study of λ on tracking performance is presented in Section 6.5.

The critical parameter for switching algorithms is the switching criterion sw_{crit} which determines when the residual approximation is included. Too great a value yields slower convergence while too small a value sometimes leads to divergence as discussed in Section 6.3.3. In this study sw_{crit} is determined by heuristically as a percentage of the initial error norm f_1 which is assumed to be the largest,

$$sw_{crit} = v \|f_1\|$$

The $v = 0.3$ or equivalently sw_{crit} is 30% of $\|f_1\|$ is utilized for all switching schemes except for the switching Fu-DB algorithm where $v = 0.5$ is used to avoid

Table 6.4: A summary of methods for estimating residual \hat{S}_k : Dynamic BFGS, DFN-BFGS, MBFGS, and Fu's Method

Method	Equation
Dynamic BFGS	$\hat{H}_k = \hat{H}_{k-1} + \frac{g_{k-1}^* g_{k-1}^{*T}}{g_{k-1}^{*T} h_{k-1}} - \frac{\hat{H}_{k-1} h_{k-1} h_{k-1}^T \hat{H}_{k-1}}{h_{k-1}^T \hat{H}_{k-1} h_{k-1}} \quad [4.59]$ $g_{k-1}^* = \hat{J}_k^T \left(f_k + \frac{\partial f_k}{\partial t} h_t \right) - \hat{J}_{k-1}^T \left(f_{k-1} + \frac{\partial f_{k-1}}{\partial t} h_t \right)$ $h_k = \theta_{k+1} - \theta_k \quad [4.60]$
DFN-BFGS	$\hat{S}_k = \hat{S}_{k-1} + \frac{z_{k-1}^* z_{k-1}^{*T}}{z_{k-1}^{*T} h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad [4.64]$ $z_{k-1}^* = J_k^T f_k - J_{k-1}^T f_k$
MBFGS	$\hat{S}_k = \hat{S}_{k-1} + \frac{z_{k-1}^* z_{k-1}^{*T}}{g_{k-1}^T h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \quad [4.67]$ $z_{k-1}^* = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k$ $g_{k-1} = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_{k-1}$
Fu's Method	$\hat{S}_k = \hat{S}_{k-1} + \frac{\left(z_{k-1}^* - \hat{S}_{k-1} h_{k-1} \right) g_{k-1}^T + g_{k-1} \left(z_{k-1}^* - \hat{S}_{k-1} h_{k-1} \right)^T}{g_{k-1}^T h_{k-1}}$ $-\frac{h_{k-1}^T \left(z_{k-1}^* - \hat{S}_{k-1} h_{k-1} \right) g_{k-1} g_{k-1}^T}{\left(g_{k-1}^T h_{k-1} \right)^2} \quad [4.32]$ $z_{k-1}^* = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k$ $g_{k-1} = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_{k-1}$

divergence.

The settling time t_s is defined as the time required to move the EE from its initial configuration θ_0 to reach and remain within a given range of the image error (the cut-off steady-state error). In this study the cut-off steady-state error value is approximately 1% of the initial average error norm $\|f_1\|$ and in this particular case the cut-off are 0.3348 pixels and 0.2883 for camera 1 and camera 2 respectively.

The switching schemes are compared with the DBM-RLS and DGN-PBM algorithms in which the residual \hat{S}_k term is ignored. Two robot systems with different camera configurations are used:

1. The spatial RRR robot with two eye-to-hand cameras for tracking one feature point
2. The PUMA 560 robot with an eye-in-hand camera configuration for tracking four feature points

6.3.1 The RRR robot with a circular trajectory

First, the RRR robot with two eye-to-hand cameras is used to test the switching MBFGS algorithm for large error residual tracking. A single point target is moving in the circular trajectory

$$\begin{aligned} x &= 300 + R \sin(\omega k T) \quad \text{mm} \\ y &= 500 + R \cos(\omega k T) \quad \text{mm} \\ z &= 500 \quad \text{mm} \end{aligned} \tag{6.1}$$

where $R = 100$ mm is the radius of the circular path, $T = 0.05$ s is the sampling period, k is the iteration number, and $\omega = 0.45$ rad/s is the angular speed. The starting robot joint angles are $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$. The two camera configurations are shown in Figure 6.1a and located by the homogeneous matrices in an arrangement

similar to [25]

$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 0.4453 \\ 0 & 0.9988 & 0.0499 & 0.4307 \\ 0 & -0.0499 & 0.9988 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.2)$$

$$T_2 = \begin{bmatrix} 1 & 0 & 0 & 0.4453 \\ 0 & 0.9988 & -0.0499 & 0.6307 \\ 0 & 0.0499 & 0.9988 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.3)$$

To compare tracking performance of all switching algorithms with the dynamic-Gauss-Newton based algorithms (the DBM-RLS and the DGN-PBM) the EE and feature points in the image plane, the task space view of one camera, and the error norm of each algorithm are shown in Figure 6.3, Figure 6.4, and Figure 6.5 respectively.

Table 6.5: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\lambda = 0.5$, and $v = 0.3$ for all algorithms.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
DBM-RLS	0.0171	0.0183	1.6
DGN-PBM	0.1254	0.1258	2.3
DBFGS-DB	0.0948	0.0954	3.7
DFN-BFGS-DB	0.0870	0.0870	5.6
MBFGS-DB	0.1059	0.1054	1.3
Fu-DB	0.0905	0.0903	5.5

Figure 6.4 demonstrates that the initial EE location is far from the target trajectory to ensure large residual to start. The MBFGS-DB algorithms offers the fastest

Pseudo-code: The Switching DBFGS-DB, DFN-BFGS-DB, MBFGS-DB , and Fu-DB Algorithms

Given: $\mathbb{R}^n \rightarrow \mathbb{R}^m ; \theta_0, \theta_1 \in \mathbb{R}^n ; \hat{J}_0 \in \mathbb{R}^{m \times n}, P_0 \in \mathbb{R}^{n \times n}, \lambda \in (0, 1)$

Initialize: $\hat{J}_0, \theta_0, \theta_1, H_0$ and P_0

for $k = 1, \dots$ **do**

Calculate: \hat{J}_k using a Jacobian estimation in the DGN-PBM algorithm [59]

$$\Delta f = f_k - f_{k-1} ; \quad h_{k-1} = \theta_k - \theta_{k-1}$$

$$\tilde{h} = \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix}$$

$$\tilde{J}_{k-1} = \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix}$$

$$\tilde{J}_k = \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1}$$

$$\tilde{P}_k = \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right)$$

Calculate the switching parameter φ_k

if $\|f_k\| < sw_{crit} (= v \|f_1\|)$ **then**

$$\varphi_k = 0$$

else

$$\varphi_k = 1$$

Select algorithms for update the residual \hat{S}_k

ALG 1: DBFGS (switching DBFGS-DB)

ALG 2: DFN-BFGS (switching DFN-BFGS-DB)

ALG 3: MBFGS (switching MBFGS-DB)

ALG 4: Fu Method (switching Fu-DB)

end if

Calculate $\hat{H}_{s,k}$

$$\hat{H}_k = \hat{J}_k^T \hat{J}_k + \varphi_k \hat{S}_k$$

Calculate: θ_{k+1} using the dynamic full quasi-Newton method

$$\theta_{k+1} = \theta_k - \left(\hat{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

end for

Figure 6.2: Pseudo-code for the switching DBFGS-DB, DFN-BFGS-DB, MBFGS-DB, and Fu-DB algorithms

settling time t_s but a higher RMS tracking error compared to the DBM-RLS algorithm as shown in Table 6.5. One important observation is that all algorithms initially move the EE away from the desired trajectory due to insufficient information available at the time for the control system to properly learn about its environment. This is the nature of uncalibrated visual control since neither the robot kinematics nor the camera models are known. The EE motion on the image space and task space views in Figure 6.3e and Figure 6.4e shows that the MBFGS-DB algorithm quickly converges to the desired target compared to other algorithms. This result is confirmed in Figure 6.5 in which the MBFGS-DB algorithm starts moving toward the desired target at a lower image error norm. Furthermore, the MBFGS-DB method yields the largest step size and that in fact results in a faster settling time.

6.3.2 The Effect of Partitioned and Non-Partitioned Broyden’s Estimator for Approximating the Jacobian

The DBM-RLS algorithm in which the non-partitioned (NP) Broyden’s estimator is used to approximate \hat{J}_k generates better results than the DGN-PBM algorithm with the partitioned (P) Broyden’s method. The NP-Jacobian estimator residual approximation in Table 6.5 is compared to the P-Jacobian estimation in Table 6.6. Results from the switching algorithms using NP-Jacobian estimation are shown in Table 6.6.

The prefix “NP” is used to distinguish the switching algorithms using non-partitioned Broyden’s estimator to calculate \hat{J}_k the partitioned are where Jacobian \tilde{J} is utilized. Figure 6.6 compares the switching MBFGS-DB and the switching NP-MBFGS-DB algorithms. Results for the other switching algorithms are similar.

The NP Jacobian estimation provides better RMS tracking error with faster settling time for most residual approximations except the switching NP-MBFGS-DB algorithm where the settling time t_s is slightly longer. Since these algorithms provide similar results, a faster angular speed $\omega = 0.9$ rad/s of the circular target trajectory

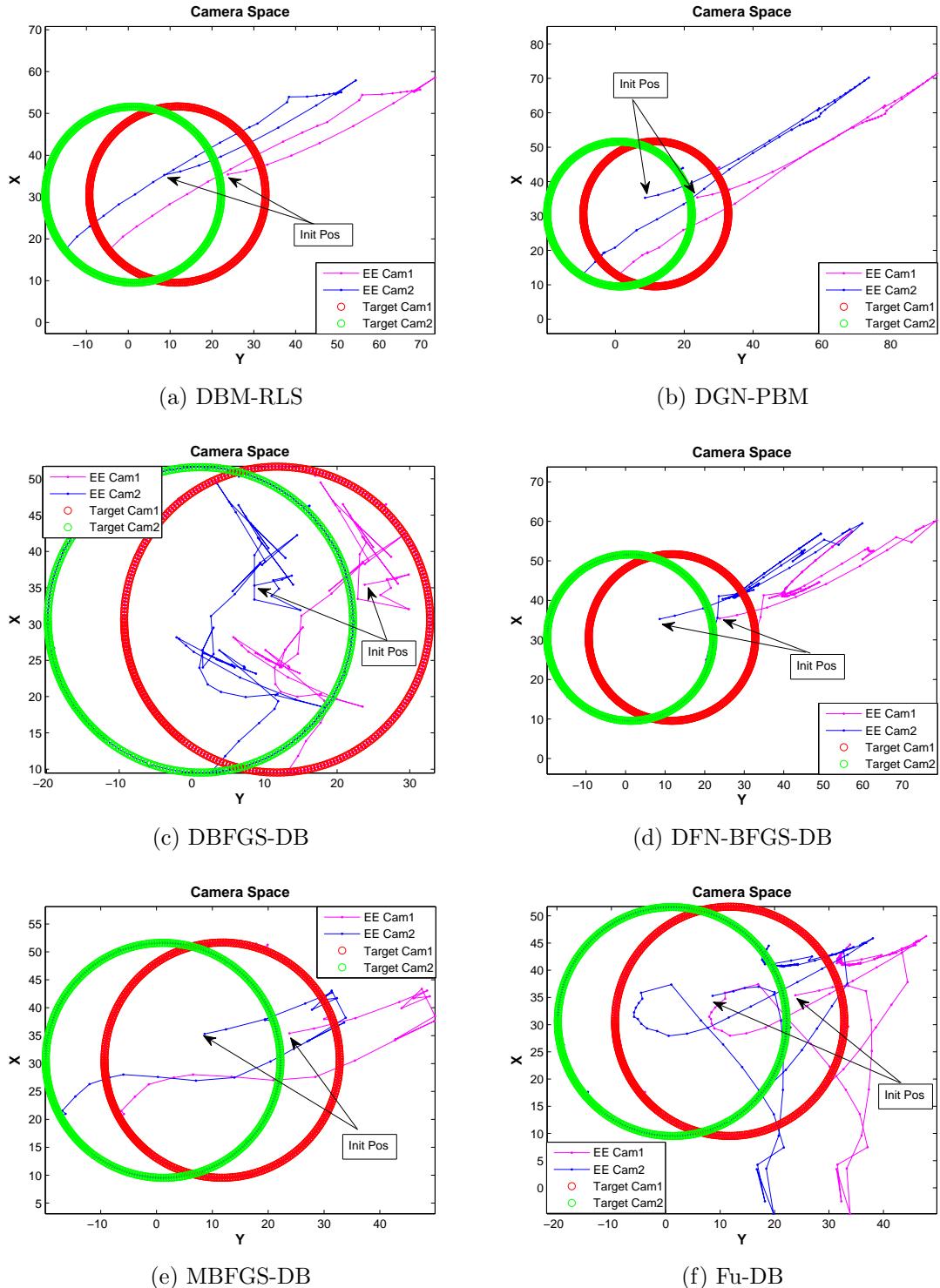


Figure 6.3: The camera space of the RRR manipulator with two eye-to-hand camera configurations tracks a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ for all algorithms except the switching Fu-DB where $v = 0.5$ is used.

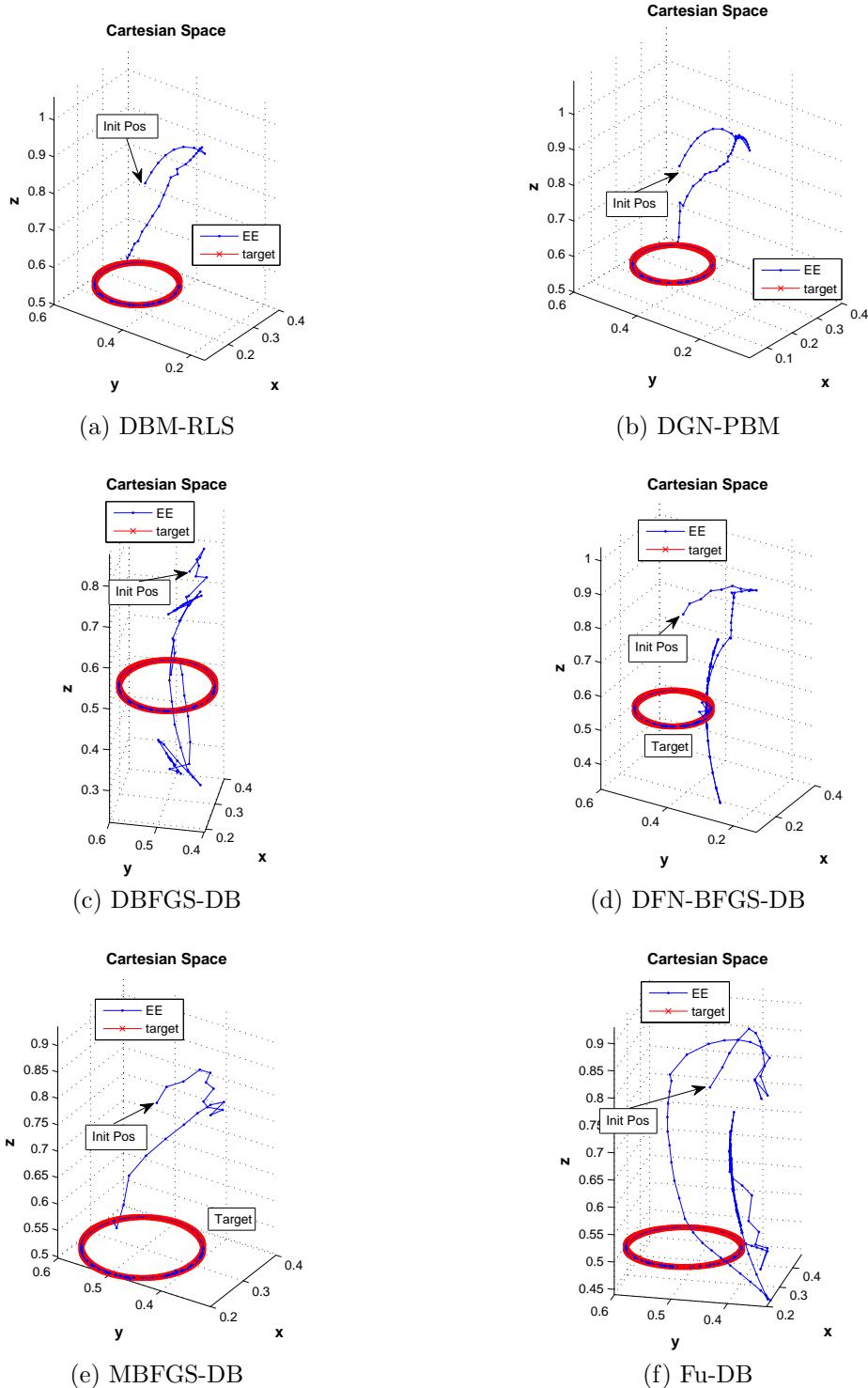


Figure 6.4: The task space view showing one camera and one target point for clarity (the others are similar) for the RRR manipulator with two eye-to-hand cameras tracking a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ for all algorithms except the switching Fu-DB where $v = 0.5$ is used.

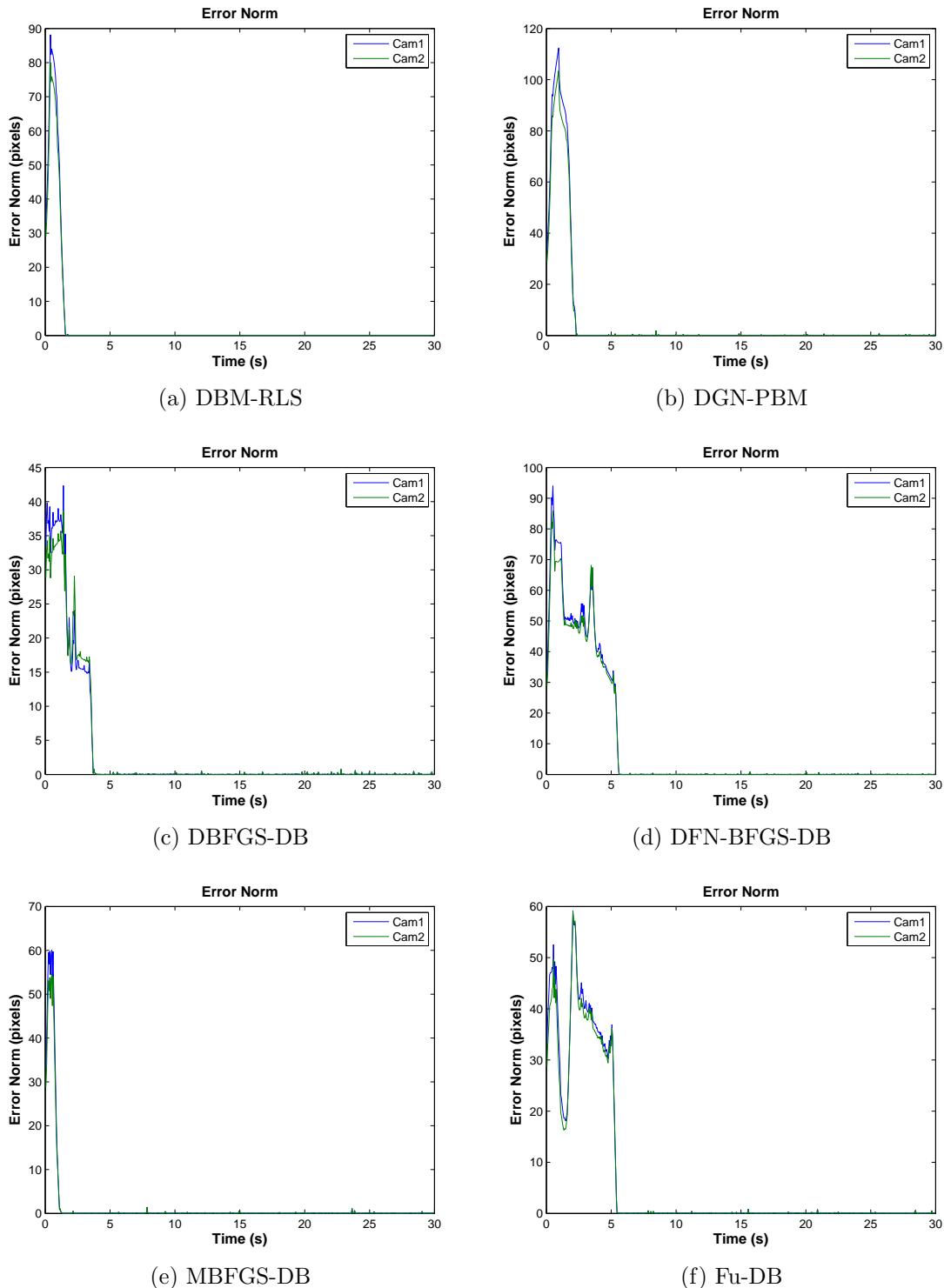


Figure 6.5: The error norm of the RRR manipulator with two eye-to-hand camera configurations tracks a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ for all algorithms except the switching Fu-DB where $v = 0.5$ is used.

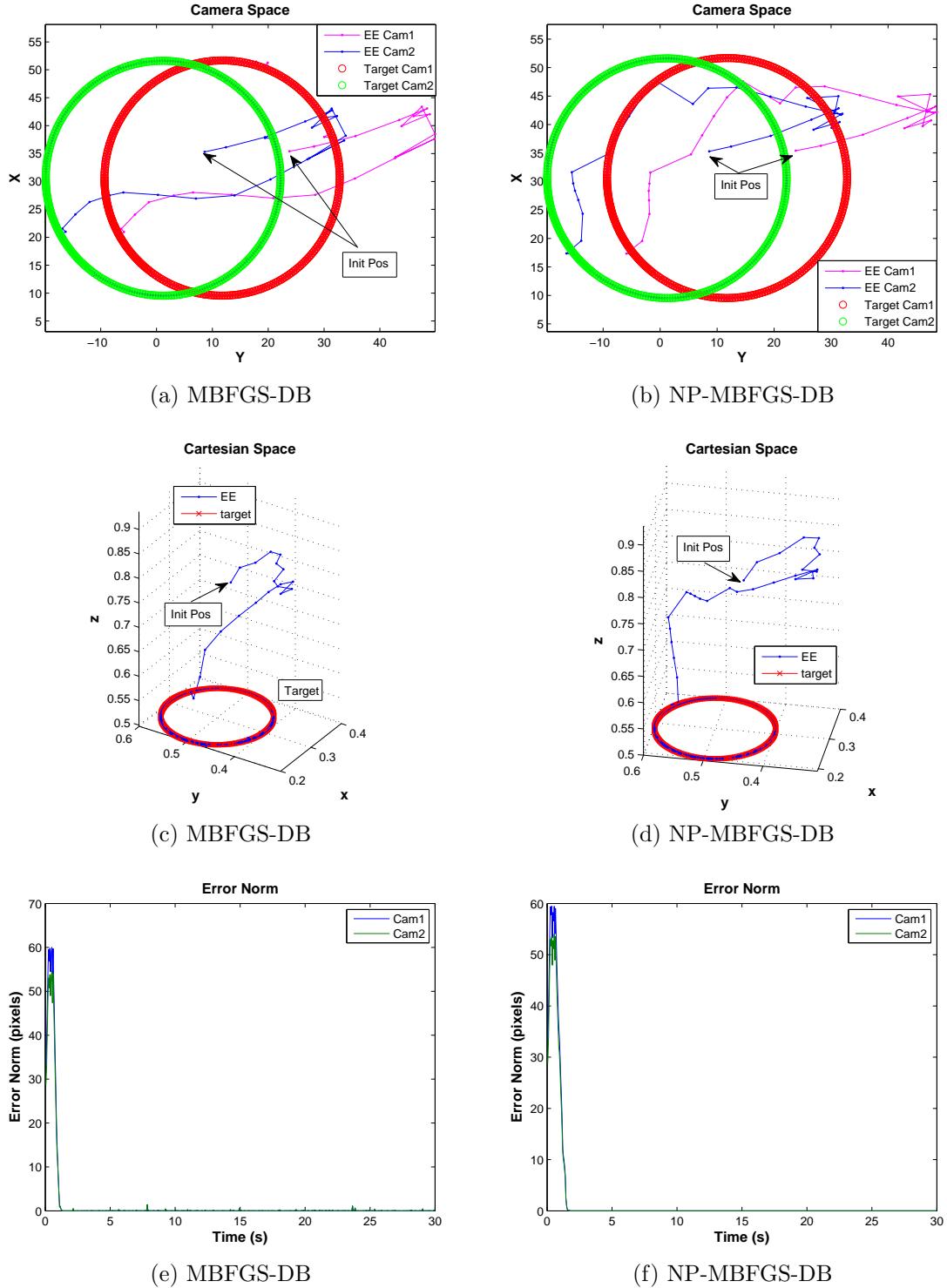


Figure 6.6: The performance comparison between the switching MBFGS-DB and NP-MBFGS-DB algorithms (a)-(b) the image plane, (c)-(d) the task space view, and (e)-(f) error norm of the RRR manipulator using two eye-to-hand cameras tracking a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$

Table 6.6: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes using NP-Jacobian estimation for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\omega = 0.45 \text{ rad/s}$, $\lambda = 0.5$, and $v = 0.3$ for all algorithms except the Fu-DB algorithm where $v = 0.5$ is used.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
DBFGS-DB	0.0111	0.0111	1.2
DFN-BFGS-DB	0.0143	0.0144	1.3
MBFGS-DB	0.0130	0.0131	1.6
Fu-DB	0.0110	0.0110	1.2

is tested with the NP-Jacobian estimation. The RMS error and settling time are summarized in Table 6.7.

Table 6.7: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes using NP-Jacobian estimation for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\omega = 0.9 \text{ rad/s}$, $\lambda = 0.5$, and $v = 0.3$ for all algorithms.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
DBM-RLS	0.0430	0.0430	1.7
DBFGS-DB	0.0427	0.0427	1.8
DFN-BFGS-DB	0.0431	0.0431	1.5
MBFGS-DB	0.0428	0.0428	1.7
Fu-DB	0.0485	0.0486	1.5

Results from Table 6.7 show that all switching algorithms using the NP-Jacobian approximation generate similar tracking performances. However, for an eye-in-hand camera configuration the dynamic error term $\frac{\partial f_k}{\partial t}$ cannot be estimated using a simple first order difference as in the eye-to-hand camera case since the moving target features

y^* is now a function of both robot joint angles θ and time t . Thus the dynamic term can only be approximated using the partitioned Broyden's method. In that case the switching MBFGS-DB algorithm is necessary. To validate tracking performance of the eye-in-hand camera case the switching algorithms are tested using the PUMA 560 robot in Section 6.3.4.

6.3.3 The Effect of Switching Criterion

The effectiveness of each switching algorithm is considerably influenced by the switching criterion sw_{crit} . This value determines when the residual approximation is included into the controller. Too great a value leads to an insufficient number of iterations with the inclusion of the residual calculation causing a small step size and a longer settling time. Too small a value creates redundancy in inclusion of a residual term that often leads to a longer settling time and divergence in the worse case.

Instability occurs when the robot joint angles θ_{k+1} are calculated using the residual approximation algorithm for the small image error case. This is because residual approximation \hat{S}_k is recursively calculated from the previous value \hat{S}_{k-1} , and the current \hat{S}_k may not be diminished even if the actual residual is zero or becomes relatively small. Thus, the estimation of θ_k is calculated with extraneous information that does not represent the true nature of the system and results in deteriorated tracking performance. One solution to this problem is the switching approach in which the residual approximation \hat{S}_k is only included into the calculation of θ_{k+1} if the error norm $\|f_k\|$ is greater than a specified switching criterion,

$$sw_{crit} = v \|f_1\|$$

where $v \geq 0$.

Newton's method with residual \hat{S}_k approximation has been proposed by Dennis et al. [17, 19] who suggest updating \hat{S}_k from a scaled $\varsigma \hat{S}_{k-1}$ where $\varsigma \leq 1$ (this algorithm is used in Fu's method [25] and the switching Fu-DB algorithm.) The scaling factor ς

is introduced so that the residual term is properly terminated when f_k becomes zero,

$$\varsigma = \min \left\{ \frac{z_{k-1}^{*T} h_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}}, 1 \right\}$$

so that

$$\hat{S}_{k-1} = \varsigma \hat{S}_{k-1}$$

Then \hat{S}_k is updated as

$$\hat{S}_k = \hat{S}_{k-1} + \frac{z_{k-1}^{*T} z_{k-1}^T}{g_{k-1}^{*T} h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}}$$

where

$$z_{k-1}^* = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k$$

$$g_{k-1} = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_{k-1}$$

This strategy is implemented into the NL2SOL algorithm and details are presented in [20]. This strategy is implemented into the switching MBFGS-DB algorithm shown in the pseudo-code in Figure 6.7 and is referred to as the *MBFGS-DB Scheme 1*.

Unlike switching algorithms in which the residual term is either included into or excluded from the controller, [49] proposed a hybrid method to systematically calculate a weighted average between the Gauss-Newton ($\hat{J}_k^T \hat{J}_k$) and the residual (\hat{S}_k) terms for Hessian \hat{H}_k approximation (Section 4.7),

$$\hat{H}_k = J_k^T J_k + (1 - \kappa_k) \hat{S}_k$$

This hybrid method is only briefly mentioned, without details, in [49] that κ_k is updated based on how well the current quadratic model of the objective function F_k can be trusted. Inspired by this idea, an analogous algorithm is used to readjust the trust-region size presented in [25] for calculating κ_k .

Pseudo-code: The MBFGS-DB Scheme 1

Given: $\mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$, $P_0 \in \mathbb{R}^{n \times n}$, $\lambda \in (0, 1)$

Initialize: \hat{J}_0 , θ_0 , θ_1 , H_0 and P_0

for $k = 1, \dots$ **do**

Calculate: \hat{J}_k using a Jacobian estimation in the DGN-PBM algorithm [59]

$$\begin{aligned}\Delta f &= f_k - f_{k-1}; \quad h_{k-1} = \theta_k - \theta_{k-1} \\ \tilde{h} &= \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix} \\ \tilde{J}_{k-1} &= \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix} \\ \tilde{J}_k &= \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1} \\ \tilde{P}_k &= \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right)\end{aligned}$$

Calculate the scaling ς

$$\varsigma = \min \left\{ \frac{z_{k-1}^{*T} h_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}}, 1 \right\}$$

Update \hat{S}_{k-1}

$$\hat{S}_{k-1} = \varsigma \hat{S}_{k-1}$$

Update the residual \hat{S}_k

$$\begin{aligned}\hat{S}_k &= \hat{S}_{k-1} + \frac{z_{k-1}^{*} z_{k-1}^{*T}}{g_{k-1}^{*T} h_{k-1}} - \frac{\hat{S}_{k-1} h_{k-1} h_{k-1}^T \hat{S}_{k-1}}{h_{k-1}^T \hat{S}_{k-1} h_{k-1}} \\ z_{k-1}^{*} &= \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_k; \quad g_{k-1} = \hat{J}_k^T f_k - \hat{J}_{k-1}^T f_{k-1}\end{aligned}$$

Calculate $\hat{H}_{s,k}$

$$\hat{H}_k = \hat{J}_k^T \hat{J}_k + \hat{S}_k$$

Calculate: θ_{k+1} using the hybrid method proposed by [49]

$$\theta_{k+1} = \theta_k - \left(\hat{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

end for

Figure 6.7: Pseudo-code for the MBFGS-DB Scheme 1 using a similar scaling method presented in [17, 19]

The value of κ_k is determined according to the ratio r between the actual reduction of the function F_k and the predicted value obtained from the quadratic model q_k , is

$$q_k = \frac{1}{2} f_k^T f_k + (J_k^T f_k)^T h_{k-1} + \frac{1}{2} h_{k-1}^T (J_k^T J_k) h_{k-1}$$

The actual reduction of the objective function is

$$\Delta F = F_{k-1} - F_k$$

while, the predicted reduction is

$$\Delta q = F_k - q_{k+1}$$

yielding the ratio r as

$$r = \frac{\Delta F}{\Delta q}$$

Based on the value of r , κ_k is determined as

$$\kappa_k = \begin{cases} 0 & \text{if } r \leq 0.25 \\ 1 & \text{if } r \geq 0.75 \\ 0.5 & \text{if otherwise} \end{cases}$$

This method is integrated with the MBFGS algorithm and is referred to as the *MBFGS-DB Scheme 2*. A summary of MBFGS-DB Scheme 2 is shown in the pseudo-code in Figure 6.8.

To investigate the effectiveness of these two schemes, they are compared with the switching MBFGS-DB algorithm. Since the NP Broyden's method demonstrates better RMS tracking error with fast convergence compared to the partitioned Jacobian approximation as discussed in Section 6.3.2, both partitioned (P) and the non-partitioned (NP) Jacobian approximations are tested with MBFGS-DB Schemes 1 and 2.

Since all tests use the MBFGS method to, the “MBFGS” term is dropped. Table 6.8 shows a summary of the algorithms being tested in this section. The RMS tracking

Pseudo-code: The MBFGS-DB Scheme 2

Given: $\mathbb{R}^n \rightarrow \mathbb{R}^m$; $\theta_0, \theta_1 \in \mathbb{R}^n$; $\hat{J}_0 \in \mathbb{R}^{m \times n}$, $P_0 \in \mathbb{R}^{n \times n}$, $\lambda \in (0, 1)$

Initialize: \hat{J}_0 , θ_0 , θ_1 , H_0 and P_0

for $k = 1, \dots$ **do**

Calculate: \hat{J}_k using a Jacobian estimation in the DGN-PBM algorithm [59]

$$\begin{aligned}\Delta f &= f_k - f_{k-1}; \quad h_{k-1} = \theta_k - \theta_{k-1} \\ \tilde{h} &= \begin{bmatrix} (\theta_k - \theta_{k-1}) \\ (t_k - t_{k-1}) \end{bmatrix} \\ \tilde{J}_{k-1} &= \begin{bmatrix} \hat{J}_{k-1} & (\hat{f}_t)_{k-1} \end{bmatrix} \\ \tilde{J}_k &= \tilde{J}_{k-1} + \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\Delta f - \tilde{J}_{k-1} \tilde{h} \right) \tilde{h}^T \tilde{P}_{k-1} \\ \tilde{P}_k &= \frac{1}{\lambda} \left(\tilde{P}_{k-1} - \left(\lambda + \tilde{h}^T \tilde{P}_{k-1} \tilde{h} \right)^{-1} \left(\tilde{P}_{k-1} \tilde{h} \tilde{h}^T \tilde{P}_{k-1} \right) \right)\end{aligned}$$

Calculate the ratio r between the actual and the predicted reduction of the function

$$\begin{aligned}q_k &= \frac{1}{2} f_k^T f_k + (J_k^T f_k)^T h_{k-1} + \frac{1}{2} h_{k-1}^T (J_k^T J_k) h_{k-1} \\ F_k &= \frac{1}{2} f_k^T f_k \\ \Delta F &= F_{k-1} - F_k \\ \Delta q &= F_k - q_k \\ r &= \frac{\Delta F}{\Delta q}\end{aligned}$$

Calculate the weighting average ϕ_k

$$\kappa_k = \begin{cases} 0 & \text{if } r \leq 0.25 \\ 1 & \text{if } r \geq 0.75 \\ 0.5 & \text{if otherwise} \end{cases}$$

Update the residual \hat{S}_k using the MBFGS method

Calculate $\hat{H}_{s,k}$

$$\hat{H}_k = J_k^T J_k + (1 - \kappa_k) \hat{S}_k$$

Calculate: θ_{k+1} using the hybrid method proposed by [49]

$$\theta_{k+1} = \theta_k - \left(\hat{H}_k \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

end for

Figure 6.8: Pseudo-code for the MBFGS-DB Scheme 2 using a hybrid method similar to [49]

error and the settling time t_s are compared as shown in Table 6.9. The camera space and the task space views are shown in Figure 6.9 and Figure 6.10 respectively.

Table 6.8: A summary of the MBFGS algorithm varying in Jacobian approximation and strategies for \hat{S}_k inclusion using the RRR robot with two eye-to-hand cameras starting at $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$ with $\lambda = 0.5$.

Scheme	Jacobian Method	Inclusion of \hat{S}_k
P-Switching	P	Switching $v = 0.3$
NP-Switching	NP	Switching $v = 0.3$
P-Scheme 1	P	Scheme 1
NP-Scheme 1	NP	Scheme 1
P-Scheme 2	P	Scheme 2
NP-Scheme 2	NP	Scheme 2

From Table 6.9 the NP Jacobian estimator yields a smaller RMS error compared to the partitioned one. Although the RMS error and settling time of the NP-Scheme 1 is similar to the NP-Switching algorithms, Figure 6.10 shows that both P- and NP-Scheme 1 generate EE motion that deviates from the desired target plane. However, the NP-Scheme 1 causes less deviation compared to the P-Scheme 1. The NP-Scheme 2 gives about the same RMS error but longer settling time compared to others as can be seen on Figure 6.9. From these results, the switching algorithm either using P- or NP-Jacobian approximation yields more desirable results compared to the Schemes 1 and 2.

The results of using the RRR robot with two eye-to-hand cameras to track one feature point of the circular trajectory target suggest that the switching DFN-BFGS-DB and the MBFGS-DB algorithms are the most effective methods over the other

schemes. These results are improved if the NP-Jacobian approximation is utilized. The heuristic selection of the switching criterion v demonstrates better RMS tracking error and settling time as compared to the other existing algorithms (Schemes 1 and 2). To further investigate the performance of the switching algorithms a 6 DOF robot is used in Section 6.3.4.

Table 6.9: The RMS error and the settling time comparison between the switching, Scheme 1, and Scheme 2 MBFGS-DB algorithms with/without partitioned Broyden's method.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
P-Switching	0.1059	0.1054	1.3
NP-Switching	0.0130	0.0131	1.6
P-Scheme 1	0.3639	0.3401	1.8
NP-Scheme 1	0.0405	0.0414	1.7
P-Scheme 1	0.1110	0.1101	2.8
NP-Scheme 1	0.0108	0.0108	2.4

6.3.4 The PUMA 560 Robot

A six DOF Puma 560 manipulator with one eye-in-hand camera (using the MATLAB Robotics and Machine Vision Toolboxes [10]) is used in this section for validating the proposed switching algorithms with different camera configurations. The intrinsic parameters of the camera are the same as in Table 6.3. The target consists of four feature points at the vertices of a 50 mm square. To make the end-effector and target trajectories visually distinct they are offset by a constant vector $[-393.7, 19.9, 142]^T$ mm. The starting robot joint angles are $\theta_0 = [15.73^\circ, 132.5^\circ, -135.6^\circ, -4.27^\circ, -108.75^\circ, 14.27^\circ]^T$ for all tests simulated in this section. The end-effector camera has a 10 mm focal length and is coincident with the final frame of the robot. The sampling time is

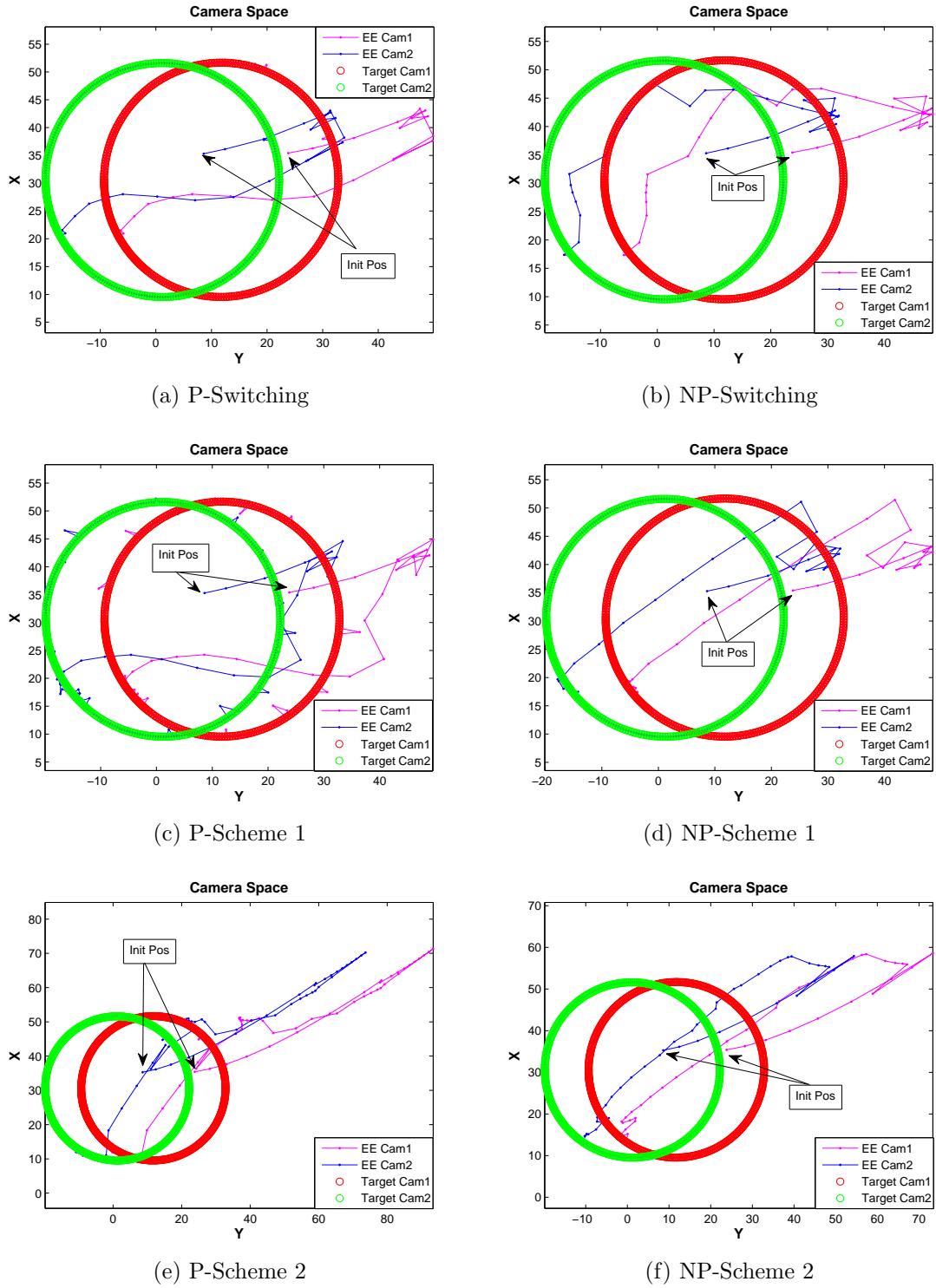


Figure 6.9: The camera space of the RRR manipulator with two eye-to-hand cameras using (a) P-Switching ($v = 0.3$), (b) NP-Switching ($v = 0.3$), (c) P-Scheme 1, (d) NP-Scheme 1, (e) P-Scheme 2, and (f) NP-Scheme 2. A circular target trajectory is moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$.

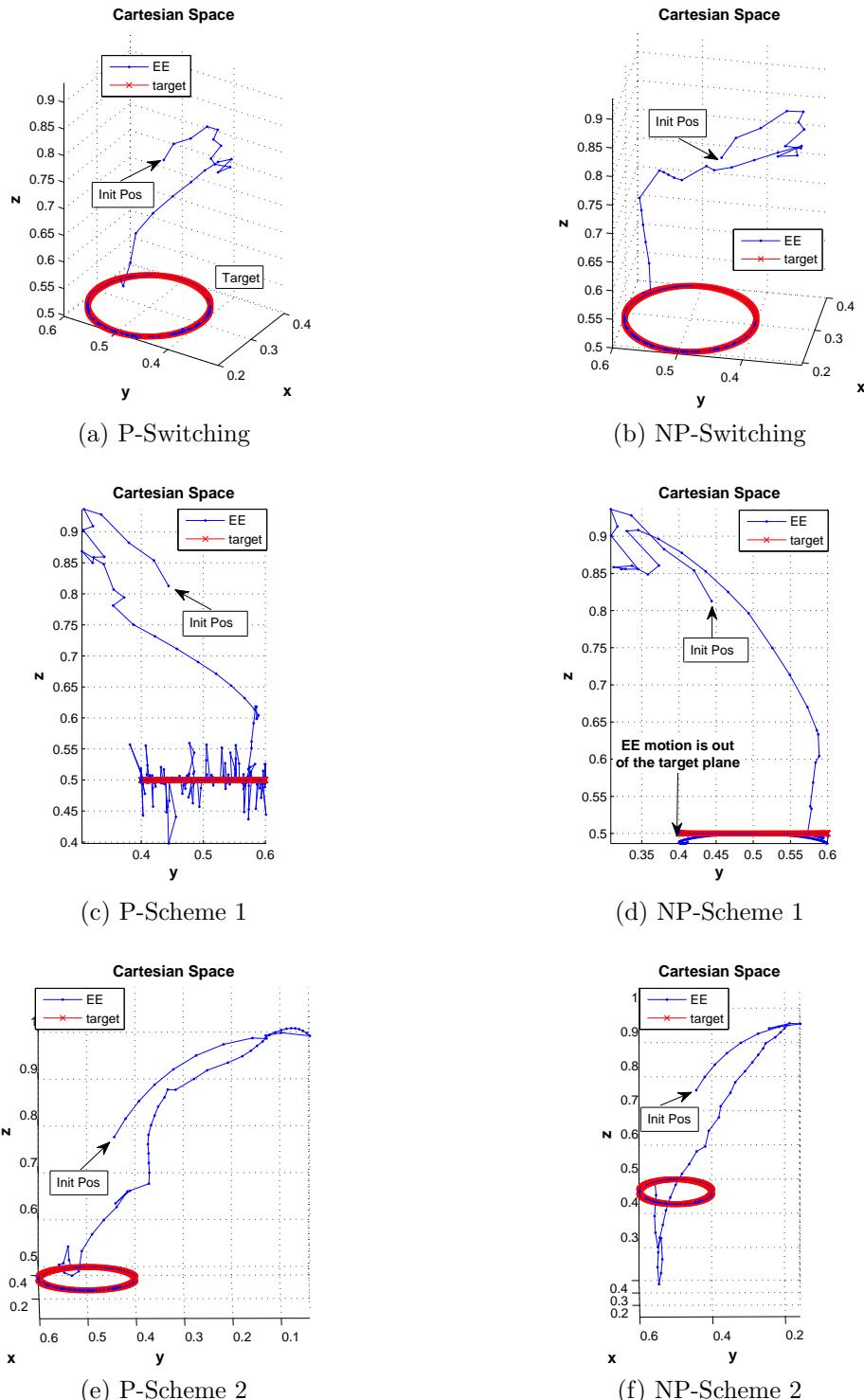


Figure 6.10: The task space view showing one camera and one target point using (a) P-Switching ($v = 0.3$), (b) NP-Switching ($v = 0.3$), (c) P-Scheme 1, (d) NP-Scheme 1, (e) P-Scheme 2, and (f) NP-Scheme 2. A circular target trajectory is moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$

$h_t = 50$ ms. The initial Jacobian is estimated by successively perturbing each joint by a small angle.

Three translational target trajectories are investigated in this section,

1. Circular trajectory
2. Cycloidal trajectory
3. Helical trajectory

Circular Trajectory

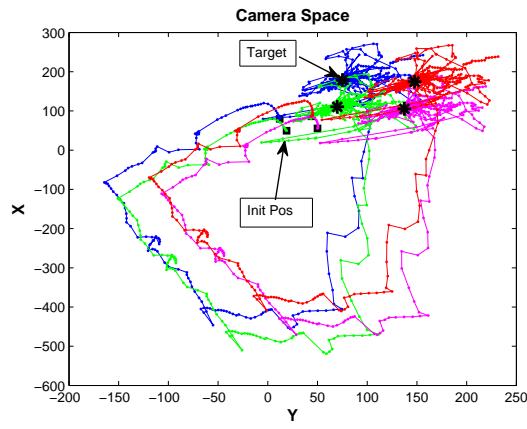
A circular trajectory has the radius of $R = 100$ mm with an angular velocity of $\omega = 0.45$ rad/s and is generated by

$$\begin{aligned} x &= 600 && \text{mm} \\ y &= -150 + R \sin(\omega kT) && \text{mm} \\ z &= 400 + R \cos(\omega kT) && \text{mm} \end{aligned} \quad (6.4)$$

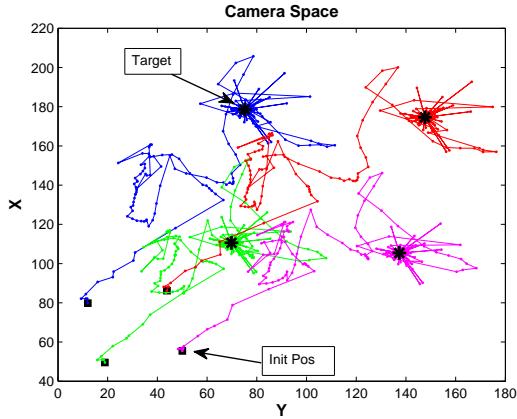
where k is the iteration number and $T = 50$ ms is the sampling period. The target motion starts from the top of the circle and moves counterclockwise.

For each switching scheme the image view, the task space view, and the error norm are shown in Figure 6.11, Figure 6.12, and Figure 6.13 respectively. These are generated by using a fixed forgetting factor $\lambda = 0.5$ and $v = 0.3$ for all switching algorithms except for the switching DBFGS-DB algorithm where $v = 0.55$ is used to avoid tracking failure. The RMS tracking error and the settling time t_s are summarized in Table 6.10.

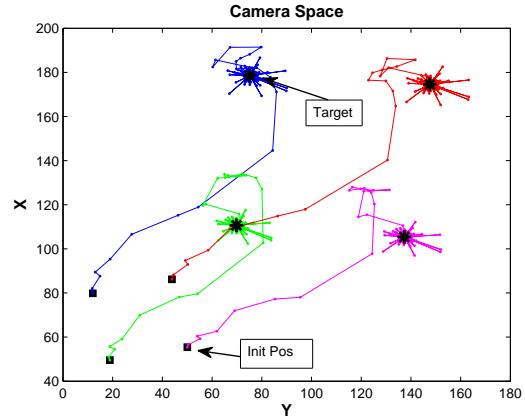
The switching MBFGS-DB, DFN-BFGS-DB, and Fu-DB algorithms generate similar EE trajectories on the camera and task space as shown in Figure 6.11 and Figure 6.12 and have significantly better RMS error and settling time values over the DGN-PBM algorithm (Table 6.10). Although the DBFGS-DB gives a longer settling time



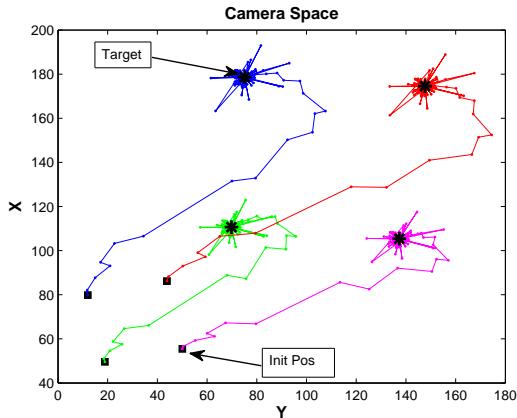
(a) DGN-PBM



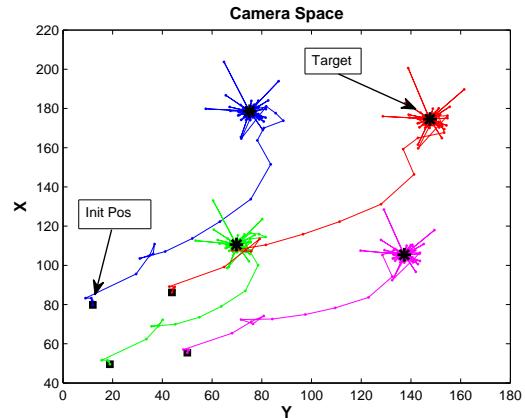
(b) DBFGS-DB



(c) DFN-BFGS-DB

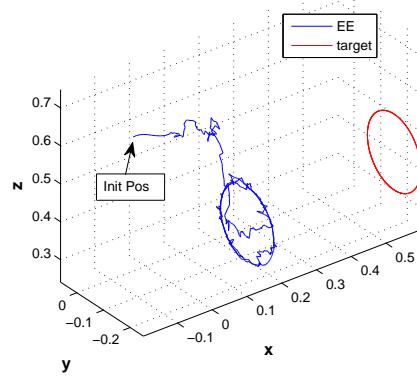


(d) MBFGS-DB

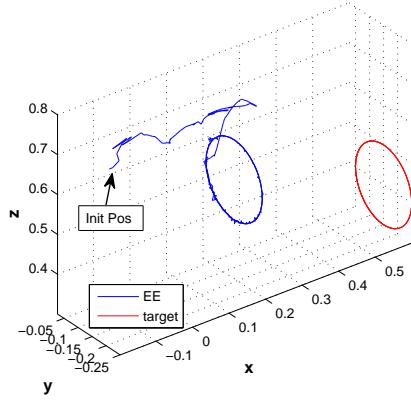


(e) Fu-DB

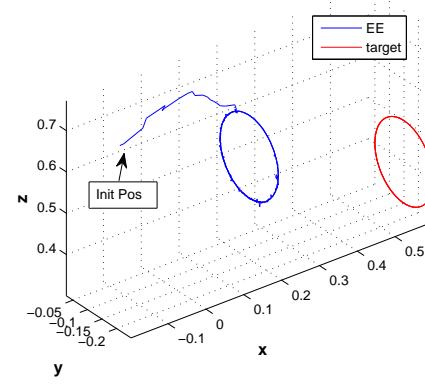
Figure 6.11: The camera space of the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ (except the DBFGS-DB $v=0.55$).



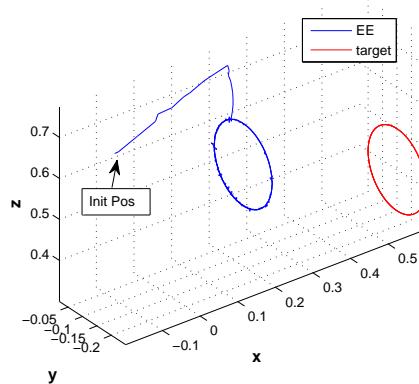
(a) DGN-PBM



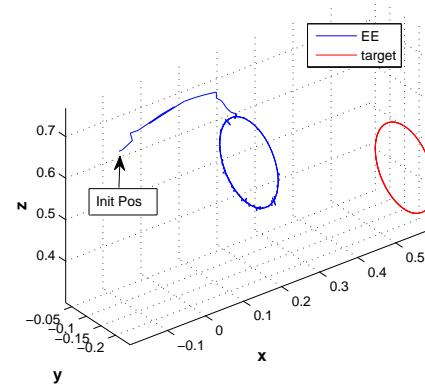
(b) DBFGS-DB



(c) DFN-BFGS-DB

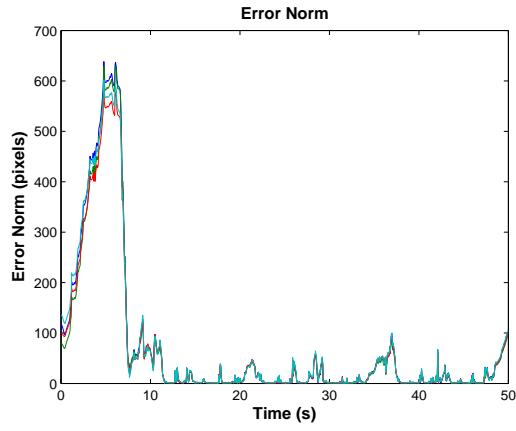


(d) MBFGS-DB

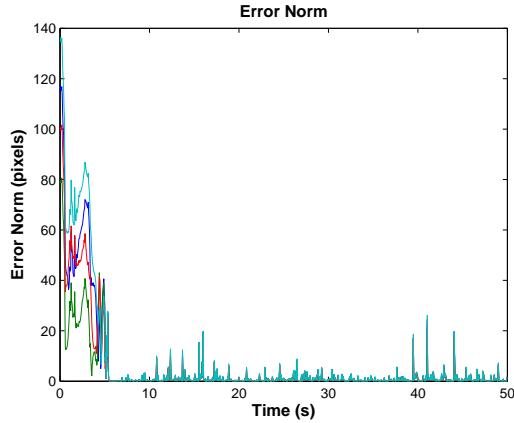


(e) Fu-DB

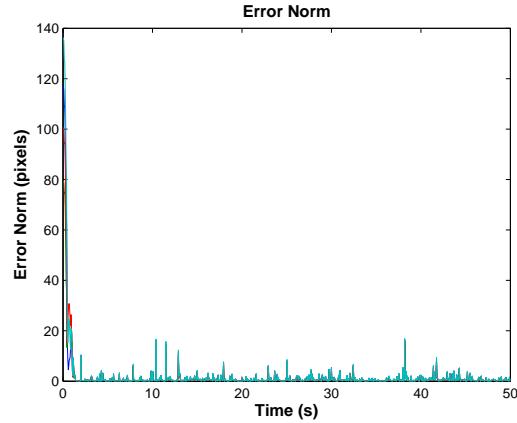
Figure 6.12: The task space view showing camera and one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ (except the DBFGS-DB $v=0.55$).



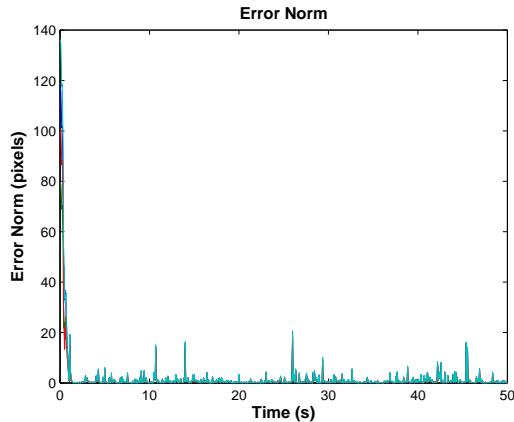
(a) DGN-PBM



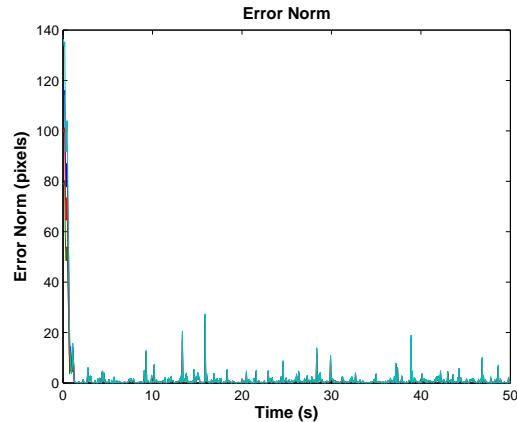
(b) DBFGS-DB



(c) DFN-BFGS-DB



(d) MBFGS-DB



(e) Fu-DB

Figure 6.13: The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$ (except the DBFGS-DB $v=0.55$).

Table 6.10: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the circular trajectory using a fixed forgetting factor $\lambda = 0.5$.

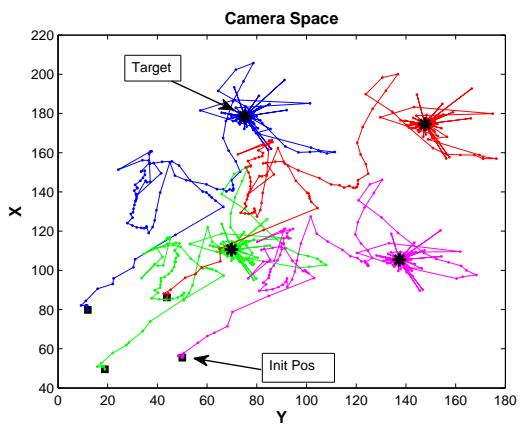
Scheme	RMS Error (pixels)	t_s (sec)
DGN-PBM	22.5085	12
DBFGS-DB	1.9956	5.5
DFN-BFGS-DB	1.4974	1.5
MBFGS-DB	1.6401	1.5
Fu-DB	1.7874	1.4

than the other switching methods, it provides improved tracking performance compared to the DGN-PBM algorithm as shown in Figure 6.13.

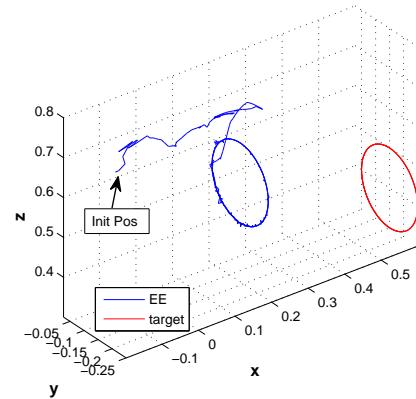
Note that for an eye-in-hand camera configuration the image error f_k is a function of both robot joint angles θ_k and time t in which the dynamic term $\frac{\partial f_k}{\partial t}$ can only be estimated using the partitioned Broyden's estimator. Therefore, the DBM-RLS algorithm and the switching algorithms using the NP Broyden's estimator are not included in this section.

To investigate alternative methods for including the residual \hat{S}_k , Scheme 1 and Scheme 2 are tested with the residual approximation methods. The image view and the task space view of the EE motion of the DBFGS-DB, DFN-BFGS-DB, MBFGS-DB, and Fu-DB using Scheme 1 and 2 are shown in Figure 6.14, Figure 6.15, Figure 6.16, and Figure 6.17 respectively.

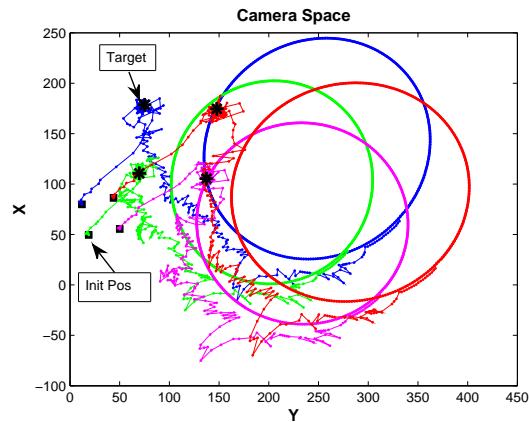
With all approaches Scheme 1 fails to converge if $\lambda = 0.5$ is used. Therefore, an alternating forgetting factor method is utilized in which $\lambda = 0.5$ is applied during the transient state and $\lambda = 0.98$ is employed if the EE reaches the steady-state tracking. The forgetting factor $\lambda = 0.5$ is applied for all residual approximations with Scheme 2.



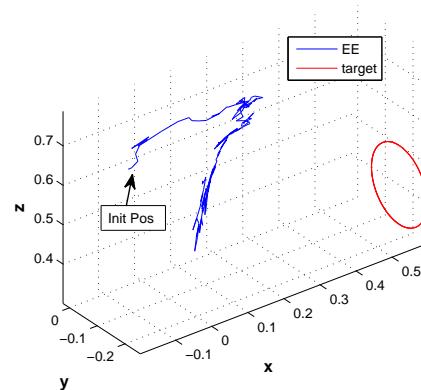
(a) Switching DBFGS-DB



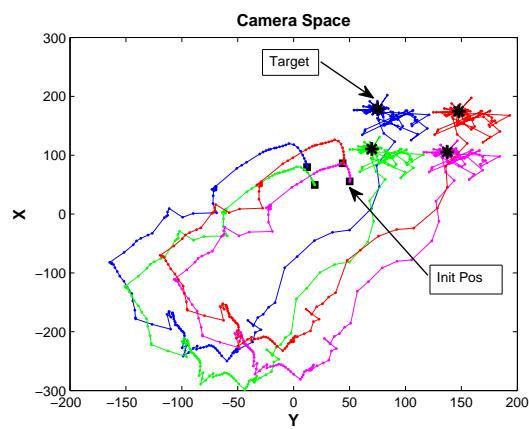
(b) Switching DBFGS-DB



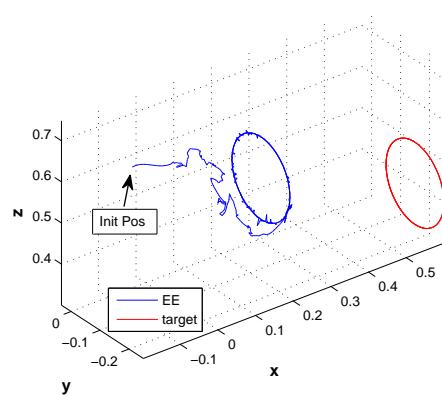
(c) DBFGS-DB Scheme 1



(d) DBFGS-DB Scheme 1

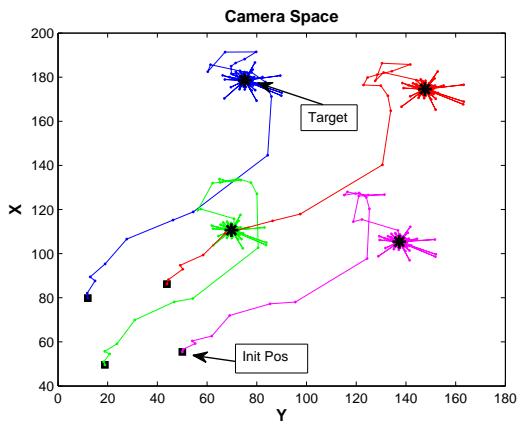


(e) DBFGS-DB Scheme 2

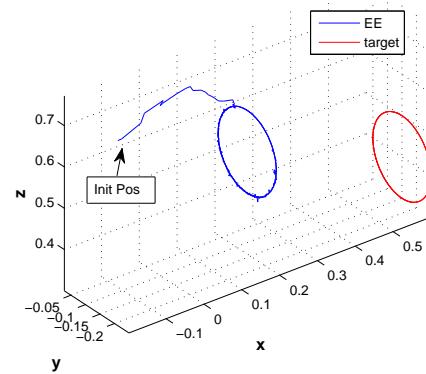


(f) DBFGS-DB Scheme 2

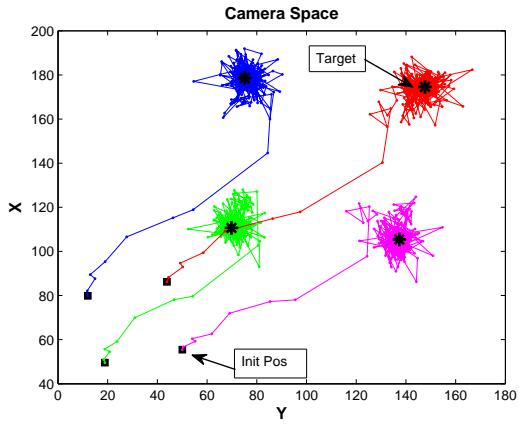
Figure 6.14: The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a circular trajectory using the DBFGS-DB with Scheme 1 and 2.



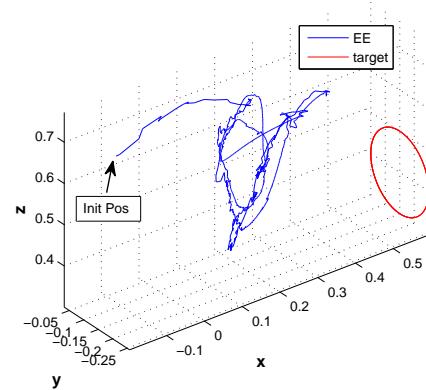
(a) Switching DFN-BFGS-DB



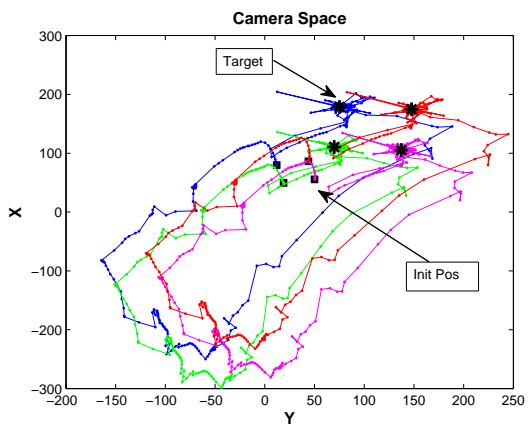
(b) Switching DFN-BFGS-DB



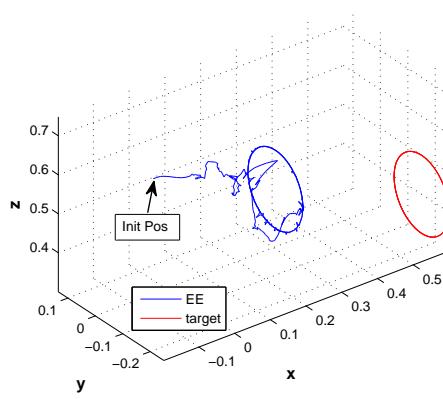
(c) DFN-BFGS-DB Scheme 1



(d) DFN-BFGS-DB Scheme 1

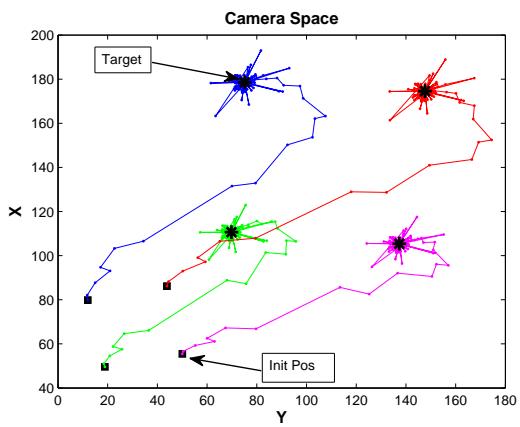


(e) DFN-BFGS-DB Scheme 2

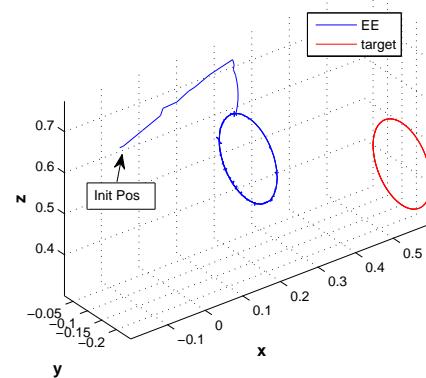


(f) DFN-BFGS-DB Scheme 2

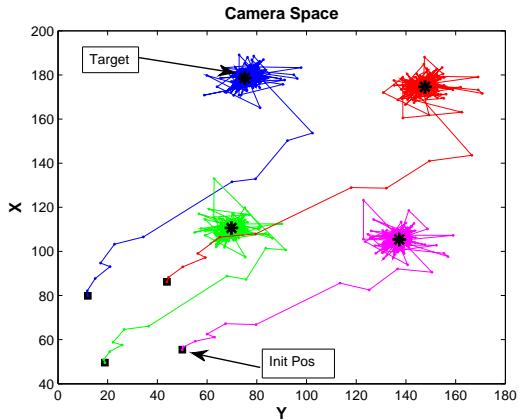
Figure 6.15: The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a circular trajectory using the DFN-BFGS-DB with Scheme 1 and 2.



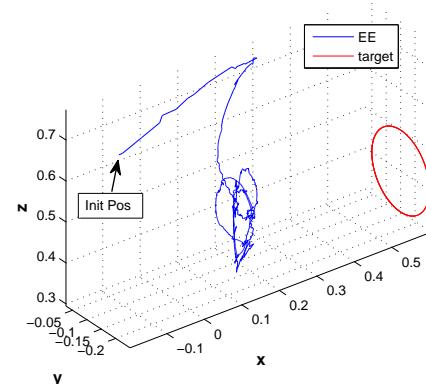
(a) Switching MBFGS-DB



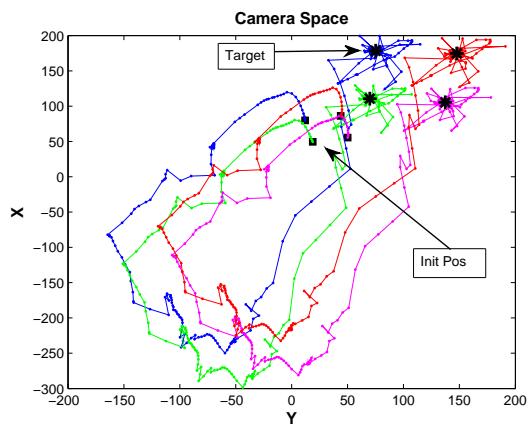
(b) Switching MBFGS-DB



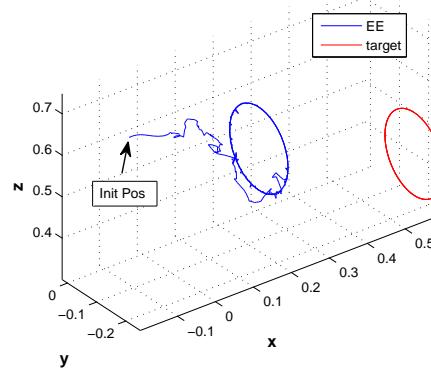
(c) MBFGS-DB Scheme 1



(d) MBFGS-DB Scheme 1



(e) MBFGS-DB Scheme 2



(f) MBFGS-DB Scheme 2

Figure 6.16: The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a circular trajectory using the MBFGS-DB with Scheme 1 and 2.

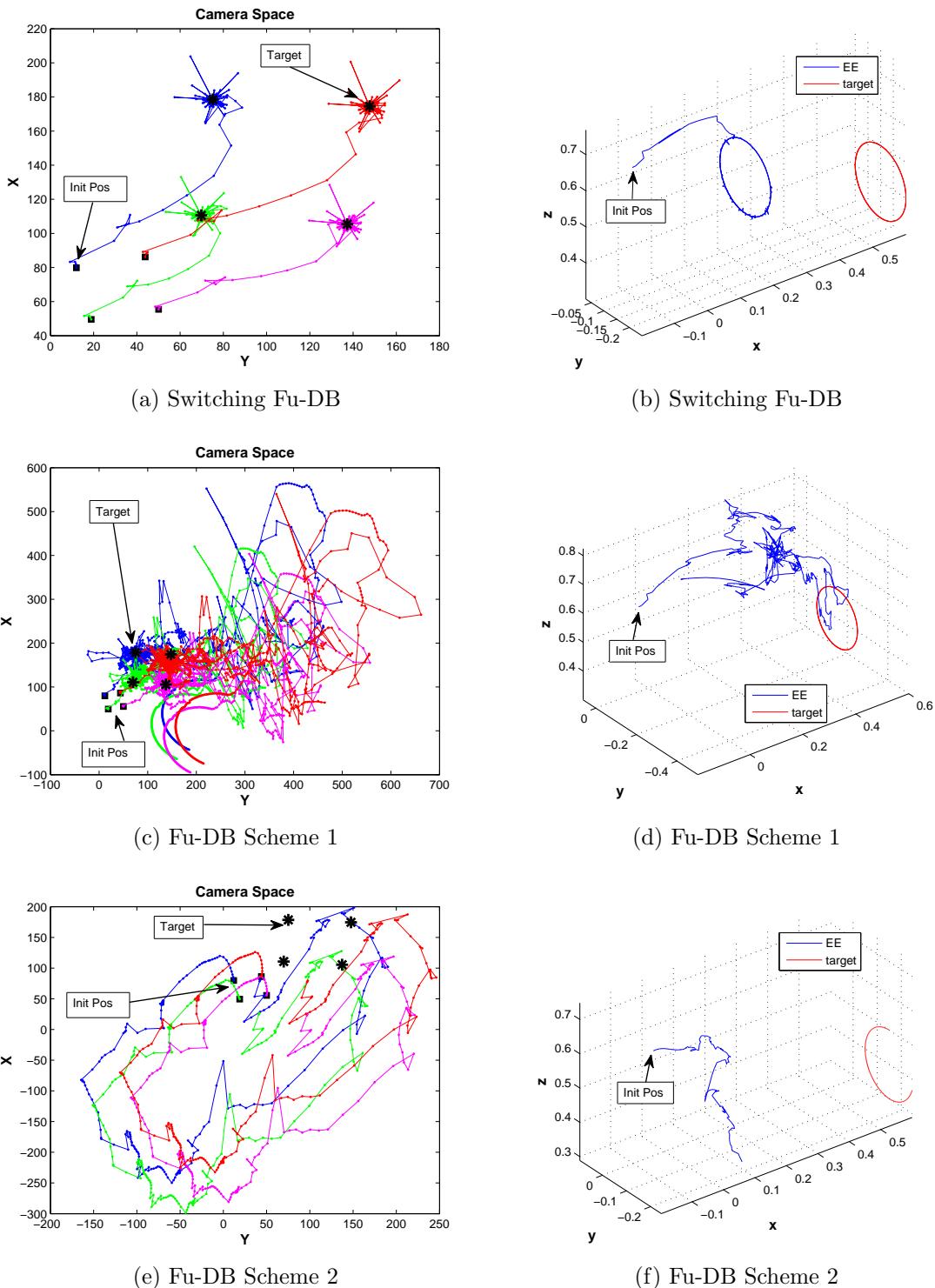


Figure 6.17: The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a circular trajectory using the Fu-DB with Scheme 1 and 2.

Table 6.11: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes with ALT Scheme 1 and 2.

Scheme	Alternate Method	RMS Error (pixels)	t_s (sec)
DGN-PBM	N/A	22.5085	12
DBFGS-DB	Switching Scheme 1	1.9956	5.5
	Scheme 2	2.4354	TF
			10
DFN-BFGS-DB	Switching Scheme 1	1.4974	1.5
	Scheme 2	4.0800	2
		3.9409	10
MBFGS-DB	Switching Scheme 1	1.6401	1.5
	Scheme 2	3.1594	2
		1.7570	10
Fu-DB	Switching Scheme 1	1.7874	1.4
	Scheme 2	TF	TF
		TF	TF

Table 6.11 summarizes the results. The DBFGS-DB with Scheme 1, the Fu-DB with the both Scheme 1 and 2 generate tracking failure (TF). As seen in Figure 6.14, the DBFGS-DB Scheme 1 initially moves the EE toward the desired trajectory but fails to follow the desired path and the robot eventually stops. The DBFGS-DB Scheme 2, on the other hand, converges and follows the target trajectory but its EE path is not as direct as the switching algorithm.

The DFN-BFGS-DB and MBFGS-DB with Scheme 1 and 2 offer similar tracking results where Scheme 1 generates a more direct path from the robot starting location to the target compared to Scheme 2 as shown in Figure 6.15 and Figure 6.16. Nevertheless, Scheme 1 has worse tracking accuracy than Scheme 2. The switching scheme delivers the most desirable RMS error and convergence time over Scheme 1 and 2.

¹Tracking failure

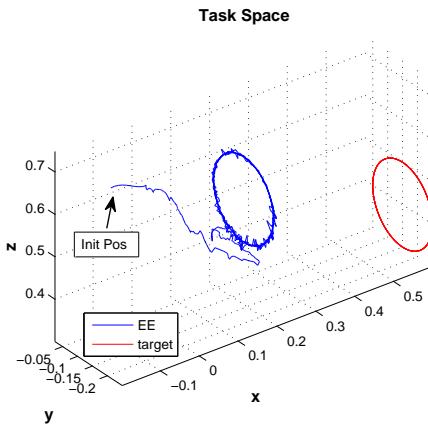
From Figure 6.17 neither Scheme 1 nor Scheme 2 succeed in tracking even though both methods initially move the EE toward the target. In contrast, the switching method has the fastest convergence and the smallest RMS tracking errors compared to Scheme 1 and Scheme 2.

Since the switching DFN-BFGS-DB, DBFGS-DB, and MBFGS-DB algorithms give similar results a the faster angular speed $\omega = 0.9$ rad/s is used. Table 6.12 summarizes the RMS error and t_s of each switching algorithm using $\lambda = 0.5$ while different values of v are required for convergence. The task space view and the average error norm are shown in Figure 6.18 and Figure 6.19 respectively.

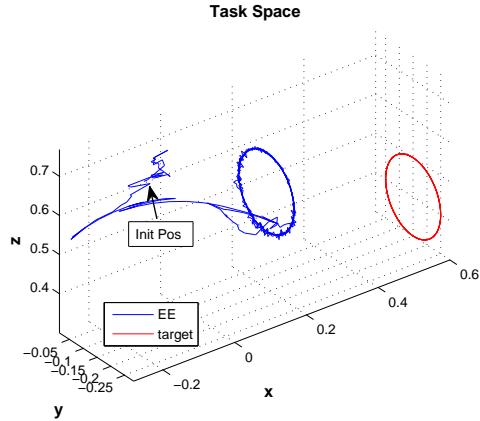
Table 6.12: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking four feature points of a circular trajectory moving at $\omega = 0.9$ rad/s using a fixed $\lambda = 0.5$ and v selected to ensure convergence.

Scheme	RMS Error (pixels)	t_s (sec)
DGN-PBM	5.5389	10
DBFGS-DB $v = 0.9$	4.9243	5
DFN-BFGS-DB $v = 0.3$	4.8742	1.5
MBFGS-DB $v = 0.3$	5.1801	1.5
Fu-DB $v = 1.3$	5.6324	8

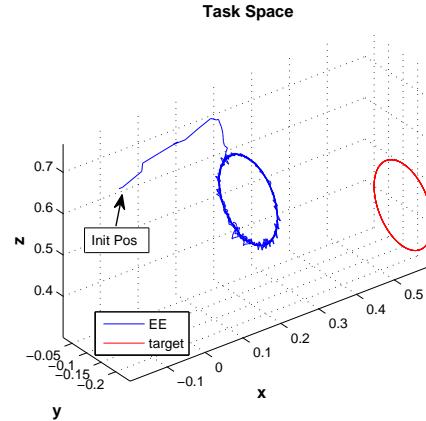
The switching DFN-BFGS-DB and MBFGS-DB provide similar results and outperforms the other algorithms. A more difficult path is then used to investigate the effectiveness of these algorithms.



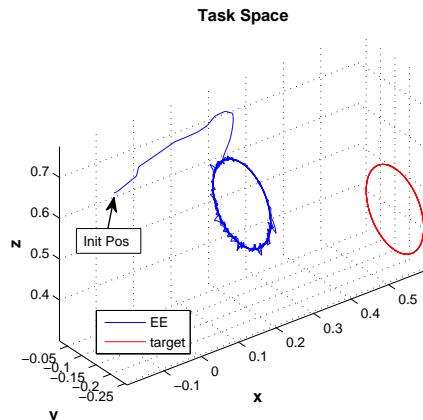
(a) DGN-PBM



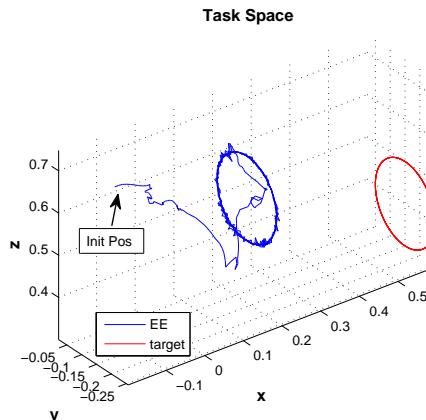
(b) DBFGS-DB



(c) DFN-BFGS-DB

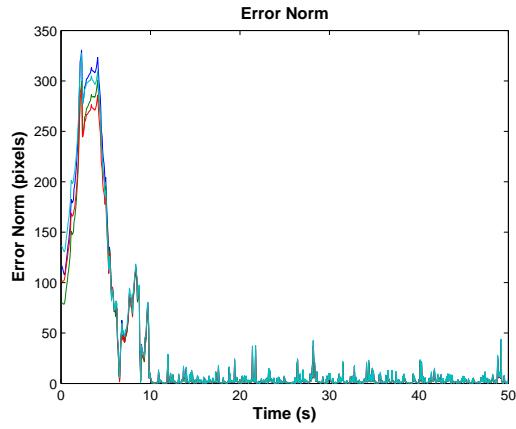


(d) MBFGS-DB

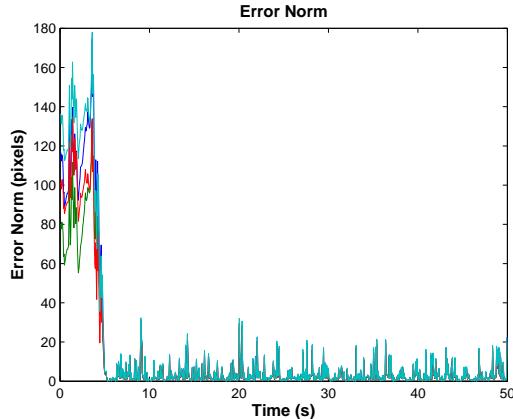


(e) Fu-DB

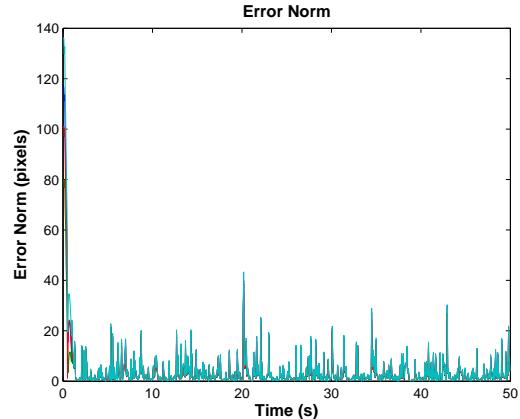
Figure 6.18: The task space view showing camera and one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.90 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.



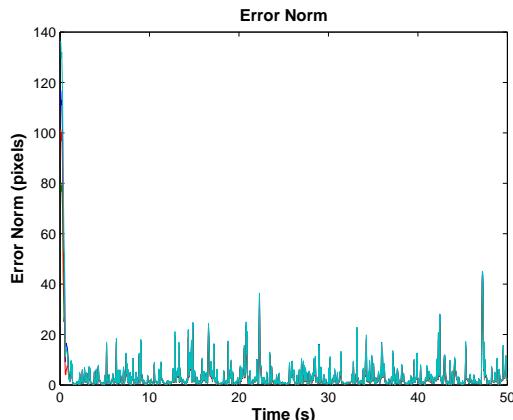
(a) DGN-PBM



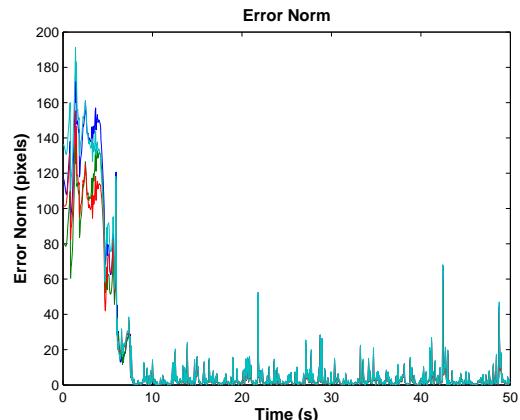
(b) DBFGS-DB



(c) DFN-BFGS-DB



(d) MBFGS-DB



(e) Fu-DB

Figure 6.19: The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a circular target trajectory moving at $\omega = 0.90$ rad/s. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.

Cycloidal Trajectory

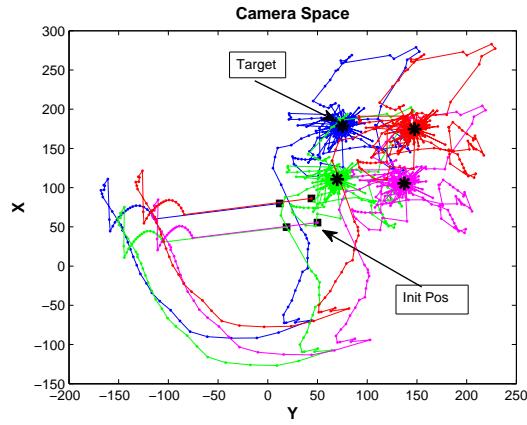
To further investigate the performance of the switching algorithms a more complex cycloid trajectory is generated in the Z-Y plane of Cartesian space,

$$\begin{aligned} x &= 600 & \text{mm} \\ y &= -150 + 100 \sin(kT) + 200 \sin(0.25kT + \frac{\pi}{2}) & \text{mm} \\ z &= 400 + 100 \sin(kT + \frac{\pi}{2}) + 200 \sin(0.25kT) & \text{mm} \end{aligned} \quad (6.5)$$

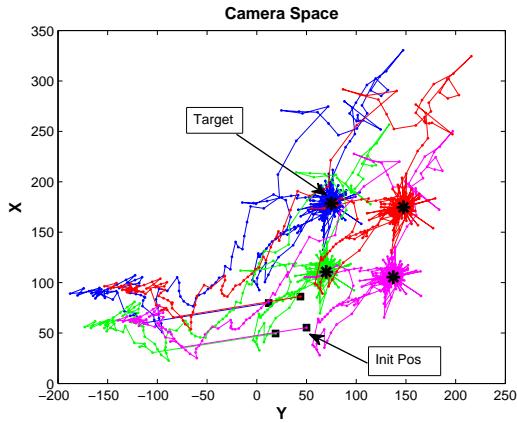
The same PUMA 560 robot with the eye-in-hand camera configuration used for the circular trajectory is also used in these simulations. The sampling period is $T = 50$ ms. The initial robot configuration is also the same as in the circular trajectory with $\theta_0 = [15.73^\circ, 132.5^\circ, -135.6^\circ, -4.27^\circ, -108.75^\circ, 14.27^\circ]^T$. To make the end-effector and target trajectories visually distinct they are offset by a constant vector $[400, -18.09, -141.3]^T$ mm. The forgetting factor $\lambda = 0.5$ is used for all algorithms.

Since the switching method employing the heuristic switching value results in better tracking performance compared to Schemes 1 and 2 in the circular trajectory, only the switching method is implemented with a variety of residual approximation algorithms. However, different values of the switching parameter v are required for each residual approximation approach to achieve convergence. The camera view, the task space view, and the RMS tracking error are presented in Figure 6.20, Figure 6.21, and Figure 6.22 respectively.

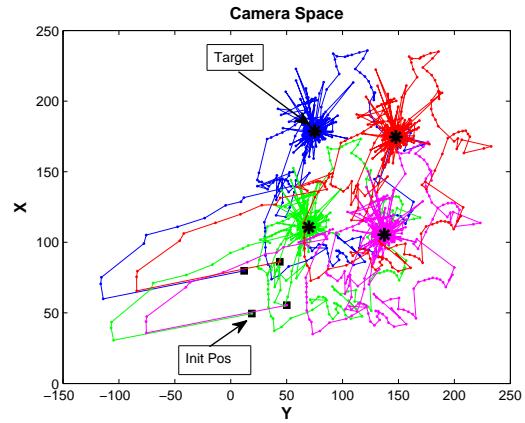
Table 6.13 reveals that the switching MBFGS-DB algorithm gives the fastest convergence with an RMS tracking error similar to the other algorithms. This result is clearly seen in Figure 6.21 where the MBFGS-DB algorithm moves the EE directly to the target plane.



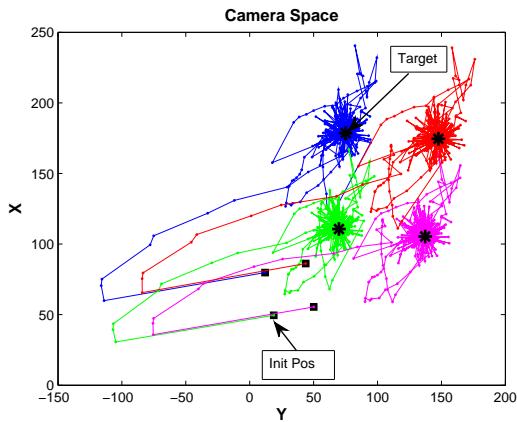
(a) DGN-PBM



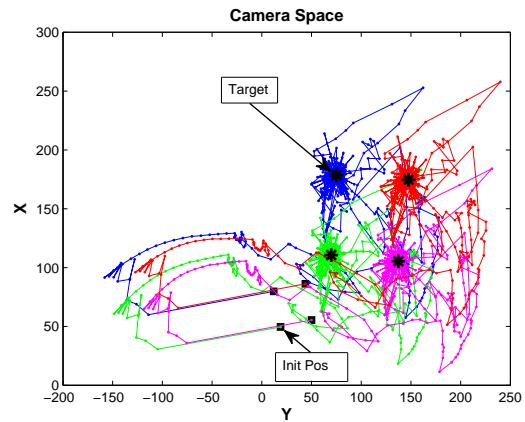
(b) DBFGS-DB



(c) DFN-BFGS-DB

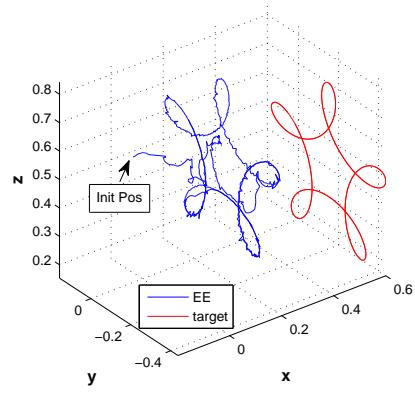


(d) MBFGS-DB

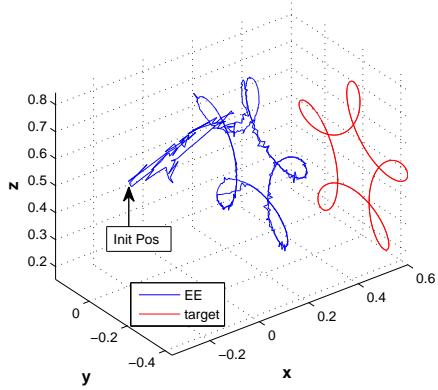


(e) Fu-DB

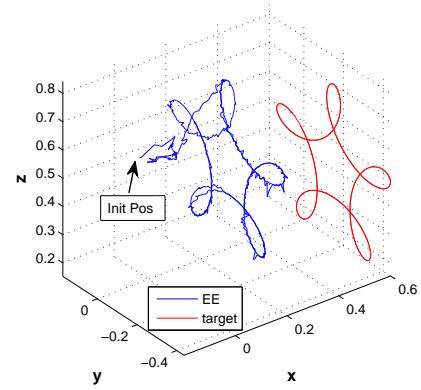
Figure 6.20: The camera space of the PUMA 560 manipulator with an eye-in-hand camera configuration tracks four feature points of a cycloid target trajectory. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.



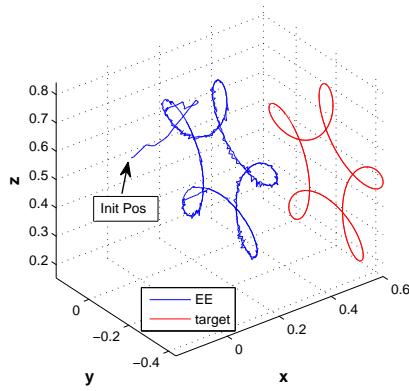
(a) DGN-PBM



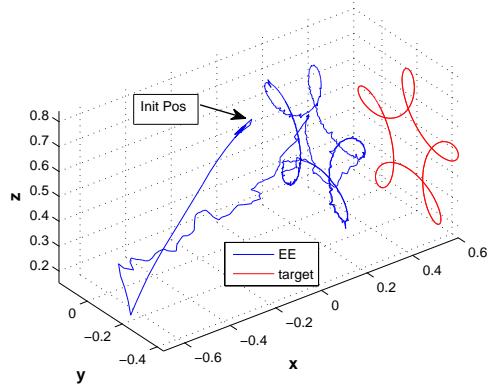
(b) DBFGS-DB



(c) DFN-BFGS-DB

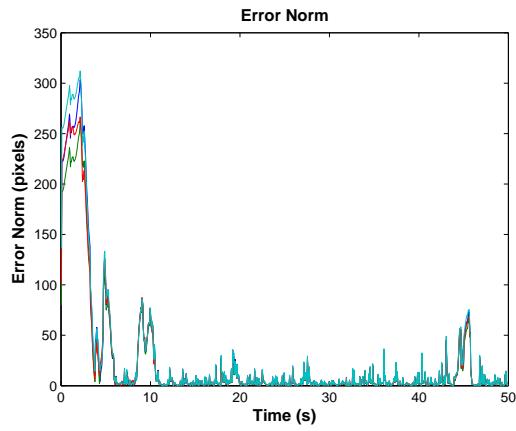


(d) MBFGS-DB

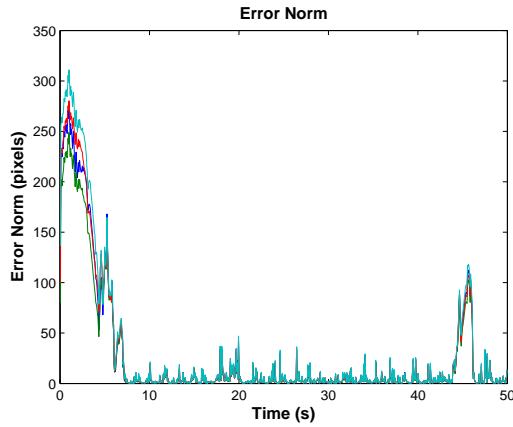


(e) Fu-DB

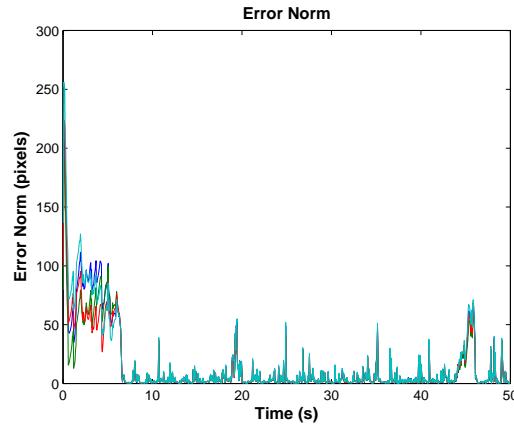
Figure 6.21: The task space view showing camera and one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a cycloid target trajectory. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.



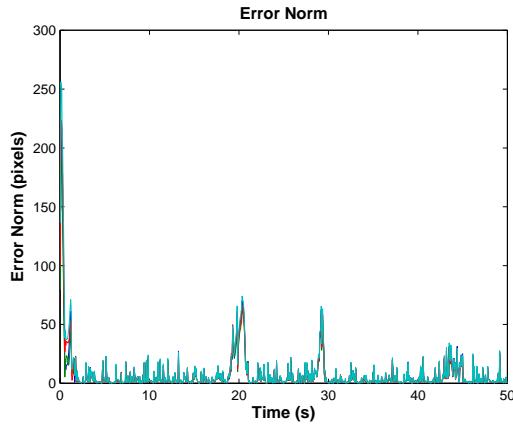
(a) DGN-PBM



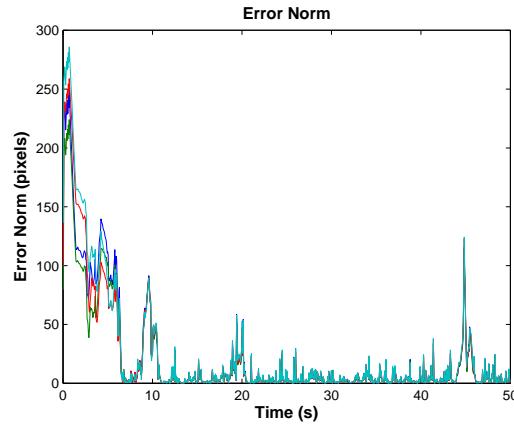
(b) DBFGS-DB



(c) DFN-BFGS-DB



(d) MBFGS-DB



(e) Fu-DB

Figure 6.22: The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration tracks four feature points of a cycloid target trajectory. The forgetting factor is $\lambda = 0.5$ and v is selected to ensure convergence.

Table 6.13: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the cycloidal trajectory using a fixed forgetting factor $\lambda = 0.5$ and v selected to ensure convergence.

Scheme	RMS Error (pixels)	t_s (sec)
DGN-PBM	11.4158	11
DBFGS-DB $v = 1$	17.4930	7.5
DFN-BFGS-DB $v = 0.8$	11.1637	7
MBFGS-DB $v = 0.5$	11.1803	2
Fu-DB $v = 1.5$	11.6818	11

A faster speed of cycloidal trajectory is also tested,

$$\begin{aligned}
 x &= 725 & \text{mm} \\
 y &= -260 + 100 \sin(3kT) + 300 \sin(0.3kT + \frac{\pi}{2}) & \text{mm} \\
 z &= 220 + 100 \sin(3kT + \frac{\pi}{2}) + 300 \sin(0.3kT) & \text{mm}
 \end{aligned} \tag{6.6}$$

Only the switching DFN-BFGS-DB and MBFGS-DB are tested due to their similar performances that are better than the other algorithms. Due to the complexity and speed of this trajectory, a smaller sampling period $T = 25$ ms is used while for the alternating forgetting factor $\lambda = 0.5$ is utilized during the transient portion and $\lambda = 0.95$ is applied during steady-state tracking. These results are compared with the DGN-PBM algorithm in Table 6.14. The task space view and average error norm are shown in Figure 6.23.

The switching MBFGS-DB algorithm yields the smallest RMS error with less large-error spikes as compared to the switching DFN-BFGS-DB and the DGN-PBM

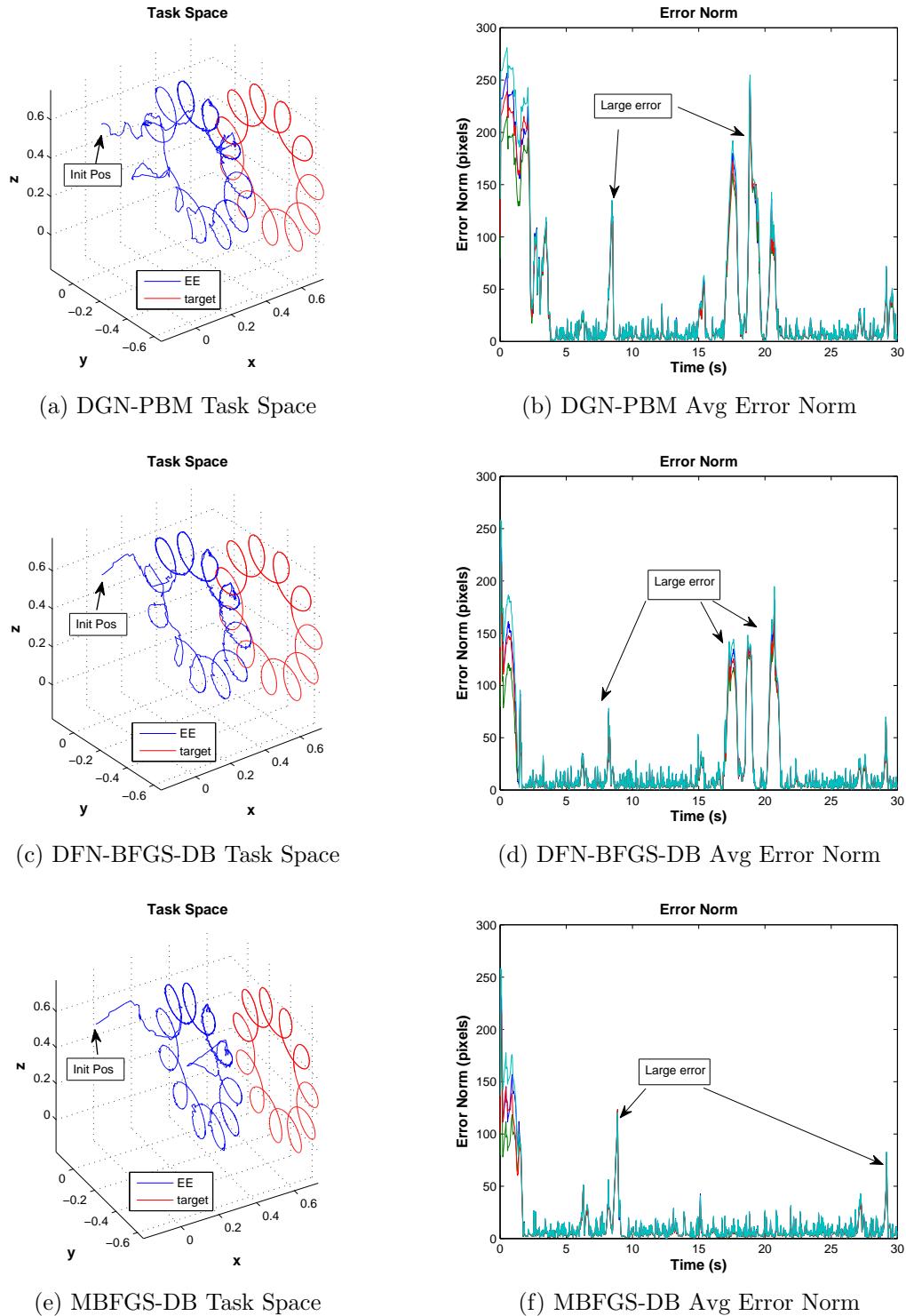


Figure 6.23: The camera space (left column) and the task space (right column) views of the PUMA 560 robot tracking four feature points of a fast cycloidal trajectory.

Table 6.14: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the cycloidal trajectory using a fixed forgetting factor $\lambda = 0.5$ and v selected to ensure convergence.

Scheme	RMS Error (pixels)	t_s (sec)
DGN-PBM	48.9476	4
DFN-BFGS-DB $v = 1.3$	35.4726	2
MBFGS-DB $v = 1.3$	14.3226	3

algorithms as seen in Figure 6.23.

Helical Trajectory

To verify applicability of the switching MBFGS-DB algorithm for out-of-plane tracking a helical path is used with the same exact setup as described for the cycloidal trajectory, the translational helix trajectory is

$$\begin{aligned} x &= 600 + v_x k T & \text{mm} \\ y &= -150 + R \sin(\omega k T) & \text{mm} \\ z &= 400 + R \cos(\omega k T) & \text{mm} \end{aligned} \tag{6.7}$$

where $R = 100$ mm, $v_x = 5$ mm/s is the speed in the x direction, $\omega = 0.45$ rad/s is the angular velocity, k is the iteration number, and $T = 50$ ms is the sampling period. Although v_x seems to be small, the helical path can be generated within the robot working space for a longer period. A faster v_x is tested later. The target motion starts from the top of the helix and moves counterclockwise in the positive x direction.

Tracking performance of the helical trajectory using the switching DFN-BFGS-DB, MBFGS-DB, and Fu-DB are similar to one another and significantly better

Table 6.15: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the helical trajectory using a fixed forgetting factor $\lambda = 0.5$, $v_x = 5$ mm/s, and v selected to ensure convergence.

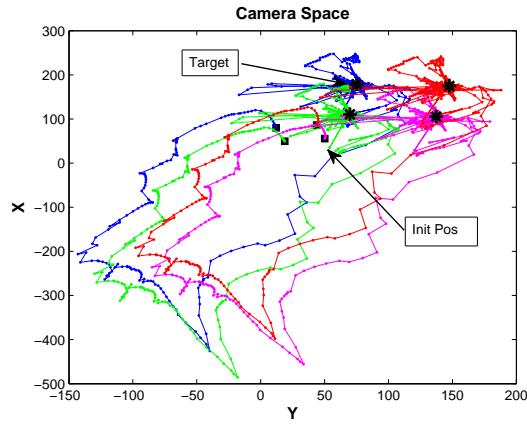
Scheme	RMS Error (pixels)	t_s (sec)
DGN-PBM	12.2552	12
DBFGS-DB $v = 0.3$	1.6108	3.2
DFN-BFGS-DB $v = 0.3$	1.8557	1.5
MBFGS-DB $v = 0.3$	2.1322	1.5
Fu-DB $v = 0.3$	1.9411	1.5

than the DGN-PBM algorithm. In fact the helical trajectory tracking performance is similar to the circular (in a plane) case.

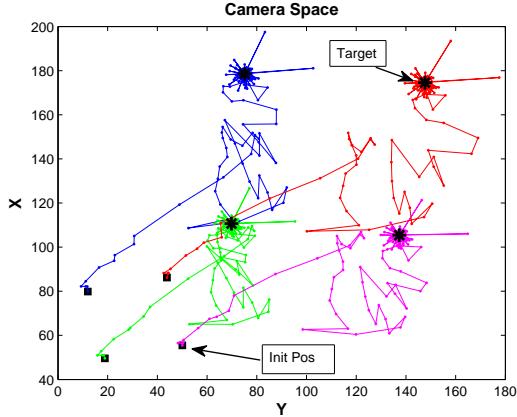
Although the switching MBFGS-DB algorithm does not distinctively outperform the switching DFN-BFGS-DB or the Fu-DB algorithms, it yields a smaller RMS tracking error for the faster speed $v_x = 10$ mm/s as shown in Table 6.16. The resultant task space views of the switching DFN-BFGS-DB, MBFGS-DB, and Fu-DB algorithm with $v_x = 10$ mm/s are shown in Figure 6.27.

6.3.5 Conclusion

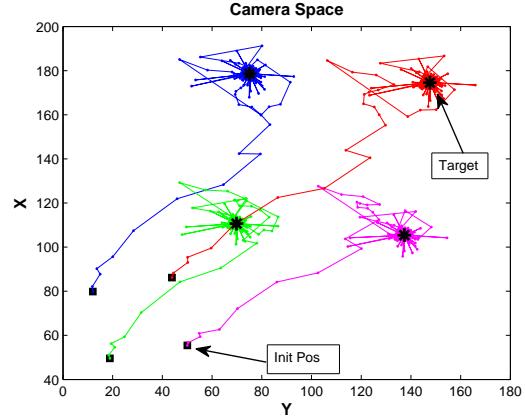
A variety of residual \hat{S}_k approximations improve tracking performance for large-residual problems. The switching DFN-BFGS-DB and the MBFGS-DB algorithms converge for a variety of trajectories using two distinct robots and different camera configurations. However, for the faster and the more complex trajectories the switching MBFGS-DB algorithm provides the best stability with RMS tracking errors and



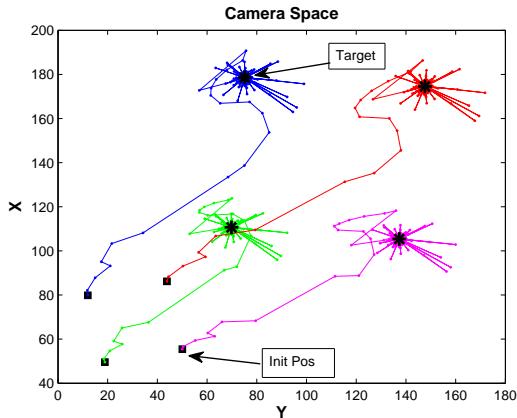
(a) DGN-PBM



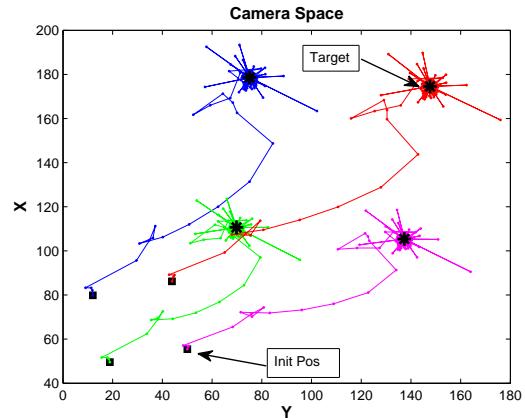
(b) DBFGS-DB



(c) DFN-BFGS-DB

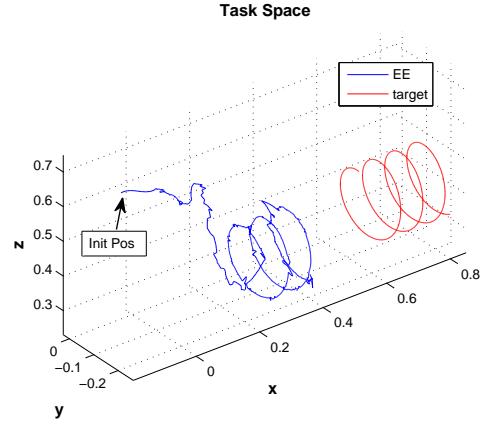


(d) MBFGS-DB

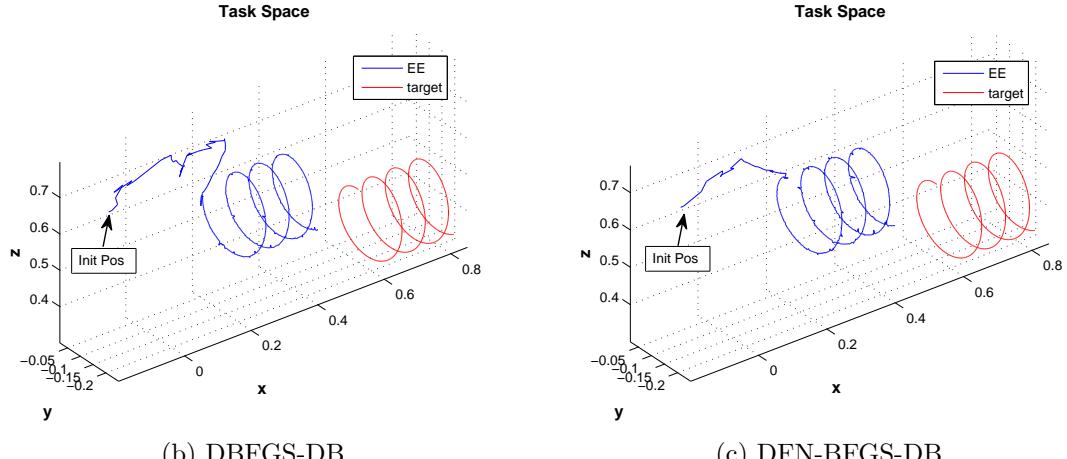


(e) Fu-DB

Figure 6.24: The camera space of the PUMA 560 manipulator with an eye-in-hand camera configuration tracks four feature points of a helical target trajectory. The forgetting factor is $\lambda = 0.5$, $v_x = 5$ mm/s, and v is selected to ensure convergence.

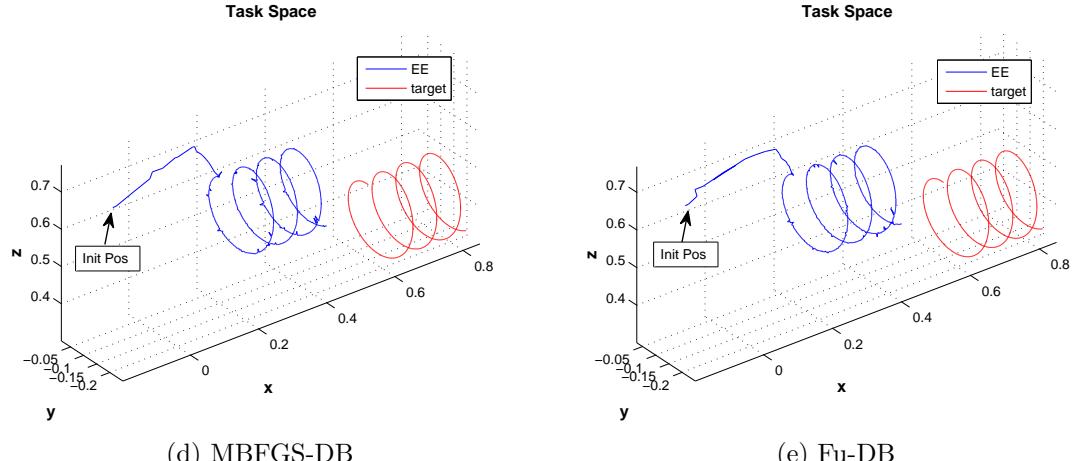


(a) DGN-PBM



(b) DBFGS-DB

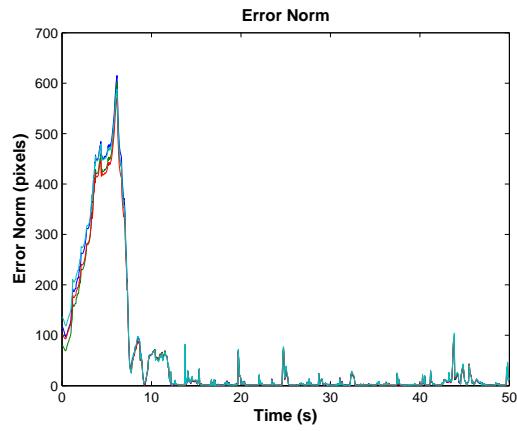
(c) DFN-BFGS-DB



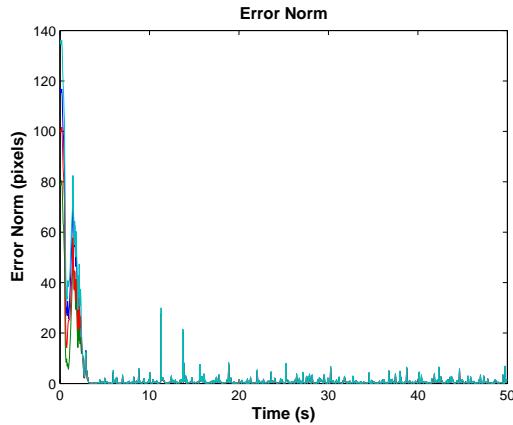
(d) MBFGS-DB

(e) Fu-DB

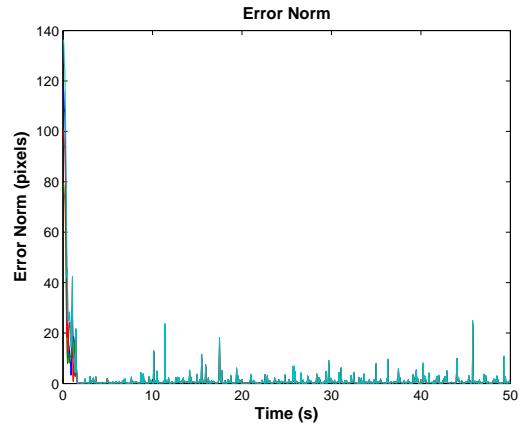
Figure 6.25: The task space view showing camera and one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a helical target trajectory. The forgetting factor is $\lambda = 0.5$, $v_x = 5$ mm/s, and v is selected to ensure convergence.



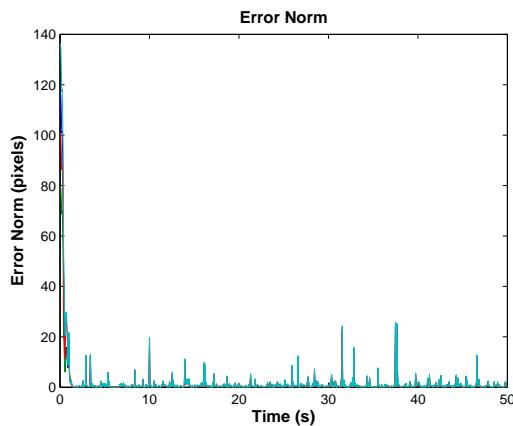
(a) DGN-PBM



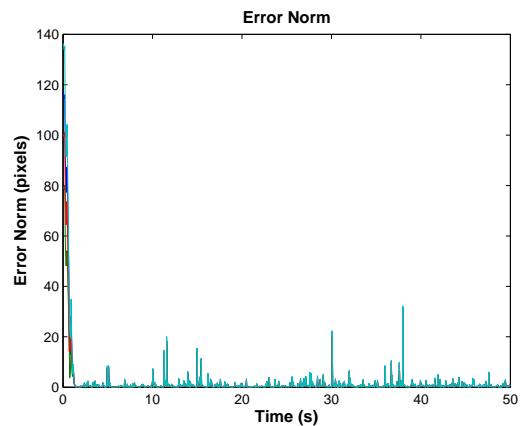
(b) DBFGS-DB



(c) DFN-BFGS-DB



(d) MBFGS-DB



(e) Fu-DB

Figure 6.26: The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration tracks four feature points of a helical target trajectory. The forgetting factor is $\lambda = 0.5$, $v_x = 5$ mm/s, and v is selected to ensure convergence.

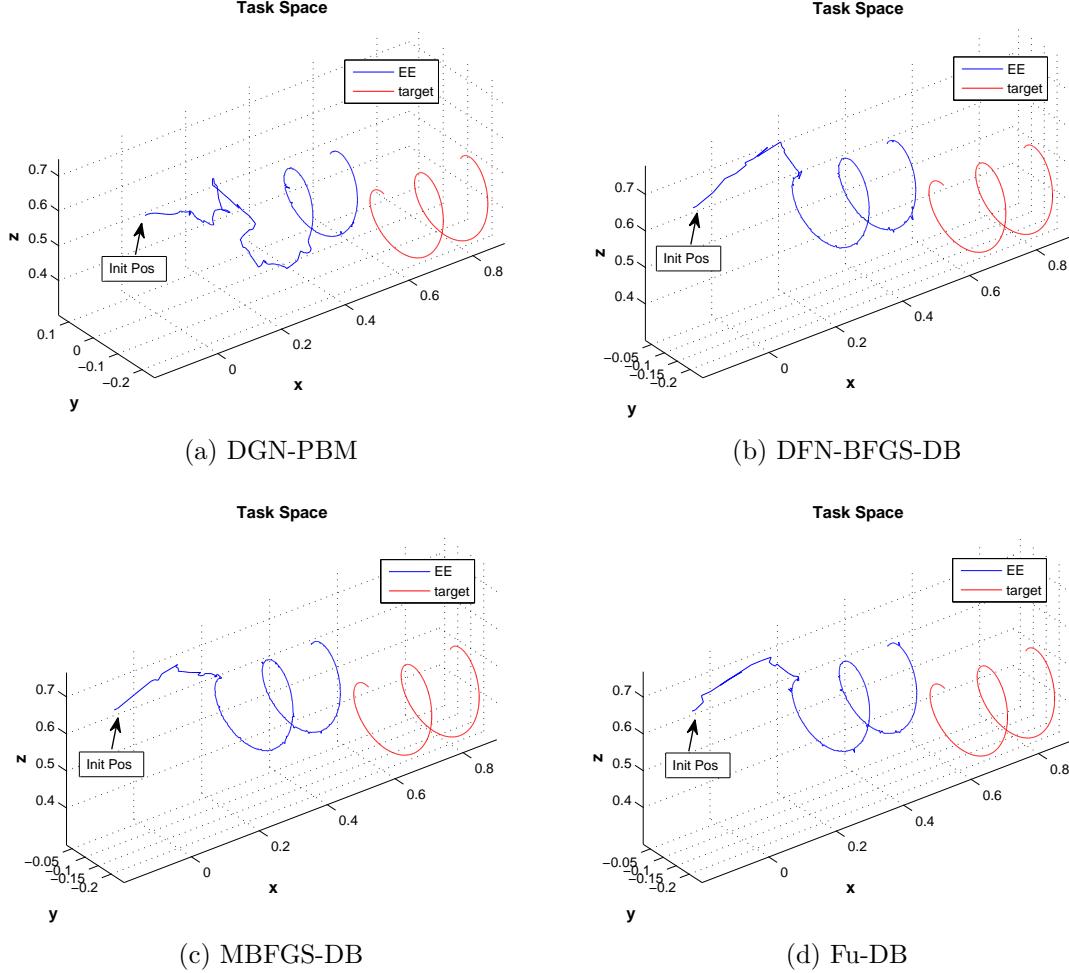


Figure 6.27: The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration tracking four feature points of a helical target trajectory moving with a faster speed in x direction. The forgetting factor is $\lambda = 0.5$, $v_x = 10$ mm/s, and v is selected to ensure convergence.

Table 6.16: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes for tracking the helical trajectory using a fixed forgetting factor $\lambda = 0.5$, $v_x = 10$ mm/s, and v selected to ensure convergence.

Scheme	RMS Error (pixels)	t_s (sec)
DGN-PBM	1.9217	10
DFN-BFGS-DB $v = 0.3$	1.8214	1.5
MBFGS-DB $v = 0.3$	1.4278	1.5
Fu-DB $v = 0.5$	1.7266	1.5

convergence times that are either the best or reasonably close to the best. These performance depends on the selection of the switching parameter v , which is heuristically chosen in this study. The switching algorithms offer better tracking stability with smaller RMS error and settling time as compared to Schemes 1 and 2.

6.4 Performance Evaluation of the switching MBFGS-DB Algorithm with LMA

The switching MBFGS-DB algorithm requires an invertible Hessian matrix for finding the robot joint angles θ_k at each iteration. An ill-conditioned or singular Hessian matrix \hat{H}_k can lead to numerical problems resulting in slow or no convergence.

In robotic tracking applications a singular Hessian matrix \hat{H}_k can occur at a kinematic singularity. The robot angles θ_k is a solution of

$$\hat{H}_k h_k = -\hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right)$$

where $h_k = \theta_{k+1} - \theta_k$. If \hat{H}_k is singular, there may exist an infinite number of solutions h_k and thus an infinite number of robot configurations giving the same EE location.

This phenomenon occurs if the robot is at a kinematic singularity configuration where the robot loses one or more degrees of freedom. This problem may lead to solutions with unbounded joint velocities.

To improve upon Hessian ill-conditioning various switching algorithms are implemented with the well-known LevenbergMarquardt algorithm (LMA). The LMA offers an alternate method for solving nonlinear optimization problems when the Hessian H_k or \hat{H}_k is not positive definite or becomes ill-conditioning. This method modifies the Hessian matrix to ensure positive definiteness to overcome this deficiency. Though this introduces nonphysical artifacts, it can improve effectiveness near singularities.

The idea is to utilize a trust-region strategy for unconstrained optimization where the Hessian is modified (see Section 2.3),

$$H_{d,k} = H_k + \mu_k D_k$$

where $H_{d,k}$ is known as the *modified Hessian matrix*, μ_k is called the *damping* or the *Levenberg-Marquardt parameter*, and D_k is a diagonal matrix.

In this study the Hessian is approximated using the quasi-dynamic Broyden's method for Jacobian \hat{J}_k and the MBFGS method for residual \hat{S}_k . The modified Hessian matrix $\acute{H}_{d,k}$ becomes

$$\acute{H}_{d,k} = \hat{J}_k^T \hat{J}_k + \varphi_k \hat{S}_k + \mu_k D_k \quad (6.8)$$

and the quadratic model q_k of the objective function $F(\theta, t)$ becomes

$$q_k = \frac{1}{2} f_k^T f_k + (J_k^T f_k)^T (\theta - \theta_k) + \frac{1}{2} (\theta - \theta_k)^T \acute{H}_{d,k} (\theta - \theta_k) \quad (6.9)$$

The *modified Newton's method* in Section 2.3 is used to solve (6.9) as

$$\theta_{k+1} = \theta_k - \left(\acute{H}_{d,k} \right)^{-1} \hat{J}_k^T \left(f_k + \frac{\partial f_k(t)}{\partial t} h_t \right) \quad (6.10)$$

where $h_k = \theta_{k+1} - \theta_k$ satisfies

$$\|h_k\| \leq \delta_k \quad (6.11)$$

and δ_k determines the size of the trust region.

The LMA updates μ_k and D_k for solutions q_k satisfying (6.11). There are various strategies to calculate μ_k , D_k , and δ_k [19, 50]. Inspired by the implementation of the LMA for large residual visual servoing in [25], which updates μ_k and D_k using Moré's approach [47], the switching MBFGS-DB algorithm also follows this approach.

To adjust the trust region δ_k size according to how well the model q_k approximates the objective function F_k , [25] uses the strategy presented in [23]. The size depends on the ratio r between the predicted and the actual reduction of image error. The actual reduction of the objective function is

$$\Delta F = F_k - F_{k-1} \quad (6.12)$$

while, the predicted reduction is

$$\Delta q = F_k - q_{k-1} \quad (6.13)$$

and

$$r = \frac{\Delta F}{\Delta q} \quad (6.14)$$

The closer r is to unity, the better the approximation of F_k using the current model q_k . The trust region size δ_k needs to be adjusted as

$$\delta_{k+1} = \begin{cases} 0.5\|D_k h_k\|_2 & \text{if } r \leq 0.25 \\ 2\|D_k h_k\|_2 & \text{if } r \geq 0.75 \\ \|D_k h_k\|_2 & \text{if otherwise} \end{cases}$$

To isolate the effect of the LMA the fixed forgetting factor $\lambda_k = 0.5$ is used in this section for all tests.

6.4.1 RRR Robot

The same RRR robot with two eye-to-hand cameras presented in Section 6.3.1 is used to track one feature point moving in the circular trajectory (6.1) with the starting

configuration at $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$. Tracking performance is compared for various residual approximations with/without the LMA and \hat{J}_k is approximated using partitioned (P-) and non-partitioned (NP-) Broyden's estimators. The RMS tracking error and the settling time t_s of each switching algorithm with/without the LMA implementation are summarized in Table 6.17 for P-Broyden's estimator and in Table 6.18 for NP-Broyden's estimator.

Table 6.17: The RMS error and the settling time comparison of the RRR robot for various residual \hat{S}_k approximation schemes with/without the LMA for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\lambda = 0.5$, and $v = 0.3$. The partitioned Broyden's estimator is used for Jacobian \hat{J}_k approximation.

Scheme	LMA	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
DBM-RLS	N/A	0.0171	0.0183	1.6
DGN-PBM	N/A	0.1254	0.1258	2.3
DBFGS-DB	No	0.0948	0.0954	3.7
	Yes	0.1254	0.1251	1.6
DFN-BFGS-DB	No	0.0870	0.0870	5.6
	Yes	0.1708	0.1703	3
MBFGS-DB	No	0.1059	0.1054	1.3
	Yes	0.3023	0.3030	2
Fu-DB	No	0.0905	0.0903	5.5
	Yes	0.1540	0.1540	2.5

All switching algorithms (except MBFGS-DB) using the P-Jacobian estimation with the LMA implementation show improvement in faster convergence, and all have slightly higher RMS tracking errors. In contrast, the LMA with NP-Jacobian estimation generates worse results for both settling time and RMS error compared to not including the LMA.

Table 6.18: The RMS error and the settling time comparison of the RRR robot for various residual \hat{S}_k approximation schemes with/without the LMA for $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$, $\lambda = 0.5$, and $v = 0.3$. The NP Broyden's estimator is used for Jacobian \hat{J}_k approximation.

Scheme	LMA	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
DBFGS-DB	No	0.0111	0.0111	1.2
	Yes	0.2449	0.2443	1.6
DFN-BFGS-DB	No	0.0143	0.0144	1.3
	Yes	0.2442	0.2436	1.5
MBFGS-DB	No	0.0130	0.0131	1.6
	Yes	0.2450	0.2444	2.2
Fu-DB	No	0.0110	0.0110	1.2
	Yes	0.2747	0.2735	4

Various starting robot configurations listed in Table 6.19 are used to further evaluate performance of the switching MBFGS-DB with/without the LMA as shown in Table 6.20. The camera and task space views are shown in Figure 6.28 and Figure 6.29.

Table 6.19: Various starting RRR robot configurations.

Position	Robot Joint Angles
1	$\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$
2	$\theta_0 = [100^\circ, 120^\circ, 50^\circ]^T$
3	$\theta_0 = [100^\circ, 120^\circ, 5^\circ]^T$

Implementing the LMA with the switching MBFGS-DB algorithm shows only small improvement in convergence time for the second starting robot angles in Table 6.19 but it actually degrades the tracking performance for both settling time and the

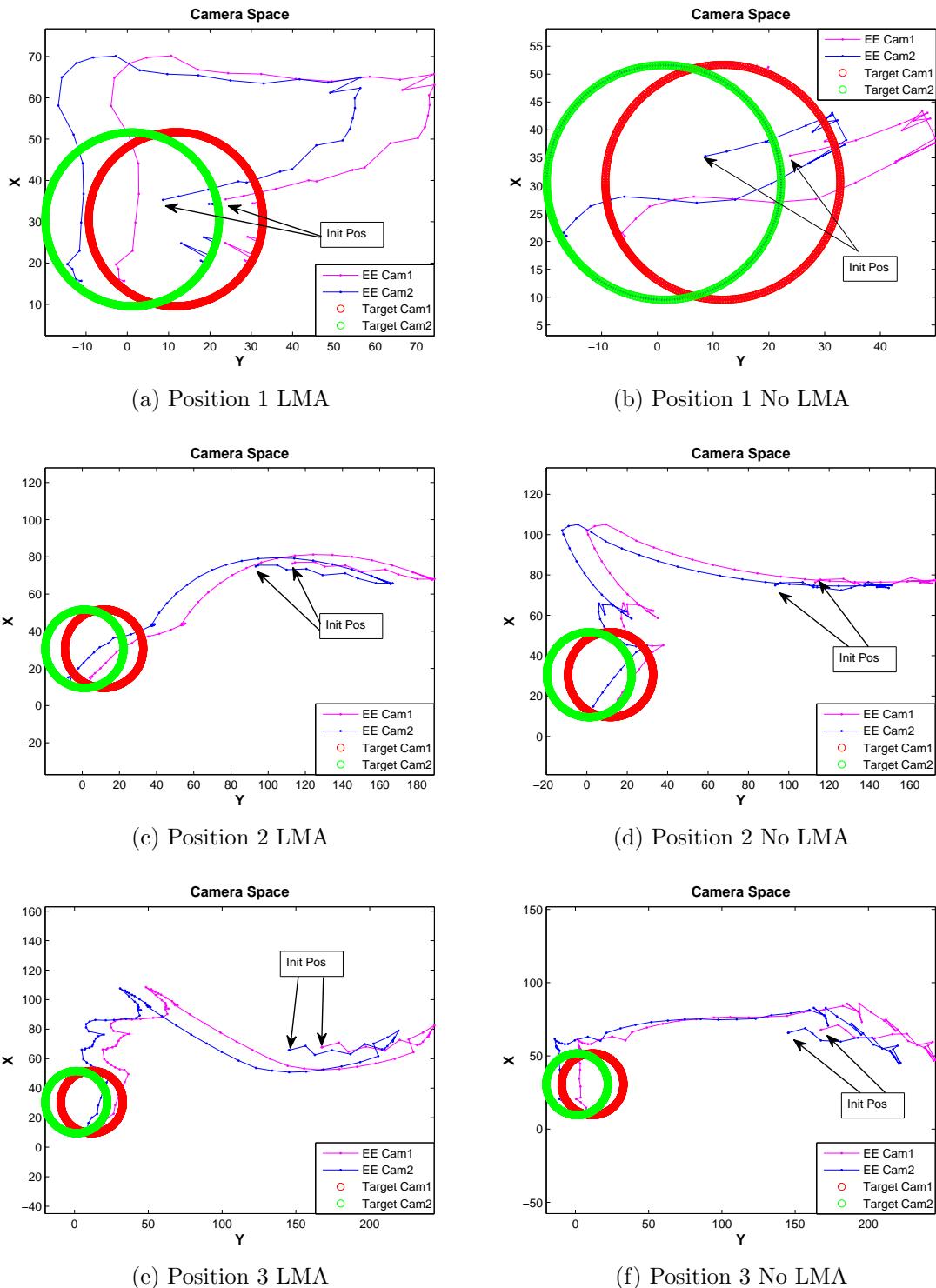


Figure 6.28: The camera space comparison of implementing the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting RRR robot configurations.

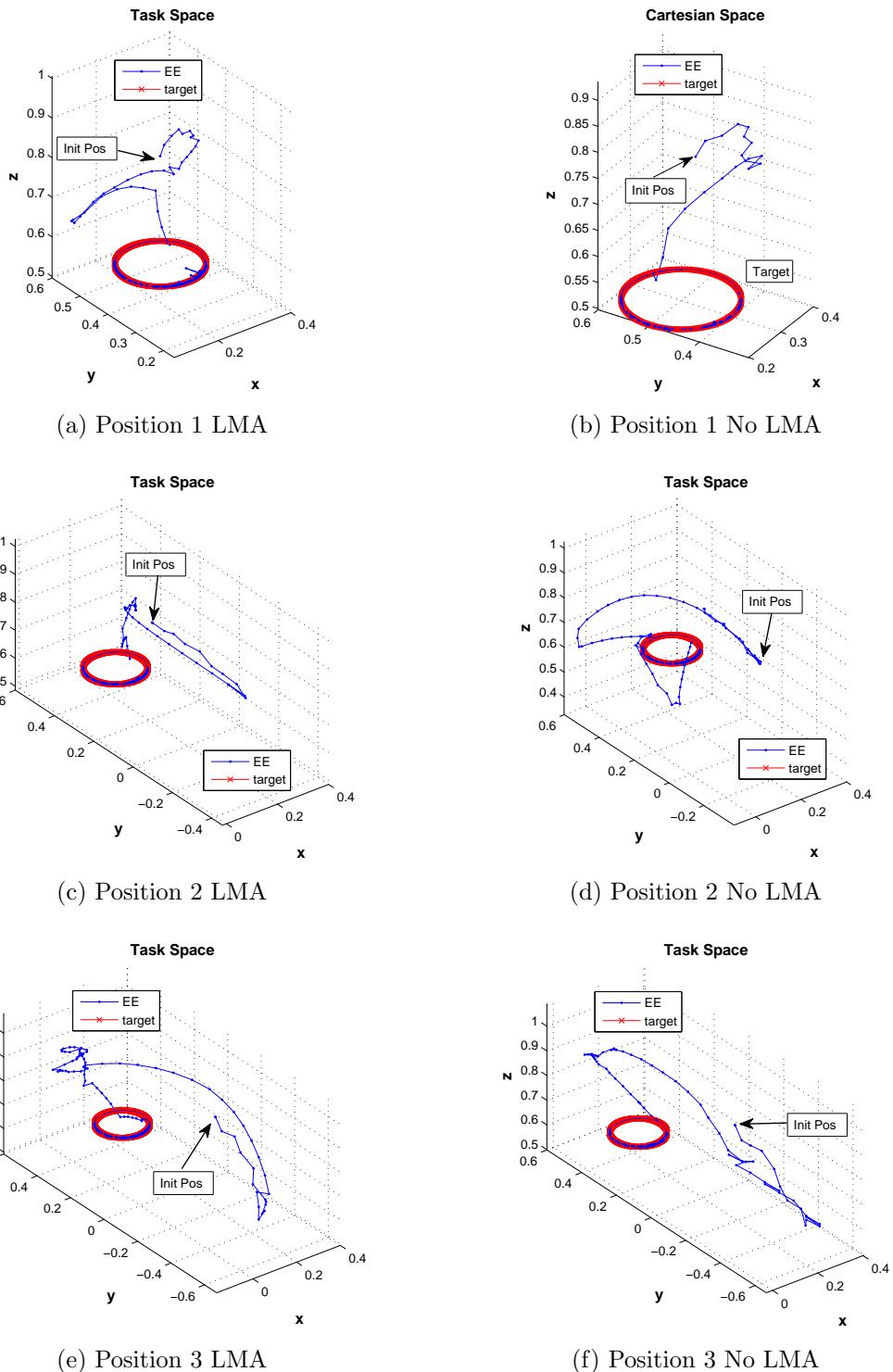


Figure 6.29: The camera space comparison of implementing the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting RRR robot configurations.

Table 6.20: The RMS error and the settling time comparison of the switching MBFGS-DB algorithm with/without the LMA using $v = 0.3$ and $\lambda = 0.5$ at various starting RRR robot configurations.

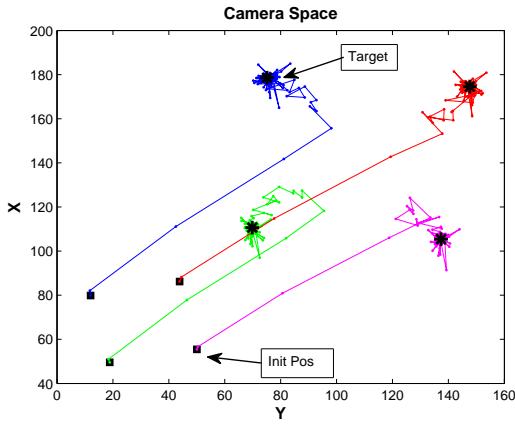
Position	LMA	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
1	No	0.1059	0.1054	1.3
	Yes	0.3023	0.3030	2
2	No	0.1151	0.1147	3.3
	Yes	0.1088	0.1087	2.5
3	No	0.1323	0.1319	3.4
	Yes	0.1377	0.1374	4.4

RMS tracking error for the other cases. Overall, the results show little advantage to including the LMA into the switching algorithm for the large residual tracking using the simple RRR robot. The PUMA 560 robot with an eye-in-hand camera configuration is used to further investigate the LMA performance.

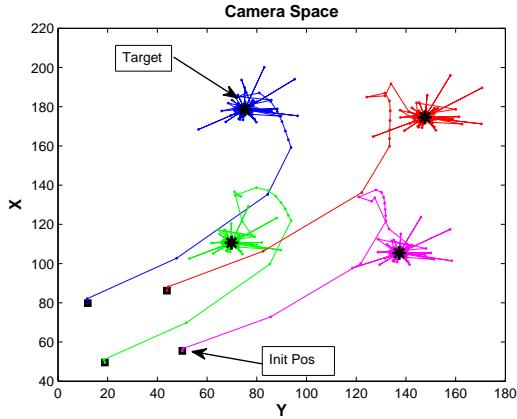
6.4.2 PUMA 560 robot

The PUMA 560 robot with the eye-in-hand camera configuration described in Section 6.3.4 is used to track the circular trajectory in (6.4) switching algorithms with the LMA. The settling time t_s and the RMS tracking error with/without LMA are summarized in Table 6.21. The camera view, task space view, and the average RMS tracking error are shown in Figure 6.30, Figure 6.31, and Figure 6.32 respectively. All switching algorithms with the LMA employ the switching criterion $v = 0.3$ and the forgetting factor $\lambda = 0.5$. The starting robot configuration is $\theta_0 = [15.73^\circ, 132.5^\circ, -135.6^\circ, -4.27^\circ, -108.75^\circ, 14.27^\circ]^T$.

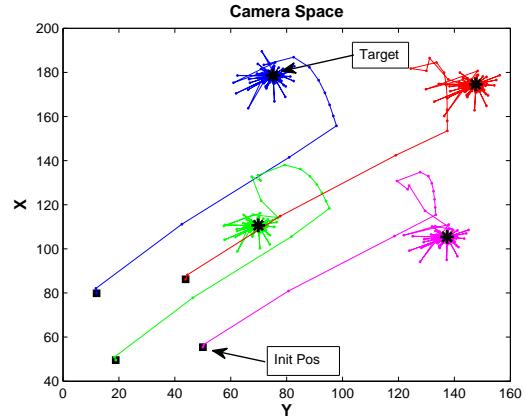
Although the DGN-PBM algorithm with the LMA yields smaller RMS error and t_s compared to the DGN-PBM algorithm with/without the LMA in Table 6.21, the EE motion in the task space of Figure 6.31a does not follow the desired circular trajectory



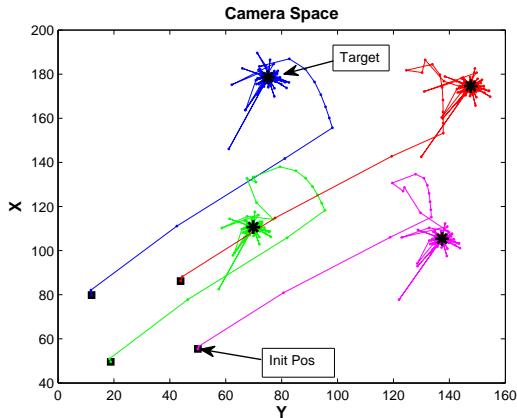
(a) DGN-PBM



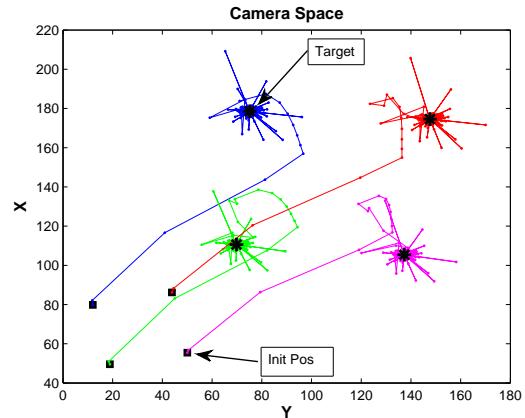
(b) DBFGS-DB



(c) DFN-BFGS-DB

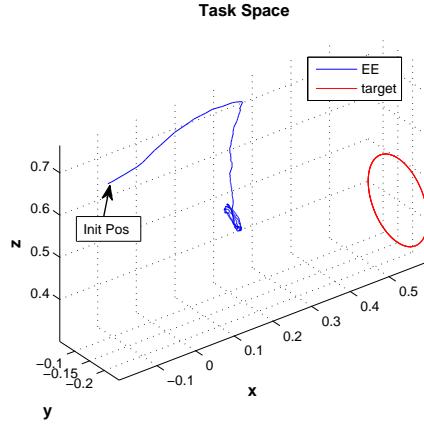


(d) MBFGS-DB

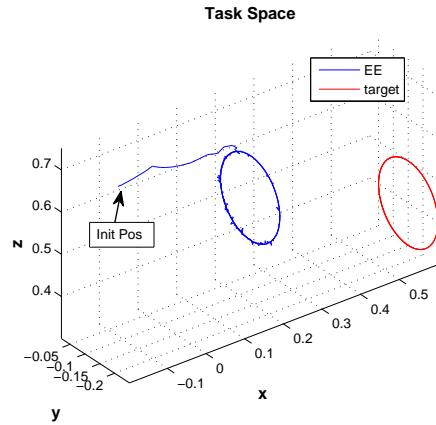


(e) Fu-DB

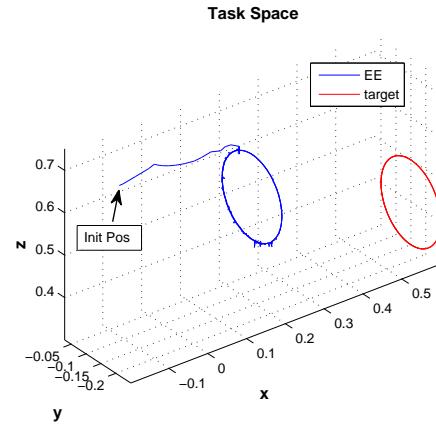
Figure 6.30: The camera space of the PUMA 560 manipulator with the eye-in-hand camera configuration using various switching algorithms implemented with the LMA to track four feature points of the circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$ and $v = 0.3$. The forgetting factor is $\lambda = 0.5$.



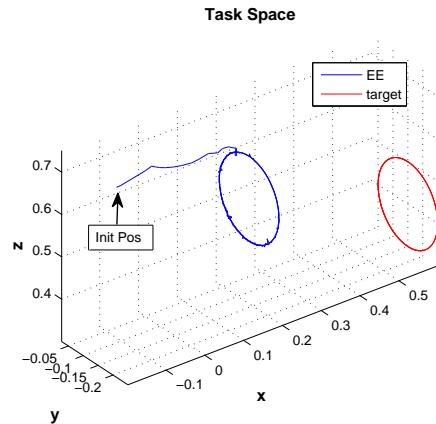
(a) DGN-PBM



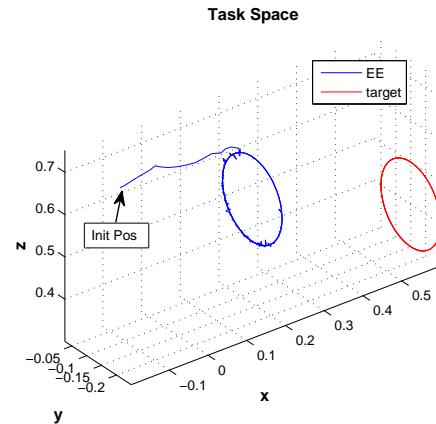
(b) DBFGS-DB



(c) DFN-BFGS-DB

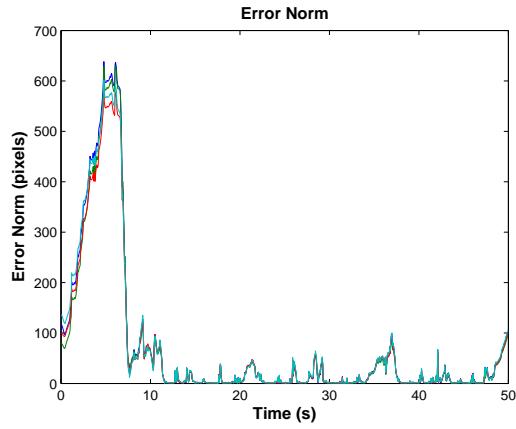


(d) MBFGS-DB

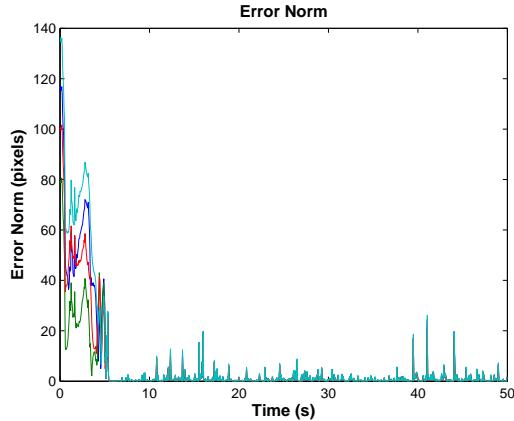


(e) Fu-DB

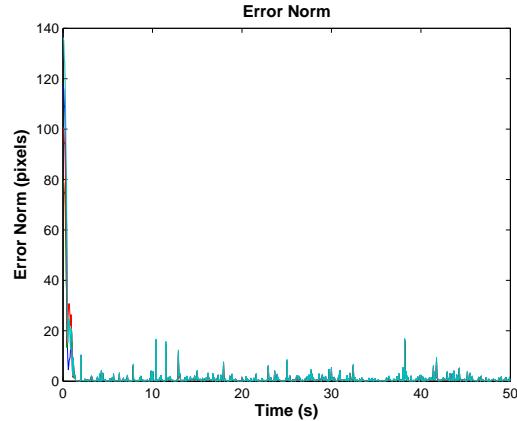
Figure 6.31: The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera configuration using various switching algorithms with the LMA tracking four feature points of a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$.



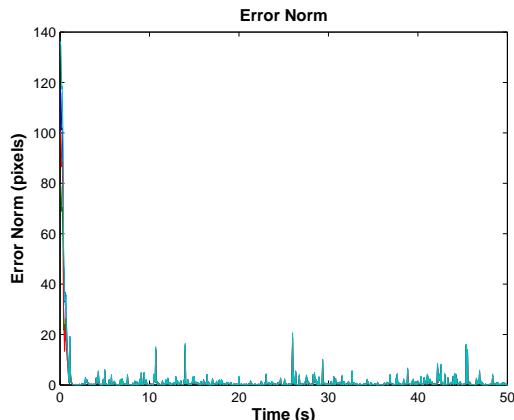
(a) DGN-PBM



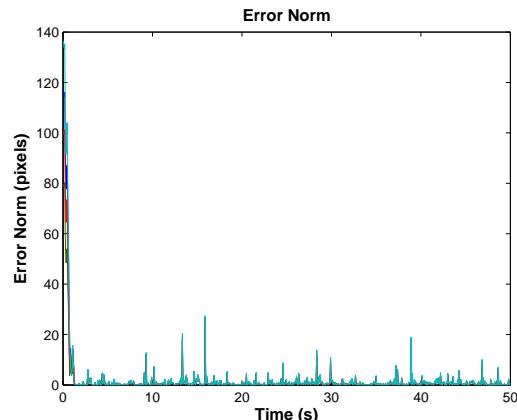
(b) DBFGS-DB



(c) DFN-BFGS-DB



(d) MBFGS-DB



(e) Fu-DB

Figure 6.32: The error norm of the PUMA 560 manipulator with an eye-in-hand camera configuration using various switching algorithms implemented with the LMA to track four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. The forgetting factor is $\lambda = 0.5$ and $v = 0.3$.

Table 6.21: The RMS error and the settling time comparison of various residual \hat{S}_k approximation schemes using the PUMA 560 robot to track a circular trajectory using $\lambda = 0.5$ and $v = 0.3$.

Scheme	LMA	RMS Error (pixels)	t_s (sec)
DGN-PBM	No	22.5085	12
	Yes	1.5421	2
DBFGS-DB	No	1.9956	5.5
	Yes	2.0413	1.2
DFN-BFGS-DB	No	1.4974	1.5
	Yes	1.5100	1.3
MBFGS-DB	No	1.6401	1.5
	Yes	1.7135	1.3
Fu-DB	No	1.7874	1.4
	Yes	2.0106	1.8

at all. For the other switching algorithms the inclusion of the LMA yields faster convergence time while the RMS error is slightly increased, similar to the RRR robot case. Even though the switching MBFGS-DB and the DFN-BFGS-DB algorithms with the LMA provide similar tracking performance, in Section 6.3 the switching MBFGS-DB algorithm is shown as the most effective in handling a different of robot degrees-of-freedom, camera configurations, and the target trajectory. Consequently the switching MBGFS-DB algorithm is used to further evaluate the LMA for different robot starting positions. Table 6.22 summarizes three initial robot configurations used to track the circular trajectory, including the starting point used to generate results in Table 6.21.

For all tests the switching criterion $v = 0.3$ and $\lambda = 0.5$ are utilized. A summary of the switching MBFGS-DB with/without the LMA algorithm with a variety of robot starting positions is given in Table 6.23. The image plane and the task space (of one

Table 6.22: Various starting PUMA 560 robot configurations.

Position	Robot Joint Angles
1	$\theta_0 = [15.73^\circ, 132.5^\circ, -135.6^\circ, -4.27^\circ, -108.75^\circ, 14.27^\circ]^T$
2	$\theta_0 = [25.73^\circ, 132.5^\circ, -135.6^\circ, -4.27^\circ, -108.75^\circ, 44.27^\circ]^T$
3	$\theta_0 = [15.73^\circ, 132.5^\circ, -135.6^\circ, 15.73^\circ, -98.75^\circ, 14.27^\circ]^T$

feature point) views of the MBFGS-DB with the LMA algorithm at various starting positions are shown in Figure 6.33 and Figure 6.34.

Table 6.23: The RMS error and the settling time comparison for the switching MBFGS-DB algorithm with/without the LMA approach using $v = 0.3$ and $\lambda = 0.5$ at various starting PUMA 560 robot configurations tracking a circular trajectory.

Position	LMA	RMS Error Camera 1 (pixels)	t_s (sec)
1	Yes	1.7135	1.3
	No	1.6401	1.5
2	Yes	1.6360	1.6
	No	2.6015	2
3	Yes	1.9311	2.5
	No	1.5879	1.2

The LMA only marginally improves the settling time while the RMS tracking errors are slightly compromised.

For the final LMA evaluation the cycloidal target trajectory in (6.5) is tested using the switching MBFGS-DB with the LMA algorithm at the different starting robot configurations of Table 6.22. The switching criterion $v = 0.3$ and $\lambda = 0.5$ are also utilized for all robot starting positions.

Table 6.24 shows that the RMS error becomes substantial when the LMA is not included for the second starting robot position. This is due to instability occurring

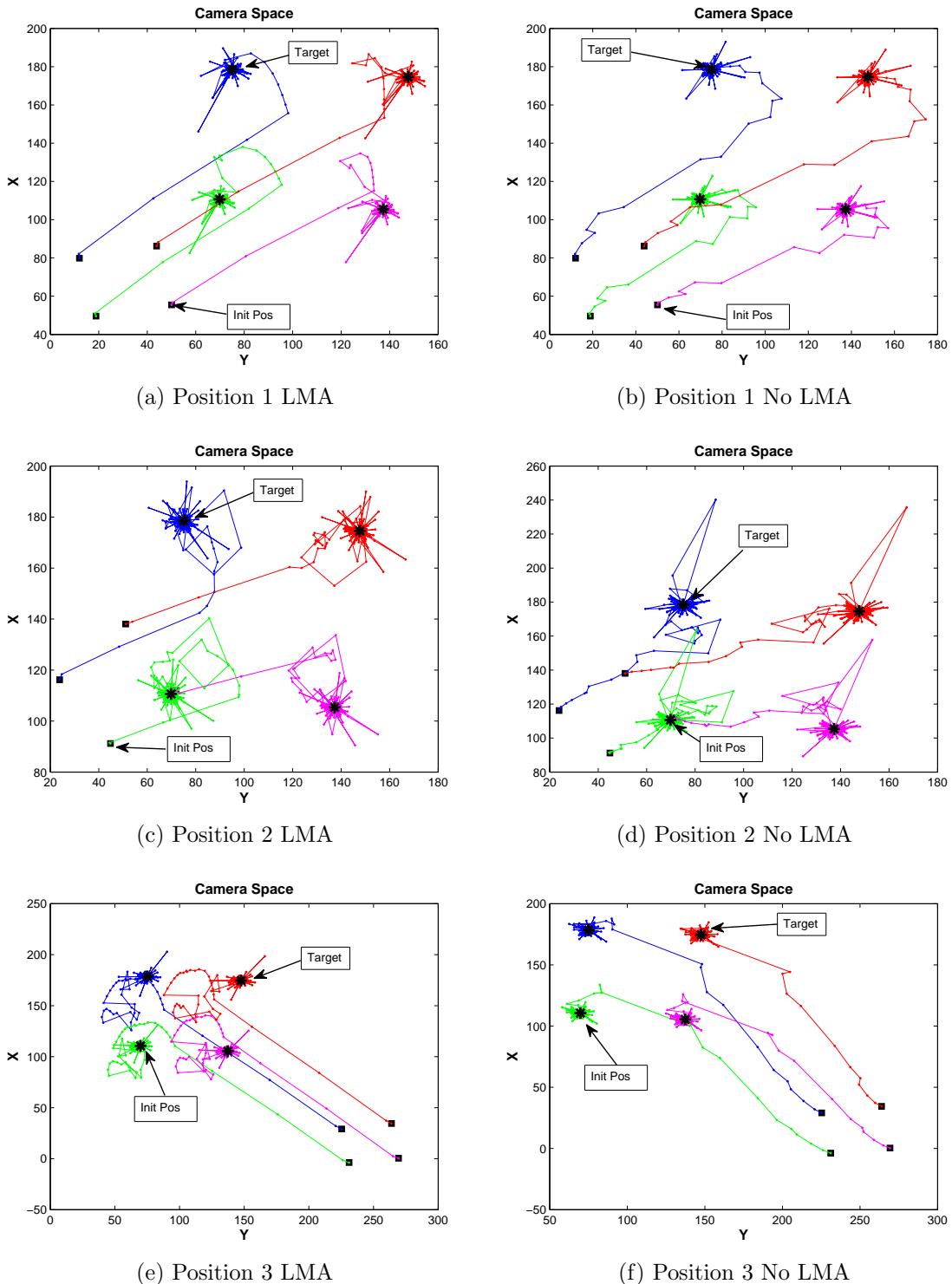


Figure 6.33: The camera space comparison of the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$, $v = 0.3$, and $\lambda = 0.5$.

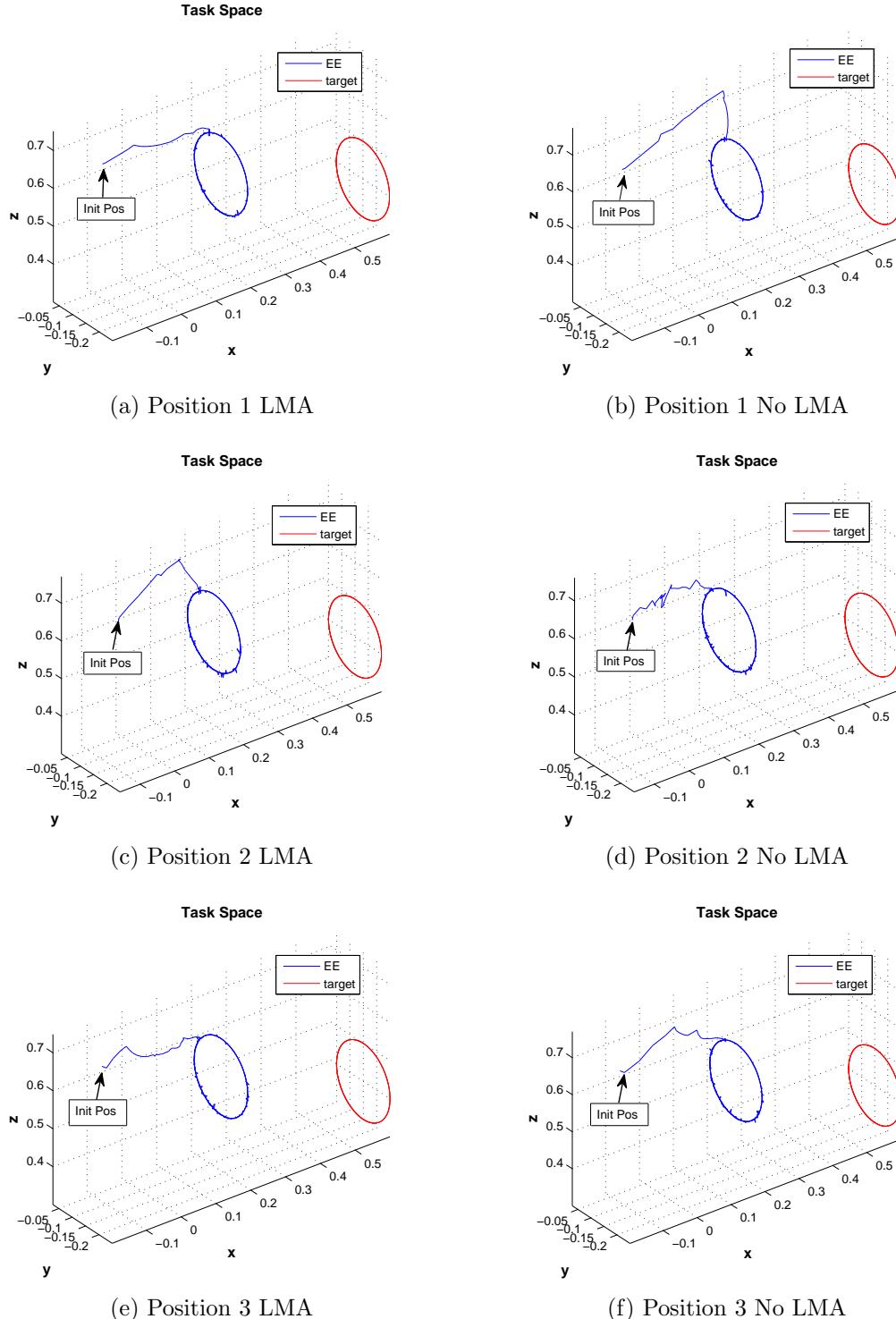


Figure 6.34: The task space view showing one camera and one target point for clarity for the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$, $v = 0.3$, and $\lambda = 0.5$.

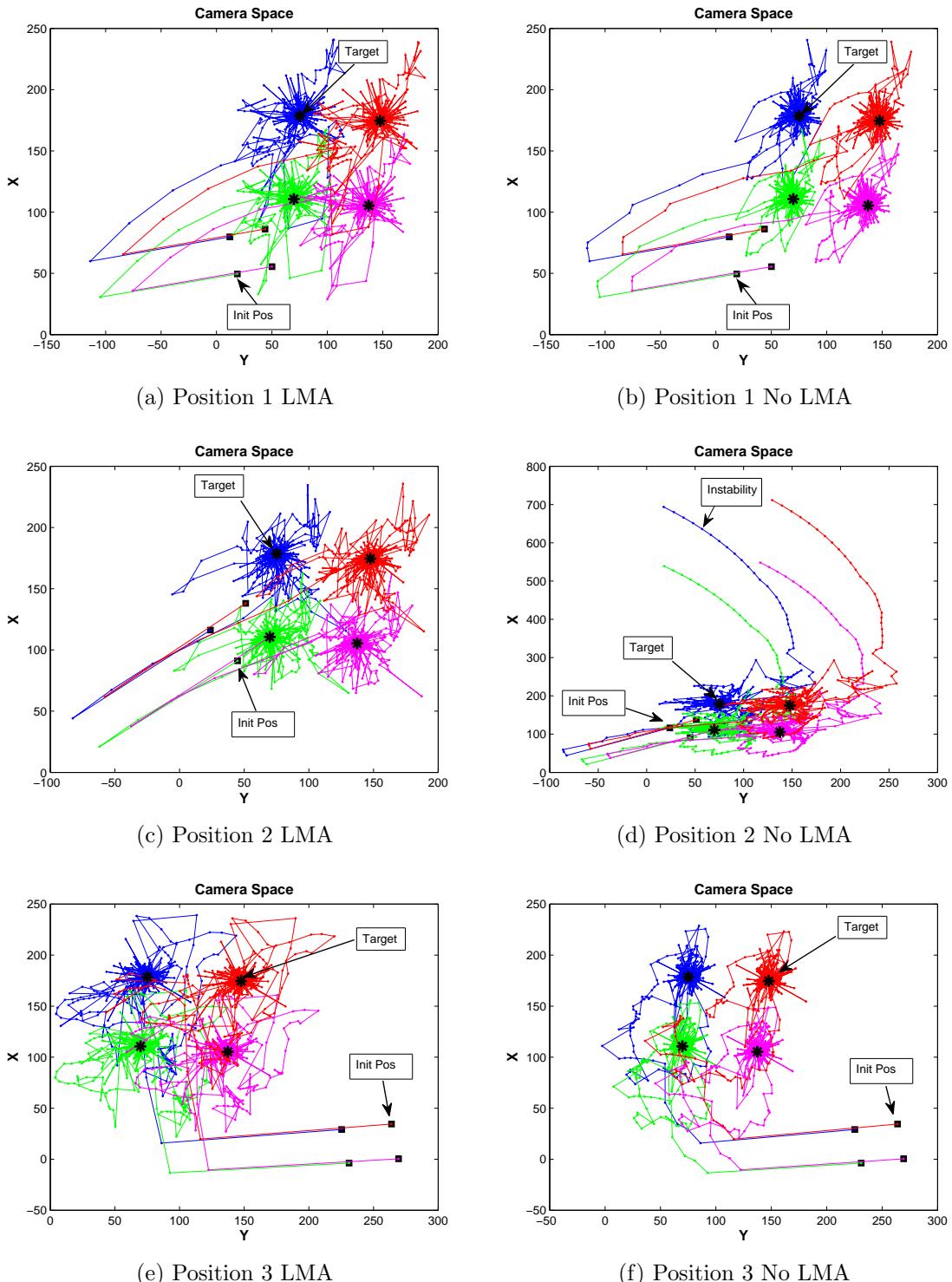


Figure 6.35: The camera space comparison of the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a cycloidal target trajectory using $v = 0.3$ and $\lambda = 0.5$.

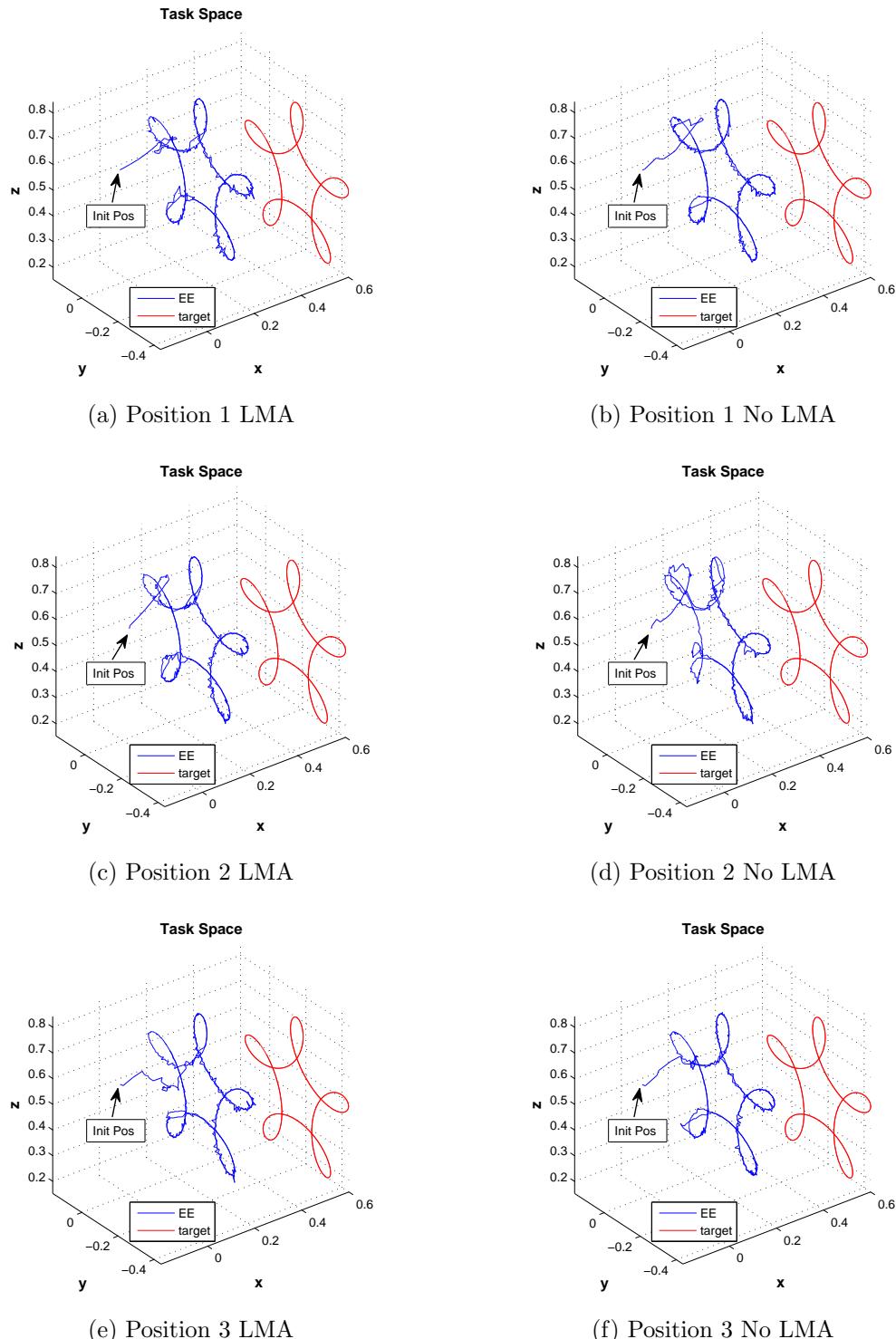


Figure 6.36: The task space of the EE motion using the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a cycloidal target trajectory using $v = 0.3$ and $\lambda = 0.5$.

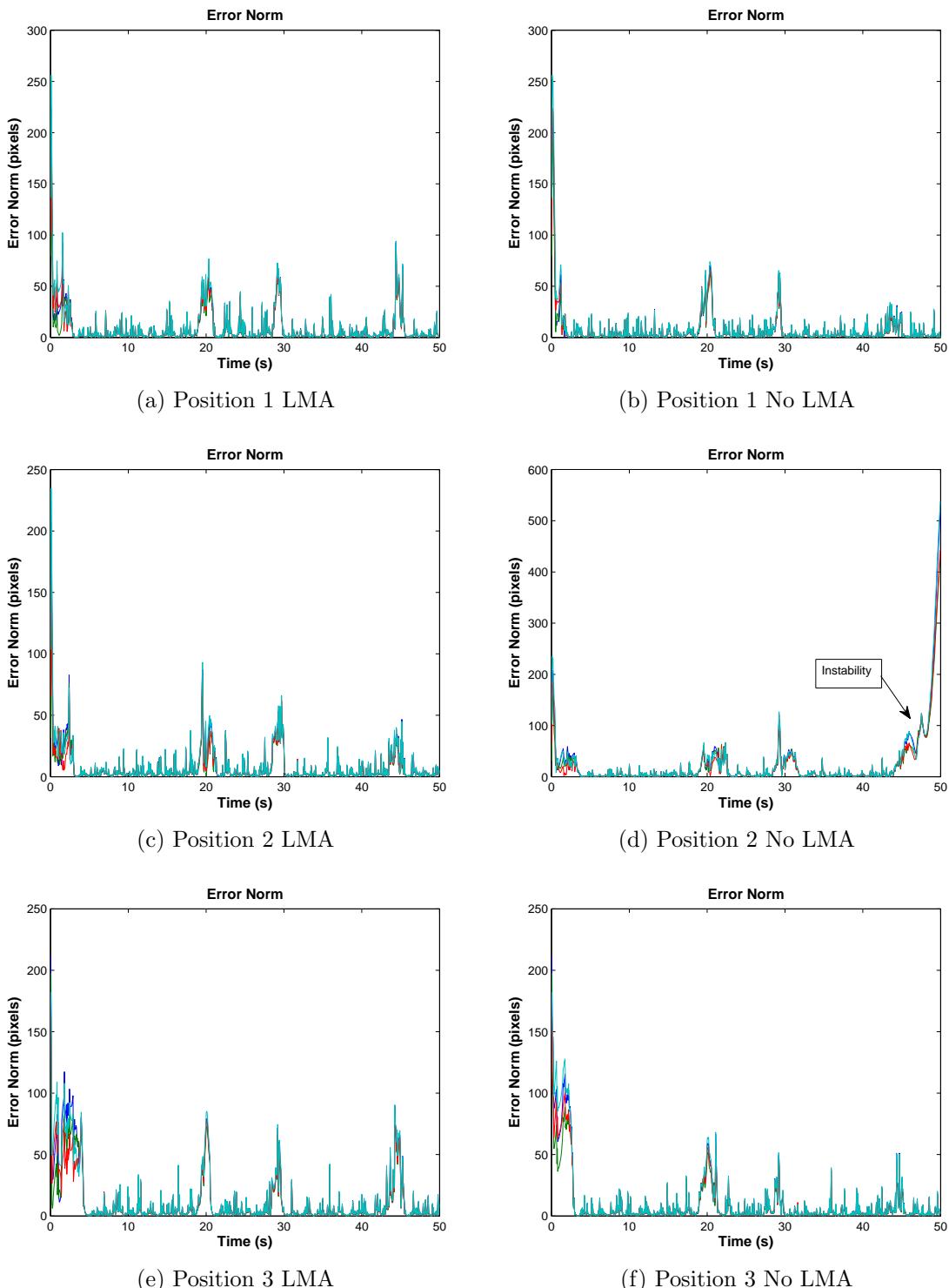


Figure 6.37: The RMS tracking error comparison of the switching MBFGS-DB algorithm with the LMA (left column) and without the LMA (right column) at various starting PUMA 560 robot configurations tracking four feature points of a cycloidal target trajectory using $v = 0.3$ and $\lambda = 0.5$.

Table 6.24: The RMS error and the settling time comparison for the switching MBFGS-DB algorithm with/without the LMA using $v = 0.3$ and $\lambda = 0.5$ at various starting PUMA 560 robot configurations tracking a cycloidal trajectory.

Position	LMA	RMS Error Camera 1 (pixels)	t_s (sec)
1	Yes	14.3171	3
	No	11.1803	2
2	Yes	11.8189	3
	No	60.5458 ²	4
3	Yes	15.2476	4.5
	No	10.5526	3

toward at the end as seen in Figure 6.35d for camera space view and is confirmed by average error norm plotted in Figure 6.37d. From this example, the LMA can help improving tracking stability.

6.4.3 Conclusion

Despite the fact that the LMA shows little improvement in convergence and tracking stability in a more complex cycloidal trajectory, this is at a trade-off for increased RMS tracking error. Overall performance for both RRR and PUMA 560 robot cases yield similar results of marginal effect implementing LMA into the switching algorithms for large residual tracking problems. However, the RMS tracking error is significantly affected by the value of the forgetting factor λ being used. As a result, various methods are investigated to optimally select λ_k at each iteration in Section 6.5. Then the effect of the LMA implementation is revisited as it is used in conjunction of a variable forgetting factor (VFF) algorithm to improve overall tracking performance.

²Instability occurs at the end of tracking

6.5 Performance Evaluations of the DAFF Method versus the Existing VFF Algorithms

Although the switching MBFGS-DB algorithm with/without the LMA implementation offers an efficient approach to achieve moving target tracking for large residual problems, this algorithm utilizes the recursive least-squares (RLS) algorithm in which its performance is dependent on the *exponential weighting factor* or *forgetting factor* λ . Different VFF algorithms presented in Chapter 5 are investigated in this section to validate tracking improvement, especially in the presence of measurement and processing noise. These algorithms are:

1. Fixed forgetting factor
2. VS-ARLS
3. GN-VFF-RLS
4. DAFF
5. Alternating between $[\lambda_{low}, \lambda_{high}]$

Due to the complexity of the GVFF-RLS algorithm [41], which requires the selection of a number of variables, the algorithm is excluded in this study.

Since all VFF algorithms require the selection of one or more parameters and a range of applicable λ , the effect of these parameters on each VFF algorithm performance is studied to reasonably select acceptable parameter ranges for simulation. A number of factors such as a target trajectory, robot degrees-of-freedom, and noise disturbance affects these values. To establish a basic understanding the PUMA 560 manipulator with an eye-in-hand camera tracking four feature points of a circular trajectory is used for an initial study and is presented in Section 6.5.1. Even though the obtained parameter values are not expected to be optimal due to the specific trajectory, a robot degrees-of-freedom, and a limited range of each test parameters,

this study helps to determine an applicable range for each parameter used in VFF algorithms. First no noise is added to the simulation and latter the effect of noise is investigated.

Each VFF algorithm is integrated into the switching MBFGS-DB algorithm so that λ_k is adaptively calculated at each iteration with a constant switching criterion v for the same testing conditions, i.e., same robot, camera configuration, and target trajectory, and is presented in Section 6.5.2. A variety of target trajectories including circular, square, and cycloidal trajectories are used to compare tracking performance. The effect of measurement and processing noise are studied with different camera arrangements to improve noise compensation.

The objective of this section is to compare the proposed DAFF algorithm for large-residual uncalibrated visual servoing with other existing VFF algorithms originally developed for adaptive filtering applications. Due to the large number of variations that may affect each algorithm, the study is limited to selected cases to establish the basic effectiveness of each VFF algorithm on improving tracking performance for large residual problems.

For simplicity only the name of each VFF algorithm is used. For example, the DAFF algorithm refers to the switching MBFGS-DB algorithm in which the DAFF algorithm is utilized for calculating λ_k using a given v .

6.5.1 Effect of Parameter Values on Each VFF Algorithm

The objective is to obtain an acceptable range of λ_k for each VFF scheme. In this section all evaluations are performed using the same circular trajectory and without additional noise. All VFF algorithms are implemented into the switching MBFGS-DB algorithm without the LMA. The switching criterion $v = 0.3$ is used for all tests.

The circular translational target trajectory in (6.4) is used to examine transient and steady-state behavior. The target consists of four feature points at the vertices

of a 50 mm square. To make the end-effector and target trajectories visually distinct they are offset by a constant vector $[-393.7, 19.9, 142]^T$ mm. The starting robot joint angles are $\theta_0 = [15.73^\circ, 132.5^\circ, -135.6^\circ, 15.73^\circ, -58.75^\circ, 14.27^\circ]^T$ for all tests simulated in this section. The end-effector camera has a 10 mm focal length and is coincident with the final frame of the robot. The sampling time is $h_t = 50$ ms. The initial Jacobian is estimated by successively perturbing each joint by a small angle.

Effect of λ on the fixed forgetting factor (FFF) scheme

Since the switching MBFGS-DB algorithm originally applies a fixed value of the forgetting factor λ , various values of λ are investigated with a circular trajectory.

As shown in Table 6.25 a small value of λ , such as $\lambda = 0.2$, generates a high RMS steady-state error, yet offers a small settling time t_s . In contrast, a higher value of λ yields a smaller RMS steady-state error with a higher t_s . Too small ($\lambda = 0.10$) or too great a value ($\lambda = 0.95$) of λ results in a longer settling time with greater RMS steady-state error and thus deteriorates the overall tracking performance.

Table 6.25: RMS error and t_s and settling time t_s for λ values in the FFF strategy.

λ	RMS Error (pixels)	t_s (sec)
0.10	2.2209	2
0.20	1.9225	1.5
0.50	1.6751	1.4
0.70	1.4800	1.6
0.80	1.0545	2
0.90	1.1080	3
0.95	14.9809	11.5

Table 6.25 shows that a range of $\lambda \in [0.20, 0.90]$ could be used for tracking a circular trajectory for the FFF strategy. The forgetting factor range $[\lambda_{min}, \lambda_{max}]$ plays an important role in determining the settling time t_s while achieving steady-state tracking.

Effect of η_1 on the VS-ARLS algorithm

Table 6.26 shows the effect of the parameter η_1 on the VS-ARLS algorithm. There exist singularity problems in the \hat{J}_k and \hat{H}_k calculation unless the minimum λ is greater than 0.6 so a range of $\lambda_k \in [0.60, 0.95]$ is used. Although varying η_1 values only show little effect on RMS error and no effect on settling time t_s (except for $\eta_1 = 0.01$), $\eta_1 = 0.05$ offers the smallest RMS error value. For this value of η_1 , different ranges of $[\lambda_{min}, \lambda_{max}]$ yield distinct RMS tracking errors and settling time t_s .

Table 6.26: RMS error and t_s comparison of the VS-ARLS algorithm for different values of η_1 with $\lambda_k \in [0.60, 0.95]$.

η_1	RMS Error	t_s (sec)
0.01	1.7321	1.6
0.05	1.0587	1.4
0.1	1.1015	1.4
0.2	1.4861	1.4
0.3	1.1573	1.4
0.5	1.6490	1.4
1	1.6490	1.4
5	1.6490	1.4

The overall performance, shown in Table 6.27, is similar for all tested λ_k ranges. This study shows that varying η_1 values and λ_k ranges yield only a small affect on the VS-ARLS algorithm. Therefore any combination is expected to give about the

Table 6.27: RMS error and t_s comparison of the VS-ARLS algorithm with $\eta_1 = 0.05$ for various ranges of $[\lambda_{min}, \lambda_{max}]$.

$[\lambda_{min}, \lambda_{max}]$	RMS Error	t_s (sec)
[0.60, 0.95]	1.0587	1.4
[0.70, 0.95]	1.5279	1.8
[0.80, 0.95]	1.1077	2
[0.85, 0.95]	0.9681	3

same result. However, $\eta_1 = 0.05$ and $\lambda \in [0.6, 0.95]$ are chosen due to slightly better performance.

Effect of η_2 on the GN-VFF-RLS algorithm

Table 6.28 shows the effect of the parameter η_2 on the GN-VFF-RLS algorithm with $\lambda_k \in [0.85, 0.95]$ to avoid \hat{J}_k and \hat{H}_k singularities.

Table 6.28: RMS error and t_s comparison of the GN-VFF-RLS algorithm for various values of η_2 with $\lambda_k \in [0.85, 0.95]$.

η_2	RMS Error	t_s (sec)
0.01	1.0633	2.5
0.05	1.4003	2.5
0.10	1.1667	3
0.50	1.0064	2.8
1	1.0553	3
5	1.4535	2

The RMS error and t_s insignificantly differ for varying values of η_2 . Since $\eta_2 = 0.01$ generates the smallest RMS error with the fastest t_s , it is used to investigate different ranges of forgetting factor in Table 6.29. For $\eta_2 = 0.01$, different λ ranges give similar results. Thus any presented range of λ can be selected. For this study $\eta_2 = 0.01$ with

$\lambda \in [0.85, 0.95]$ is selected for comparison with other VFF algorithms.

Table 6.29: RMS error and t_s comparison of the GN-VFF-RLS algorithm for various ranges of $[\lambda_{min}, \lambda_{max}]$ with $\eta_2 = 0.01$.

$[\lambda_{min}, \lambda_{max}]$	RMS Error	t_s (sec)
[0.85, 0.95]	1.0633	2.5
[0.9, 0.95]	0.9657	3.5

Effect of τ on the DAFF method

Table 6.30 shows the effect of τ with $\lambda \in [0.2, 0.95]$. Even though the RMS error value and the settling time t_s are insensitive to τ , the value $\tau = 0.1$ generates the smallest RMS tracking error and t_s .

Since the DAFF method is dependent on the maximum λ , the range of $\lambda \in [0.2, 0.90]$ is investigated as shown in Table 6.31 .

Table 6.30: RMS error and t_s comparison of the DAFF method for various values of τ with $\lambda_k \in [0.2, 0.95]$.

τ	RMS Error (pixels)	t_s (sec)
0.01	1.2028	1.5
0.05	1.3500	1.5
0.10	1.1868	1.4
0.50	1.2240	1.4
1	1.1930	1.5
5	1.1845	1.5
10	1.211	1.5

To study the effect of the upper bound $\lambda_k \in [0.50, 0.90]$ and $\lambda_k \in [0.50, 0.95]$ are tested with $\tau = (0.05, 0.10, 0.50, 1)$ with the RMS error and t_s shown in Table 6.32 and Table 6.33 respectively.

Table 6.31: The RMS error and t_s comparison of the DAFF method with various values of τ with $\lambda_k \in [0.20, 0.90]$.

τ	RMS Error (pixels)	t_s (sec)
0.05	1.0160	1.3
0.10	1.4389	1.5
0.50	0.9508	1.3
1	1.0181	1.5

Table 6.32: RMS error and t_s comparison of the DAFF method for various values of τ with $\lambda_k \in [0.50, 0.90]$.

τ	RMS Error (pixels)	t_s (sec)
0.05	1.2133	1.4
0.10	1.9085	1.4
0.50	0.9559	1.3
1	1.1067	1.4

Table 6.33: RMS error and t_s comparison of the DAFF method for various values of τ with $\lambda_k \in [0.50, 0.95]$.

τ	RMS Error (pixels)	t_s (sec)
0.05	4.6909	1.5
0.10	1.2457	1.3
0.50	1.7896	1.5
1	1.2966	1.4

From these simulation results $\tau \in [0.05, 1]$ offers approximately the same the average RMS error and the settling time t_s for different ranges of λ_k . Since the objective of the DAFF scheme is to adaptively calculate λ_k based on the current error norm $\|f_k\|$,

the widest range of permissible λ_k should be used. Therefore, the range $\lambda \in [0.2, 0.95]$ with $\tau = 0.1$ is selected.

Performance evaluation of the Alternating (Alt) method

In the Alternating (Alt) scheme $\lambda_k = \lambda_{low}$ if the error norm $\|f_k\| < e_{criterion}$ and $\lambda_k = \lambda_{high}$ if $\|f_k\| \geq e_{criterion}$. In this study $e_{criterion} \approx 0.01 \|f_{max}\|$ is selected since it is reasonably assumed that the steady-state error is approximately 1% of the maximum transient error. Different sets of $[\lambda_{low}, \lambda_{high}]$ are tested and shown in Table 6.34. The set $[0.50, 0.90]$ offers the lowest RMS error and the settling time t_s .

Table 6.34: RMS error and t_s for the Alt scheme between λ_{low} and λ_{high} .

$[\lambda_{low}, \lambda_{high}]$	RMS Error (pixels)	t_s (sec)
[0.20, 0.90]	1.0596	1.5
[0.50, 0.90]	0.9873	1.5
[0.70, 0.90]	0.9445	1.7
[0.80, 0.90]	2.3037	2
[0.20, 0.95]	1.3500	1.5
[0.50, 0.95]	1.3133	1.5
[0.70, 0.95]	1.3192	2
[0.80, 0.95]	1.3636	2

Conclusion

The performance of each VFF algorithm is insensitive to variation of parameter values, thus tracking performance of each VFF algorithm can be reasonably compared for selected values. Since selecting a minimal value of the RMS error usually yields a longer settling time t_s , proper parameter selection relies on the specified task objective. In this case the RMS error and t_s are equally weighted for their significance to

achieve fast tracking with the maximum steady-state tracking accuracy. Table 6.35 summarizes selected parameter values for each forgetting factor scheme.

Table 6.35: Selected parameters values for each forgetting factor scheme.

Scheme	$[\lambda_{min}, \lambda_{max}]$	Parameter	Value
Fixed λ	[0.50, 0.90]	λ	0.80
VS-ARLS	[0.60, 0.95]	η_1	0.05
GN-VFF-RLS	[0.85, 0.95]	η_2	0.01
DAFF	[0.20, 0.95]	τ	0.10
Alt	[0.50, 0.90]	λ_{low} λ_{high}	0.50 0.90

6.5.2 Performance Evaluation of Various VFF Algorithms

The RRR and the PUMA 560 robots are used to track feature points moving in a variety of trajectories to evaluate the effects of the switching MBFGS-DB implemented with each VFF algorithm, using the parameters in Table 6.35.

6.5.2.1 The RRR Robot

The same RRR robot used from Section 6.3.1 with the two eye-to-hand cameras is used for tracking the circular trajectory in (6.1) at a angular speed $\omega = 0.9$ rad/s. The starting robot joint angles are $\theta_0 = [60^\circ, 70^\circ, 50^\circ]^T$. The two camera arrangements are the same as shown in Figure 6.1a in which the locations are described by the homogeneous matrices in (6.2) and (6.3).

From Section 6.3.2 the NP-Jacobian yields better results than the P-Jacobian approximation and is utilized with the MBFGS-DB algorithm and $v = 0.3$ in this section. For each of the VFF algorithms the camera space view, the task space view, and average error norm are plotted in Figure 6.38, Figure 6.39, and Figure 6.40

respectively. To show the difference in forgetting factors, λ_k plots are also shown in Figure 6.40. The settling time t_s and the RMS steady-state error of each forgetting factor are summarized in Table 6.36.

Table 6.36: The RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras tracking one feature point of a circular target moving at $\omega = 0.9$ rad/s and $v = 0.3$ for the no additional noise scenario.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
FFF $\lambda_k = 0.8$	0.0499	0.0498	3.2
VS-ARLS $\lambda_k \in [0.60, 0.95]$ $\eta_1 = 0.05$	0.0682	0.0688	2.2
GN-VFF-RLS $\lambda_k \in [0.85, 0.95]$ $\eta_2 = 0.01$	0.0690	0.0689	3.4
DAFF $\lambda_k \in [0.20, 0.95]$ $\tau = 0.1$	0.0842	0.0842	1
Alt $\lambda_k = 0.50$ or $\lambda_k = 0.90$	0.1358	0.1317	1.5

As seen on the camera space in Figure 6.38 and the task space in Figure 6.39 the DAFF algorithm moves the EE to the desired target in the most direct path and has the fastest settling time. The DAFF scheme and the Alt algorithm calculate λ_k similarly.

The effects of measurement noise

$\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points. Since it is recommended in [57] to select $\lambda_k \gg 0$ to prevent excessive estimation

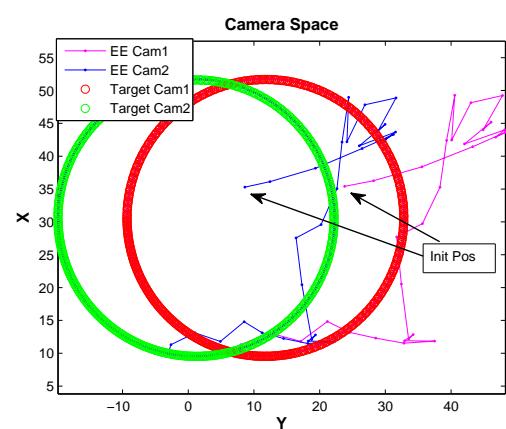
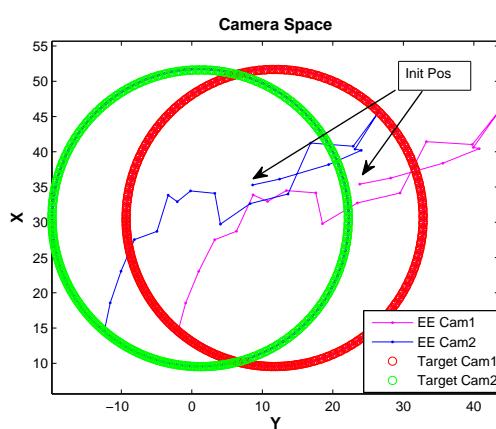
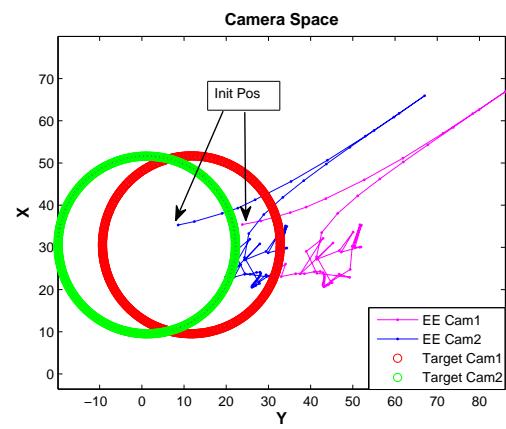
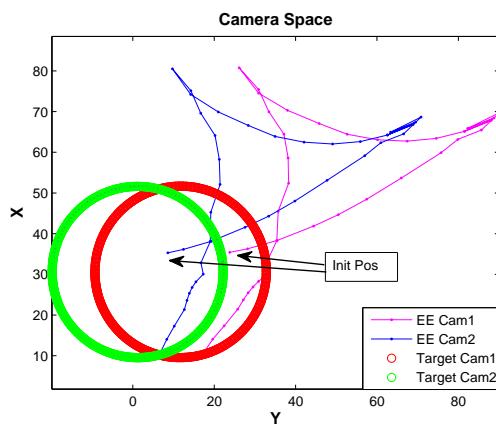
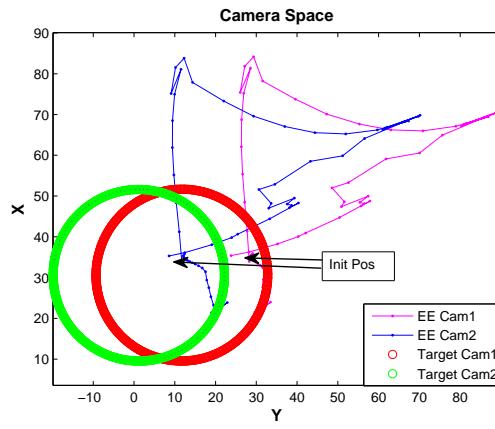
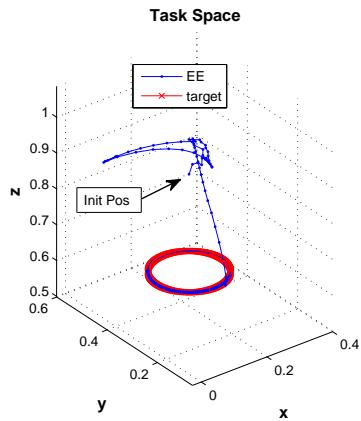
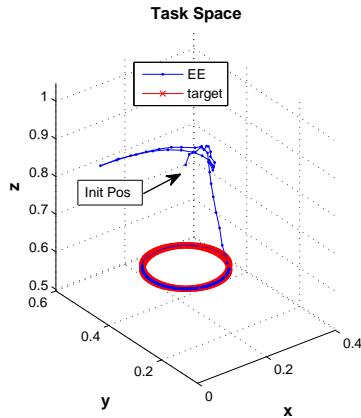


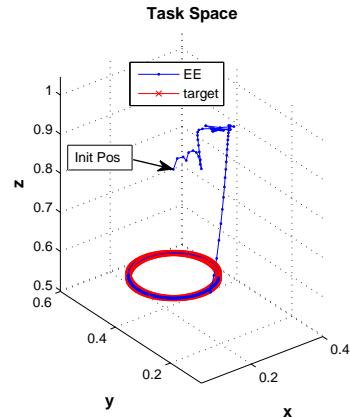
Figure 6.38: The camera space of the RRR manipulator with two eye-to-hand cameras tracking one feature point of the circular target trajectory moving at $\omega = 0.90 \text{ rad/s}$. Various VFF algorithms for λ_k are implemented with the switching MBFGS-DB with $v = 0.3$.



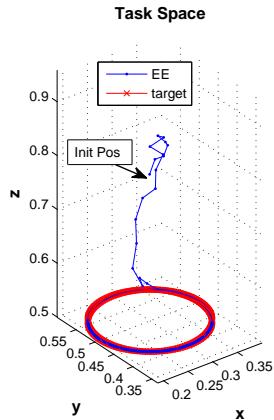
(a) FFF



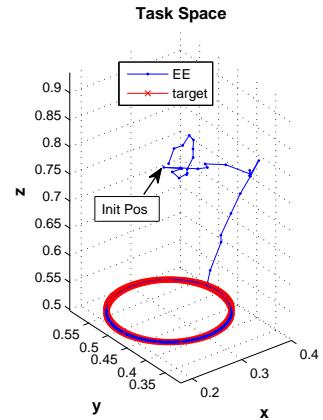
(b) VS-ARLS



(c) GN-VFF-RLS

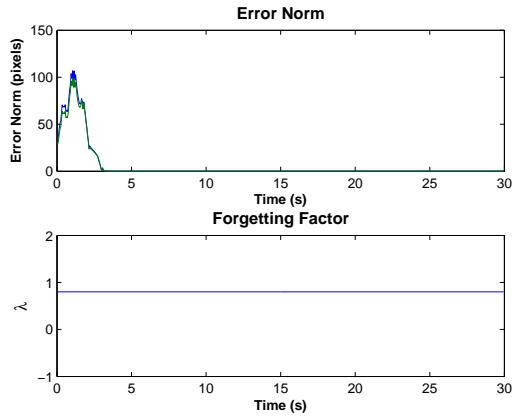


(d) DAFF

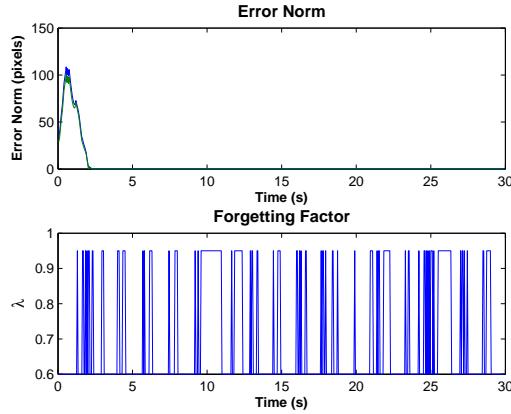


(e) Alt

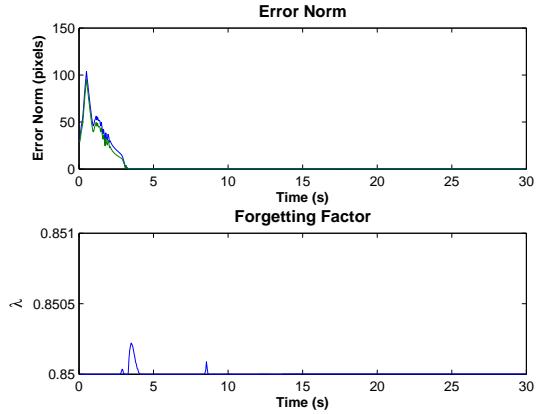
Figure 6.39: The task space view showing one target point for clarity (the other views are similar) for the RRR manipulator with two eye-to-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90 \text{ rad/s}$.



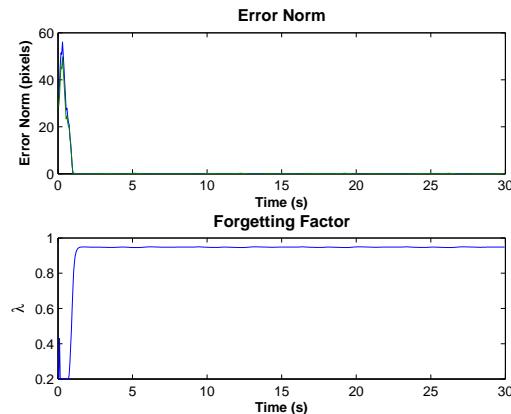
(a) FFF



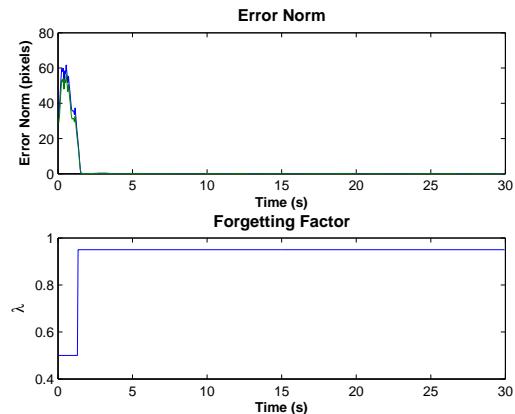
(b) VS-ARLS



(c) GN-VFF-RLS



(d) DAFF



(e) Alt

Figure 6.40: The error norm (top) and the forgetting factor λ_k (bottom) for the RRR manipulator with two eye-to-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The forgetting factor λ_k The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90 \text{ rad/s}$.

error for high noise levels so $\lambda_{max} = 0.98$ is used for all methods. Due to the small lower bound of λ used in the DAFF method for the no noise case, the range of $\lambda_k \in [0.50, 0.98]$ is employed. For noisy target and EE measurements the switching criterion is adjusted to $v = 0.5$ to avoid divergence. NP-Jacobian approximations are still utilized for all algorithms. Figure 6.41 and Figure 6.42 show the camera space view and the task space view respectively. The RMS error and t_s are summarized in Table 6.37.

Table 6.37: The RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras tracking one feature point of a circular target moving at $\omega = 0.9$ rad/s. The switching criterion $v = 0.5$ is used when $\pm\frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
FFF $\lambda_k = 0.8$	1.2412	1.1195	4.5
VS-ARLS $\lambda_k \in [0.60, 0.98]$ $\eta_1 = 0.05$	1.1783	1.1934	2
GN-VFF-RLS $\lambda_k \in [0.85, 0.98]$ $\eta_2 = 0.01$	1.5366	1.6139	4
DAFF $\lambda_k \in [0.50, 0.98]$ $\tau = 0.1$	1.1979	1.1849	1.5
Alt $\lambda_k = 0.50$ or $\lambda_k = 0.98$	1.4056	1.4013	1.5

Although the RMS error values of all algorithms in Table 6.37, and as seen on camera space in Figure 6.41, are not significantly different, the EE motion substantially deviates from the desired target plane in the $\pm z$ direction as shown in Figure

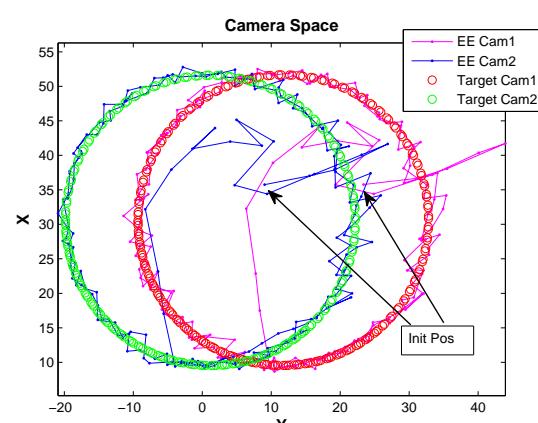
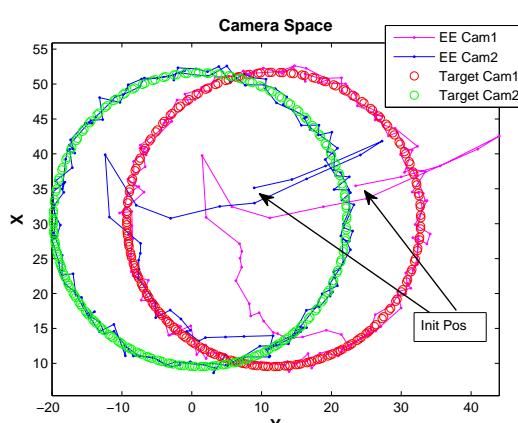
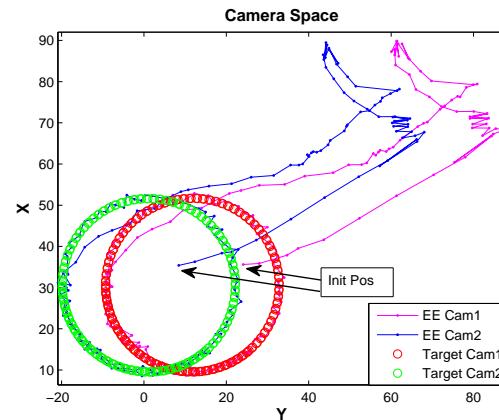
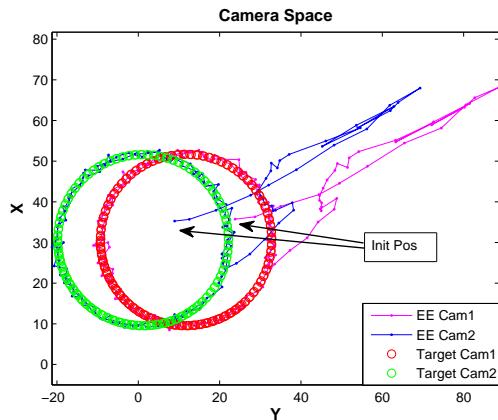
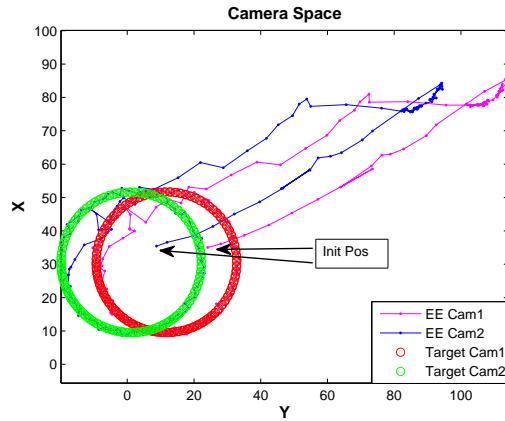


Figure 6.41: The camera space of the RRR manipulator with two eye-to-hand cameras tracking one feature point of a circular target trajectory moving at $\omega = 0.90 \text{ rad/s}$. Various VFF algorithms for λ_k are implemented with the switching MBFGS-DB with $v = 0.5$ for which $\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.

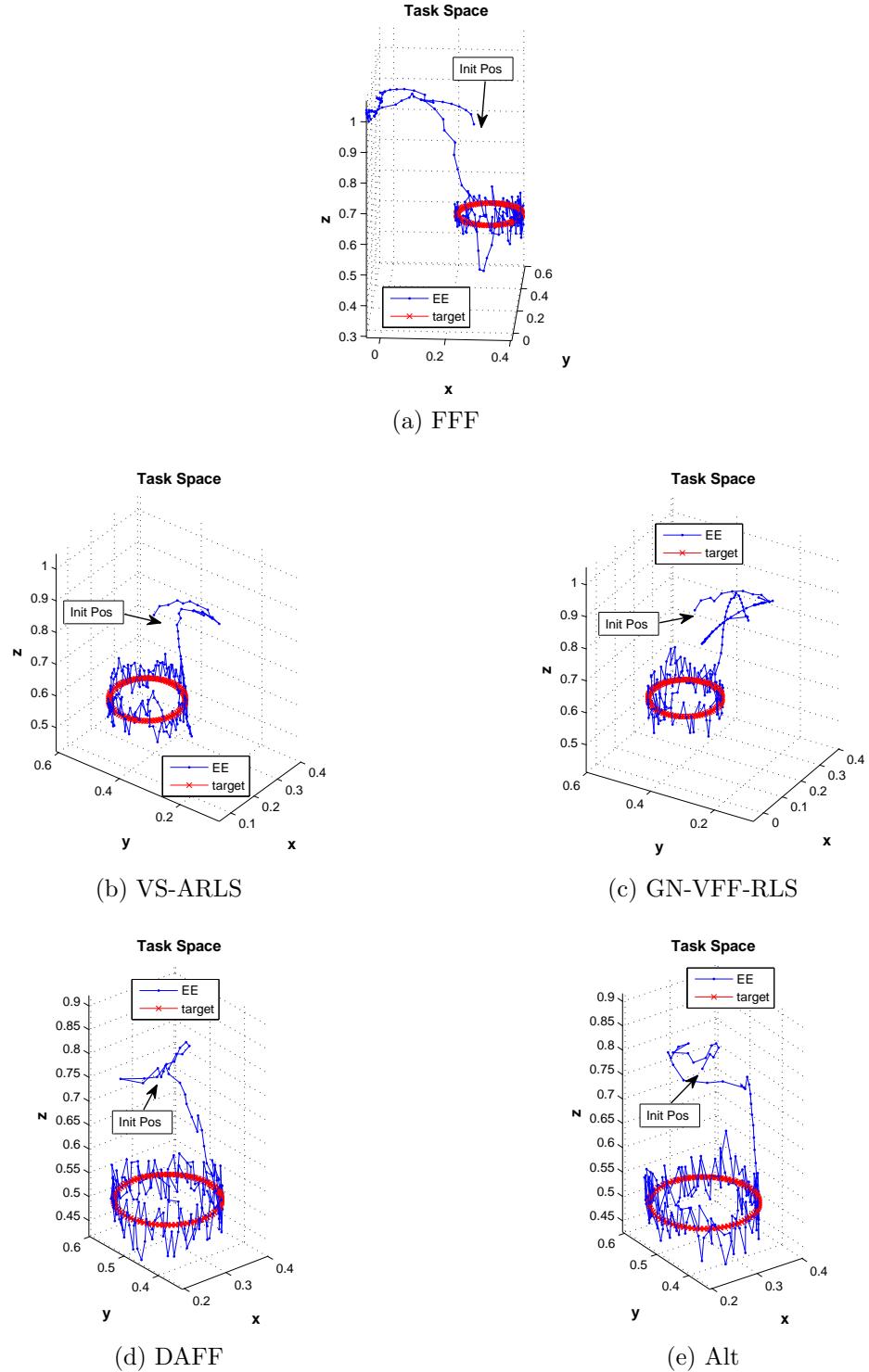


Figure 6.42: The task space view showing one camera and one target point for clarity (the other view is similar) for the RRR manipulator with two eye-to-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s for which $\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.

6.42. However, the image errors f_k are in a range of ± 1 pixel. This result indicates that the switching MBFGS-DB algorithm does not amplify the noise since uniform $\pm \frac{1}{2}$ pixel noise is added to both target and EE feature points. The significant z direction deviation is due to the fact that the two cameras are nearly parallel and primarily provide information for X-Y plane tracking.

Alternatively, the cameras are re-arranged so their optical axes are perpendicular, one optical axis is pointed into the z direction and the other one is pointed into the $-x$ direction, as shown in Figure 6.45. The cameras are located by the homogeneous matrices in an arrangement similar to [25],

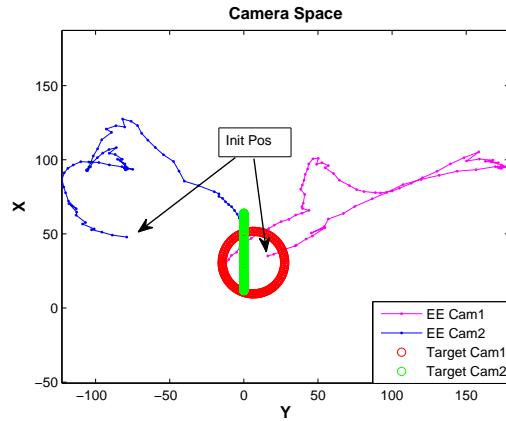
$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 0.4453 \\ 0 & 1 & 0 & 0.5307 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.15)$$

$$T_2 = \begin{bmatrix} 1 & 0 & 0 & 0.4453 \\ 0 & 0 & -1 & 2.5307 \\ 0 & 1 & 0 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.16)$$

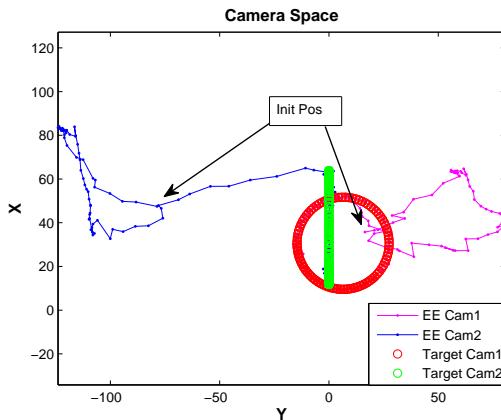
Even though the RMS errors in Table 6.38 are not noticeably improved over Table 6.37, the errors in the EE motion in the Cartesian space plane of motion are significantly minimized as seen in Figure 6.44. More cameras may be added and investigated to improve tracking error, which is beyond the scope of this study.

The effects of additional system noise

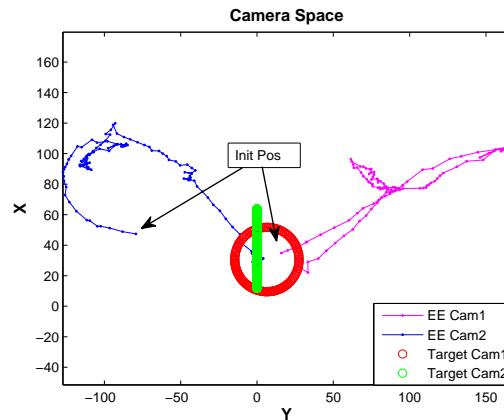
To investigate the effects of system noise, ± 1 mm noise is added to the EE location in addition to $\pm \frac{1}{2}$ pixel uniform noise added to the EE and target feature points. Initially the nearly parallel camera arrangement in (6.2) and (6.3) are used for tracking. A



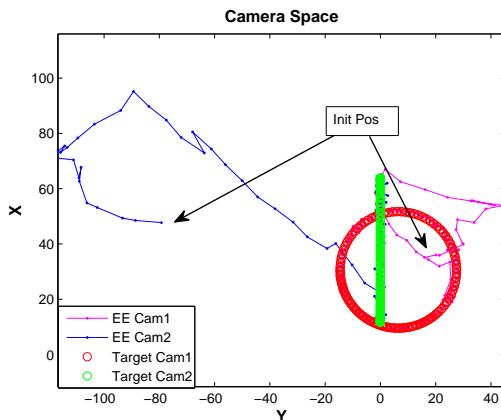
(a) FFF



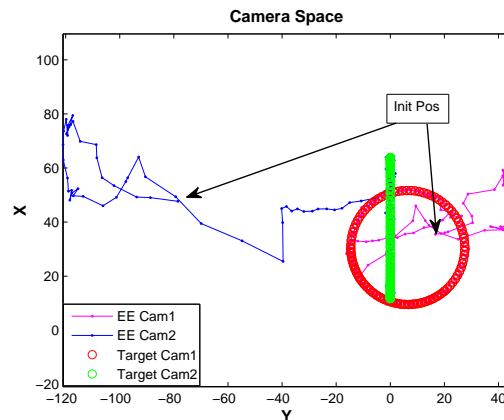
(b) VS-ARLS



(c) GN-VFF-RLS



(d) DAFF



(e) Alt

Figure 6.43: The camera space of the RRR manipulator with two eye-to-hand cameras perpendicularly arranged tracking one feature point of the circular target trajectory moving at $\omega = 0.90$ rad/s. Various VFF algorithms for λ_k are implemented into the switching MBFGS-DB with $v = 0.5$ for which $\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.

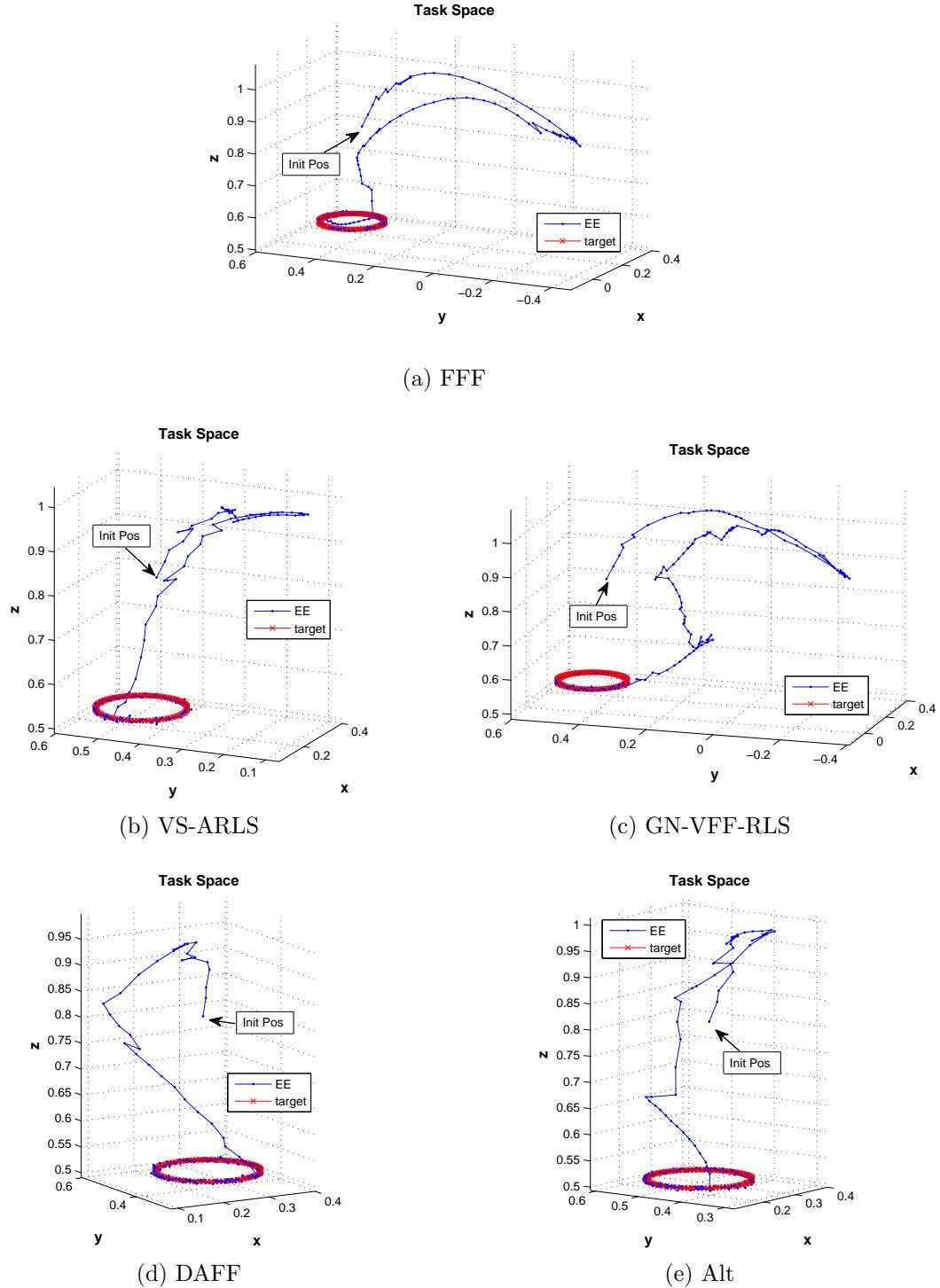


Figure 6.44: The task space view showing one camera and one target point for clarity (the camera view is similar) for the RRR manipulator with two eye-to-hand cameras perpendicularly arranged using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s for which $\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.

Table 6.38: RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras perpendicularly arranged. The target moves in a circular trajectory with an angular speed of $\omega = 0.9$ rad/s. $\pm \frac{1}{2}$ pixel uniform quantization noise is added to the target and EE feature points.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
FFF $\lambda_k = 0.85$	1.0289	1.1349	4.5
VS-ARLS $\lambda_k \in [0.60, 0.98]$ $\eta_1 = 0.05$	1.0541	1.1502	2.5
GN-VFF-RLS $\lambda_k \in [0.85, 0.98]$ $\eta_2 = 0.01$	1.9373	1.8306	6
DAFF $\lambda_k \in [0.50, 0.98]$ $\tau = 0.1$	1.0538	1.1913	1.8
Alt $\lambda_k = 0.50$ or $\lambda_k = 0.98$	1.1297	1.2118	2.6

summary of the RMS error and t_s is shown in Table 6.39. In comparison with Table 6.37 where only measurement noise is added, there is little affect on RMS error value and t_s . The camera space view is shown in Figure 6.46 while the EE deviates from the desired target plane shown in Figure 6.47 similar to adding measurement noise only.

When the perpendicular camera arrangement of (6.15) and (6.16) is also used, there is a similar improvement as the measurement noise only case. A summary of the RMS error and t_s is in Table 6.40. Only the task space view is shown in Figure 6.48 due to the similarity of results.

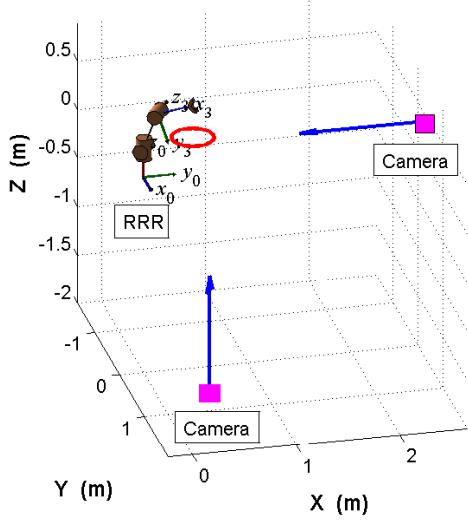


Figure 6.45: A perpendicular camera arrangement where one camera is pointed in the z direction while the other camera is pointed into the $-x$ direction is used with the RRR robot for noise compensation.

Conclusion

The switching MBFGS with the DAFF algorithm yields the fastest convergence with desirable RMS tracking error as compared to the other VFF algorithms when the $\pm \frac{1}{2}$ pixel uniform measurement noise is added to the target and robot feature points and ± 1 mm uniform system noise is added to the EE location. However, with the nearly-parallel camera arrangement, the EE motion substantially deviates from the desired target plane for all algorithms. This problem is significantly improved when the perpendicular camera arrangement is used.

For the nearly parallel camera arrangement, only X-Y plane tracking information is available thus the EE considerably deviates in the z direction. Rearranging the cameras so that their optical axes are perpendicular significantly diminishes EE deviation in the z direction in the presence of measurement and system noise.

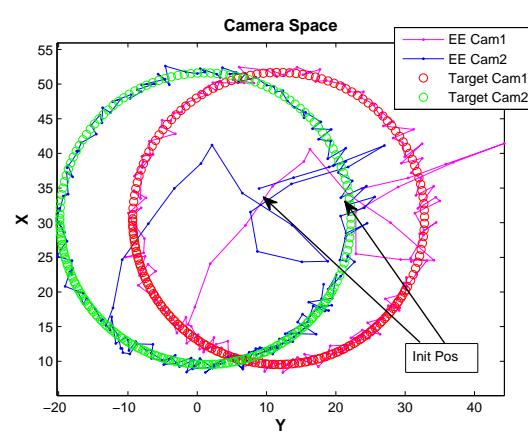
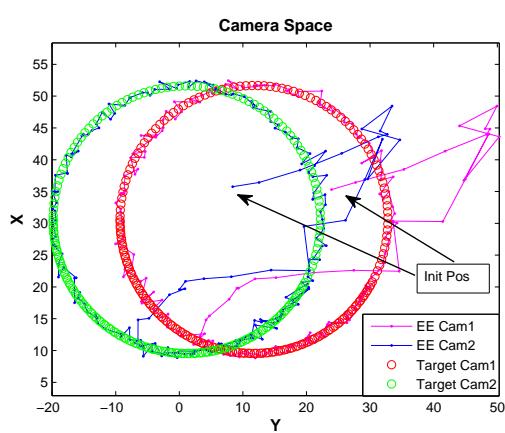
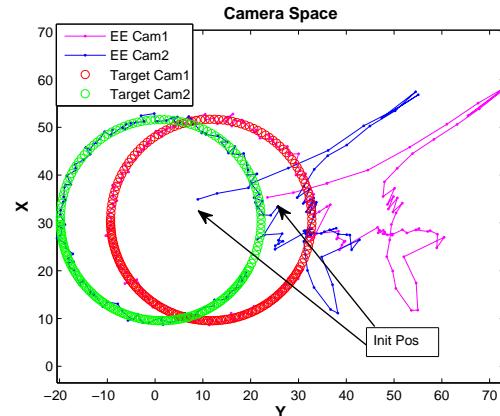
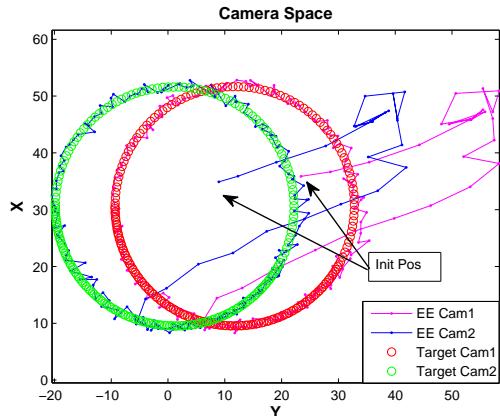
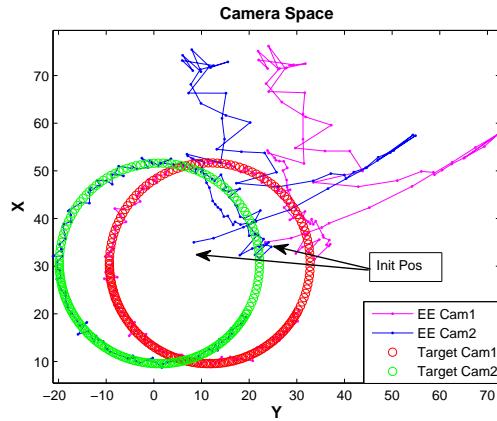


Figure 6.46: The camera space of the RRR manipulator with two eye-to-hand cameras tracking one feature point of a circular target trajectory moving at $\omega = 0.90 \text{ rad/s}$. Various VFF algorithms for λ_k are implemented with the switching MBFGS-DB with $v = 0.5$ for which $\pm 1 \text{ mm}$ noise is added to the EE location in addition to $\pm \frac{1}{2} \text{ pixel}$ uniform quantization noise added to the target and EE feature points

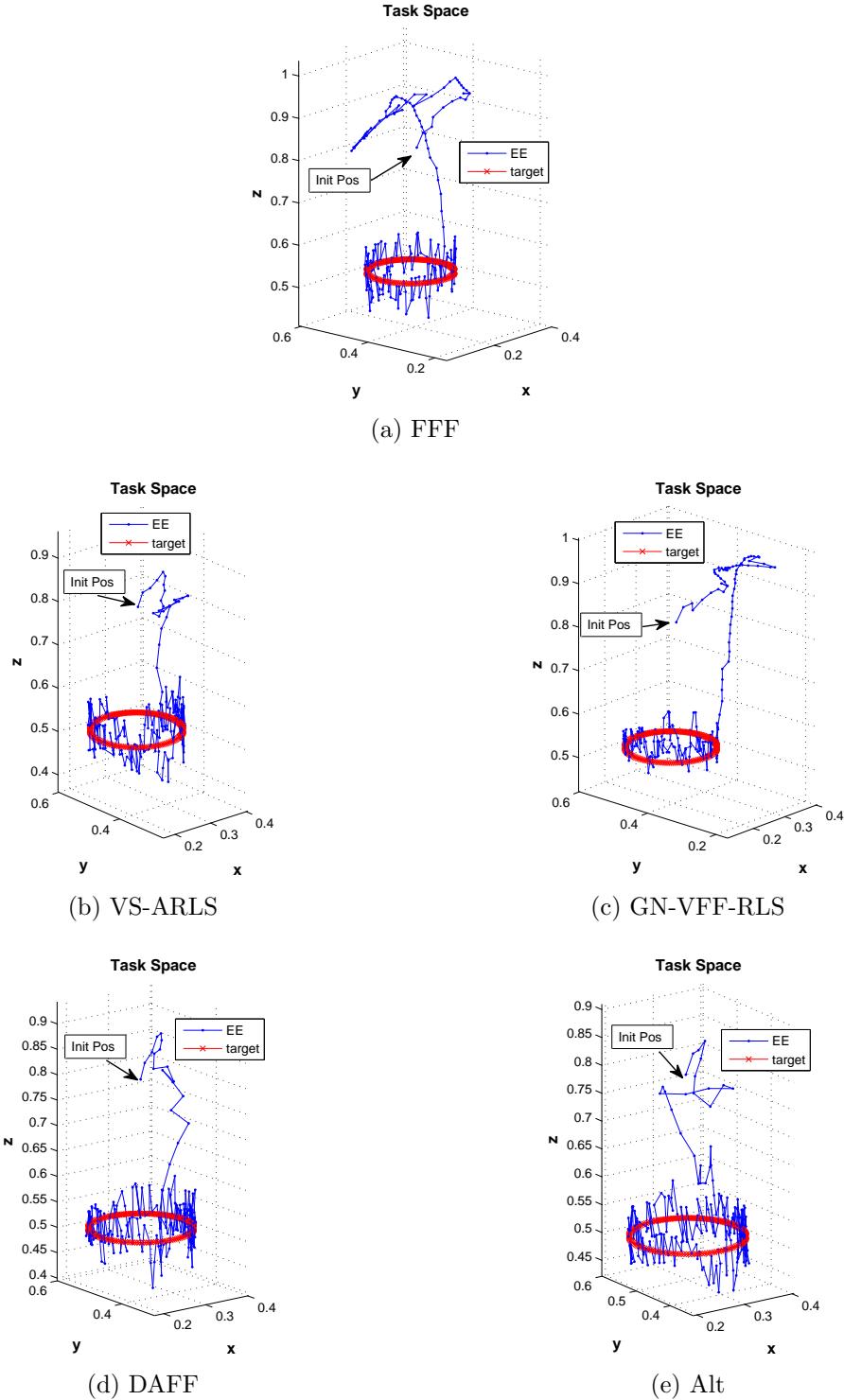


Figure 6.47: The task space view showing one camera and one target point for clarity (the other view is similar) for the RRR manipulator with two eye-to-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90 \text{ rad/s}$ for which $\pm 1 \text{ mm}$ noise is added to the EE location in addition to $\pm \frac{1}{2} \text{ pixel}$ uniform quantization noise added to the target and EE feature points.

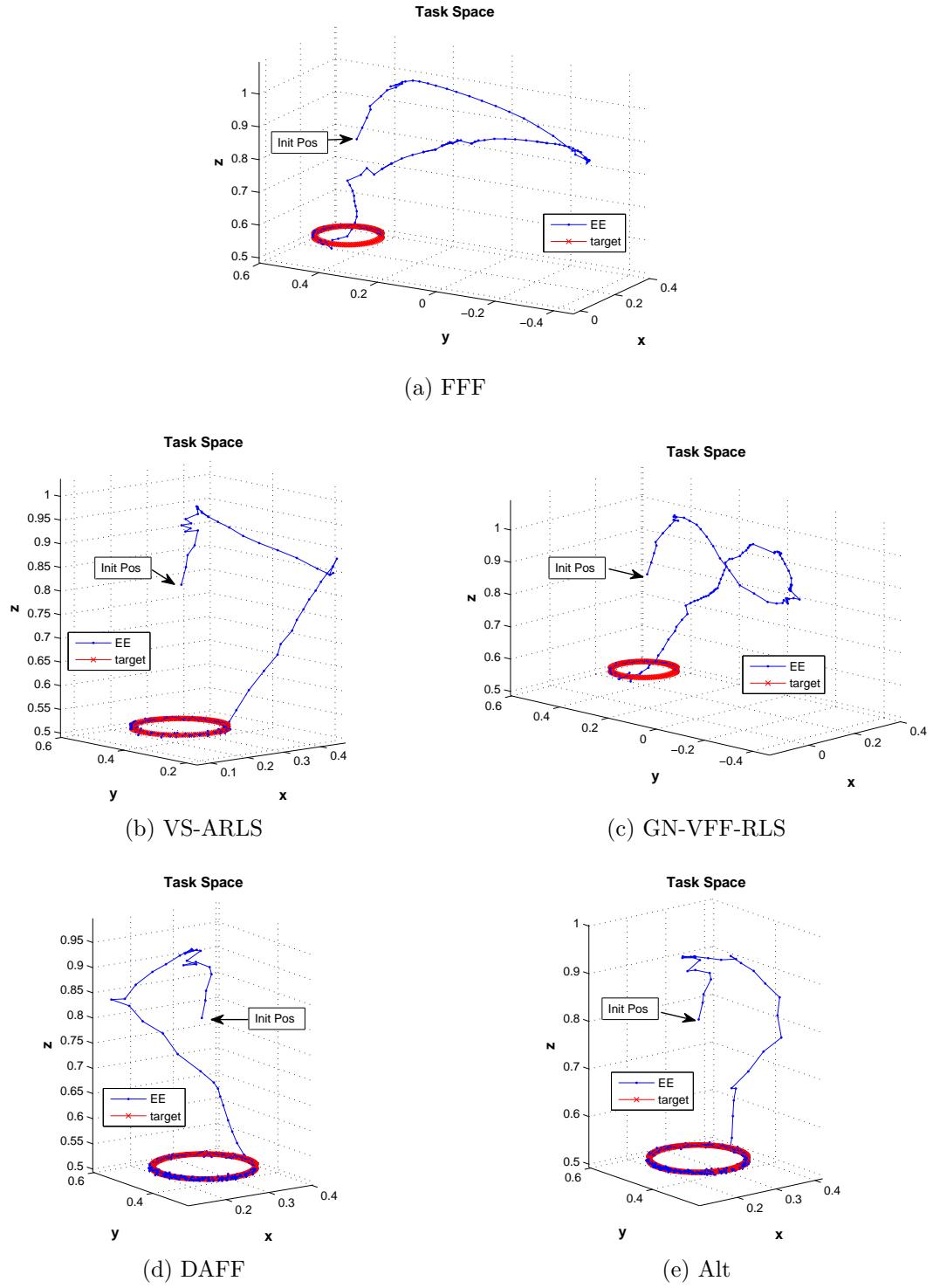


Figure 6.48: The task space view showing one camera and one target point for clarity (the camera views are similar) for the RRR manipulator with two eye-to-hand cameras perpendicularly arranged using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking one feature point of a circular target trajectory moving at $\omega = 0.90$ rad/s for which ± 1 mm noise is added to the EE location in addition to $\pm \frac{1}{2}$ pixel uniform quantization noise added to the target and EE feature points.

Table 6.39: The RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras tracking one feature point of a circular target moving at $\omega = 0.9$ rad/s. The switching criterion $v = 0.5$ is used when ± 1 mm noise is added to the EE location in addition to $\pm \frac{1}{2}$ pixel uniform quantization noise added to the target and EE feature points.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
FFF $\lambda_k = 0.85$	1.448	1.3009	3.5
VS-ARLS $\lambda_k \in [0.60, 0.98]$ $\eta_1 = 0.05$	1.1790	1.1541	1.5
GN-VFF-RLS $\lambda_k \in [0.85, 0.98]$ $\eta_2 = 0.01$	1.3218	1.2722	3.5
DAFF $\lambda_k \in [0.50, 0.98]$ $\tau = 0.1$	1.0925	1.0802	1.5
Alt $\lambda_k = 0.50$ or $\lambda_k = 0.98$	1.2788	1.3244	1

6.5.2.2 The PUMA 560 Robot

Circular Trajectory

The PUMA 560 with one eye-in-hand camera tracks four feature points moving in the circular path as described in Section 6.5.1. The parameters listed in Table 6.35 are used for each VFF algorithm.

The testing setup is similar to Section 6.3.4 is used where the target consists of four feature points at the vertices of a 50 mm square. The starting robot joint angles are $\theta_0 = [15.73^\circ, 132.5^\circ, -135.6^\circ, -4.27^\circ, -108.75^\circ, 14.27^\circ]^T$ for all tests in this section. To make the end-effector and target trajectories visually distinct they are

Table 6.40: The RMS error and the settling time comparison for VFF schemes using the RRR robot with two eye-to-hand cameras tracking one feature point of a circular target moving at $\omega = 0.9$ rad/s. The switching criterion $v = 0.5$ is used when ± 1 mm noise is added to the EE location in addition to $\pm \frac{1}{2}$ pixel uniform quantization noise added to the target and EE feature points.

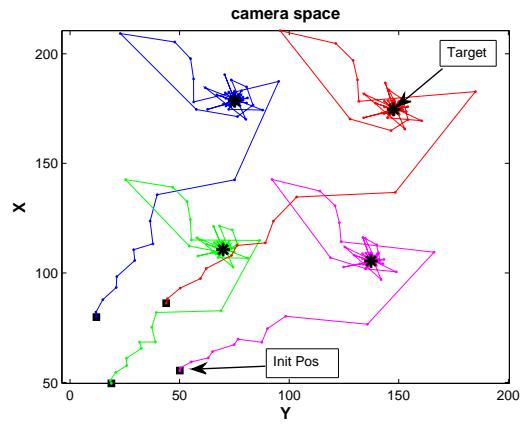
Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
FFF $\lambda_k = 0.85$	1.1119	1.2568	5.5
VS-ARLS $\lambda_k \in [0.60, 0.98]$ $\eta_1 = 0.05$	1.0480	1.1592	2.5
GN-VFF-RLS $\lambda_k \in [0.85, 0.98]$ $\eta_2 = 0.01$	1.0262	1.4081	5
DAFF $\lambda_k \in [0.50, 0.98]$ $\tau = 0.1$	1.00417	1.0737	2
Alt $\lambda_k = 0.50$ or $\lambda_k = 0.98$	1.0542	1.1084	1.6

offset by a constant vector $[-393.7, 19.9, 142]^T$ mm. The end-effector camera has a 10 mm focal length and is coincident with the final frame of the robot. The sampling time is $h_t = 50$ ms. The initial Jacobian is estimated by successively perturbing each joint by a small angle.

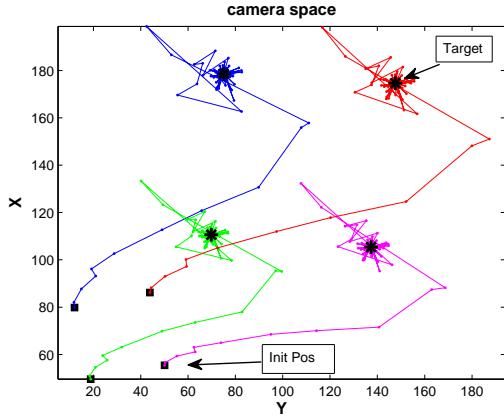
The camera space view, the task space view, and the RMS error norm are plotted in Figure 6.49, Figure 6.50, and Figure 6.51 respectively. λ_k is also shown in Figure 6.51. The RMS error and t_s are summarized in Table 6.41.

From Table 6.41 all methods offer approximately the same RMS error and t_s for circular trajectory tracking when no noise is added to the target feature points.

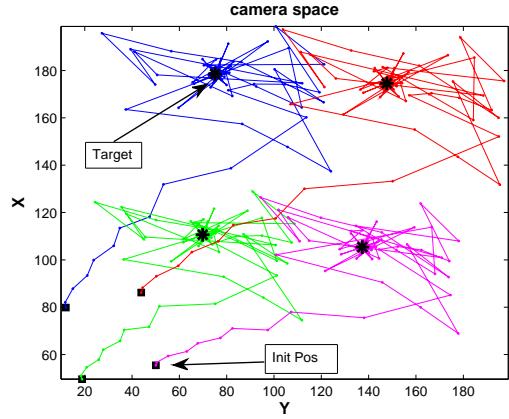
To study how each algorithm deals with faster tracking, the target speed is doubled



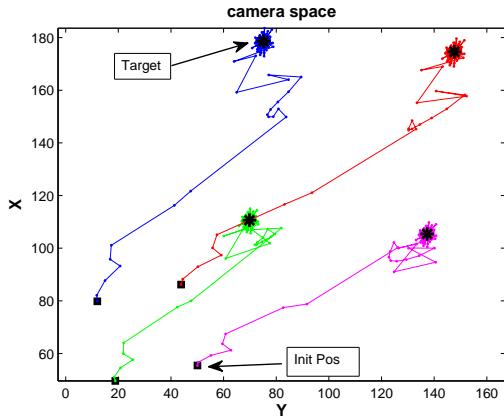
(a) FFF



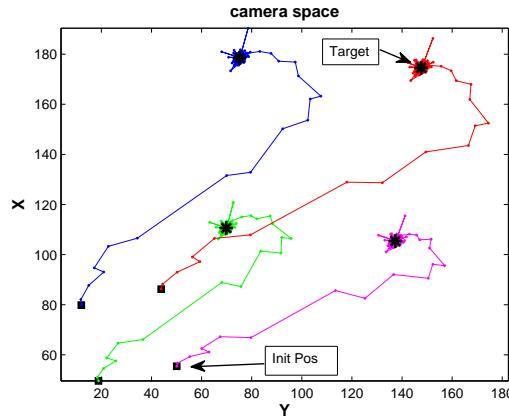
(b) VS-ARLS



(c) GN-VFF-RLS



(d) DAFF



(e) Alt

Figure 6.49: The camera space of the PUMA 560 manipulator with the eye-in-hand camera tracking four feature points of the circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. VFF algorithms for λ_k calculation are implemented into the switching MBFGS-DB with $v = 0.3$. No additional noise is added.

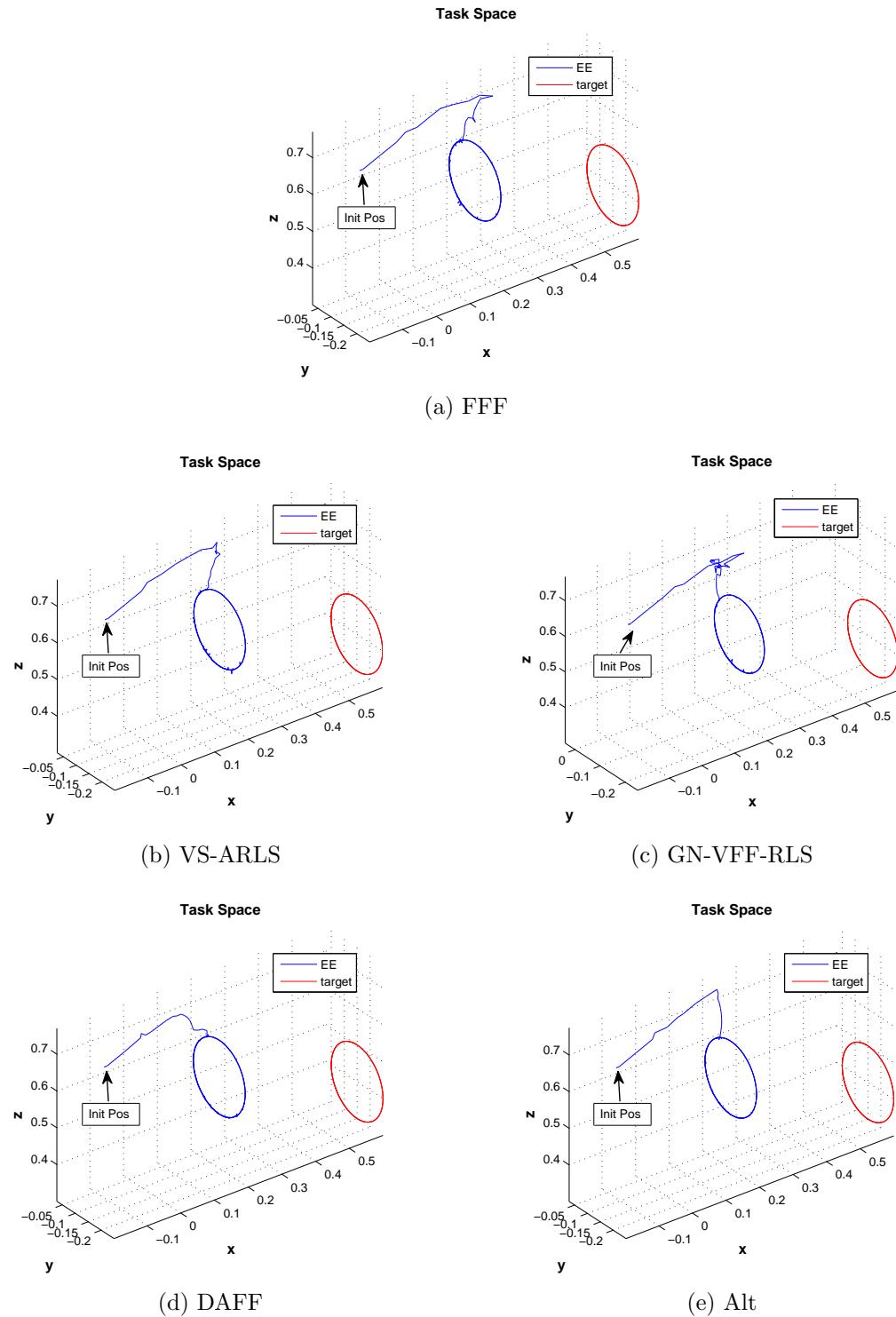
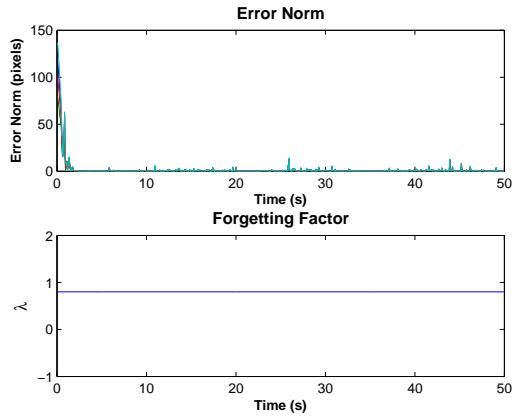
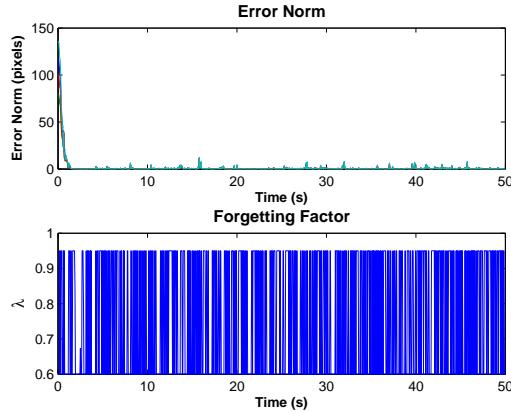


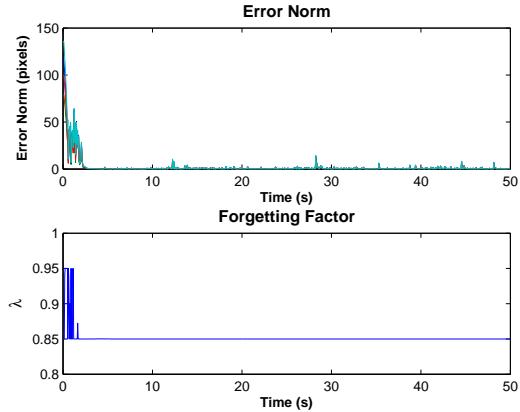
Figure 6.50: The task space view showing one camera and one target point for clarity (the other views are similar) of the PUMA 560 manipulator with the eye-in-hand camera tracking four feature points of the circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. VFF algorithms for λ_k calculation are implemented into the switching MBFGS-DB with $v = 0.3$. No additional noise is added.



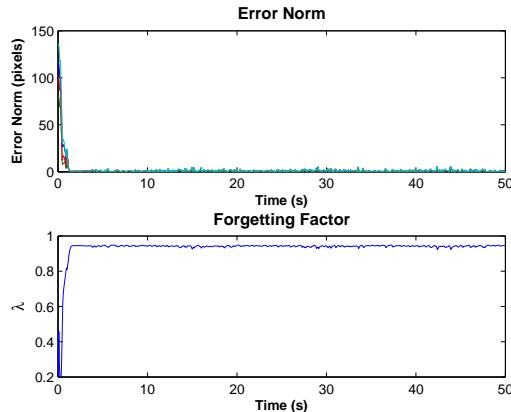
(a) FFF



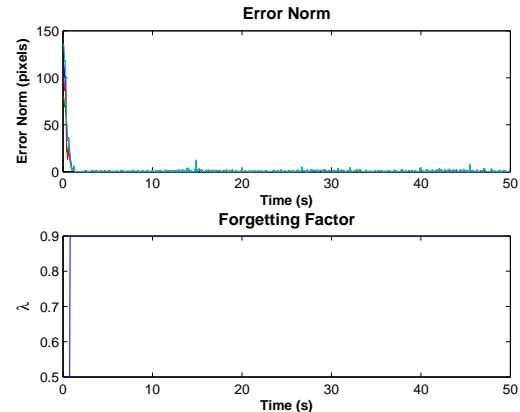
(b) VS-ARLS



(c) GN-VFF-RLS



(d) DAFF



(e) Alt

Figure 6.51: The error norm of the PUMA 560 manipulator with an eye-in-hand camera tracking four feature points of the circular target trajectory moving at $\omega = 0.45$ rad/s. VFF algorithms for λ_k calculation are implemented into the switching MBFGS-DB with $v = 0.3$. No additional noise is added.

Table 6.41: The RMS error and the settling time comparison for VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a circular target moving at $\omega = 0.45$ rad/s. The switching criterion is $v = 0.3$ and no additional noise is added.

Scheme	RMS Error (pixels)	t_s (sec)
FFF $\lambda_k = 0.8$	1.0545	2
VS-ARLS $\lambda_k \in [0.60, 0.95]$ $\eta_1 = 0.05$	1.0587	1.4
GN-VFF-RLS $\lambda_k \in [0.85, 0.95]$ $\eta_2 = 0.01$	1.0645	3
DAFF $\lambda_k \in [0.20, 0.95]$ $\tau = 0.1$	1.1868	1.4
Alt $\lambda_k = 0.50$ or $\lambda_k = 0.90$	0.9864	1.5

to, $\omega = 0.9$ rad/s. The RMS tracking error and t_s are summarized in Table 6.42.

Table 6.42: The RMS error and the settling time comparison for VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a circular target moving at a faster angular speed $\omega = 0.9$ rad/s. The switching criterion is $v = 0.3$ and no additional noise is added.

Scheme	RMS Error (pixels)	t_s (sec)
FFF	4.1006	2
VS-ARLS	4.4970	2
GN-VFF-RLS	3.8473	2.5
DAFF	3.6483	1.7
Alt	3.8368	2.5

Although the performance of these algorithms are similar, the DAFF method gives the smallest RMS error and settling time.

The effects of measurement noise

Uniform quantization noise of ± 1 pixel is added to the target feature points requiring that the parameters in Table 6.35 are updated to those in Table 6.43 to avoid divergence. The total added noise in the RRR case is ± 1 pixel ($\pm \frac{1}{2}$ pixel added to target and EE feature points). Since for the eye-in-hand case noise can only be added to the target feature points, ± 1 pixel is added. The switching criterion $v = 0.5$ is used for all algorithms. However, the GN-VFF-RLS algorithm diverges for all ranges of λ tested so it is not included. The camera space view, the task space view, and the RMS error with λ_k plots are shown in Figure 6.52, Figure 6.53, and Figure 6.54 respectively. The RMS error and t_s is summarized in Table 6.43.

Table 6.43: The RMS error and the settling time comparison for VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a circular target moving at the angular speed $\omega = 0.45$ rad/s. The switching criterion is $v = 0.5$. Uniform quantization noise of ± 1 pixel is added to the target feature points.

Scheme	RMS Error (pixels)	t_s (sec)
FFF $\lambda_k = 0.98$	5.2661	10
VS-ARLS $\lambda_k \in [0.90, 0.98]$ $\eta_1 = 0.05$	18.9683	3
DAFF $\lambda_k \in [0.85, 1]$ $\tau = 0.1$	3.5424	2
Alt $\lambda_k = 0.70$ or $\lambda_k = 0.98$	4.5215	2

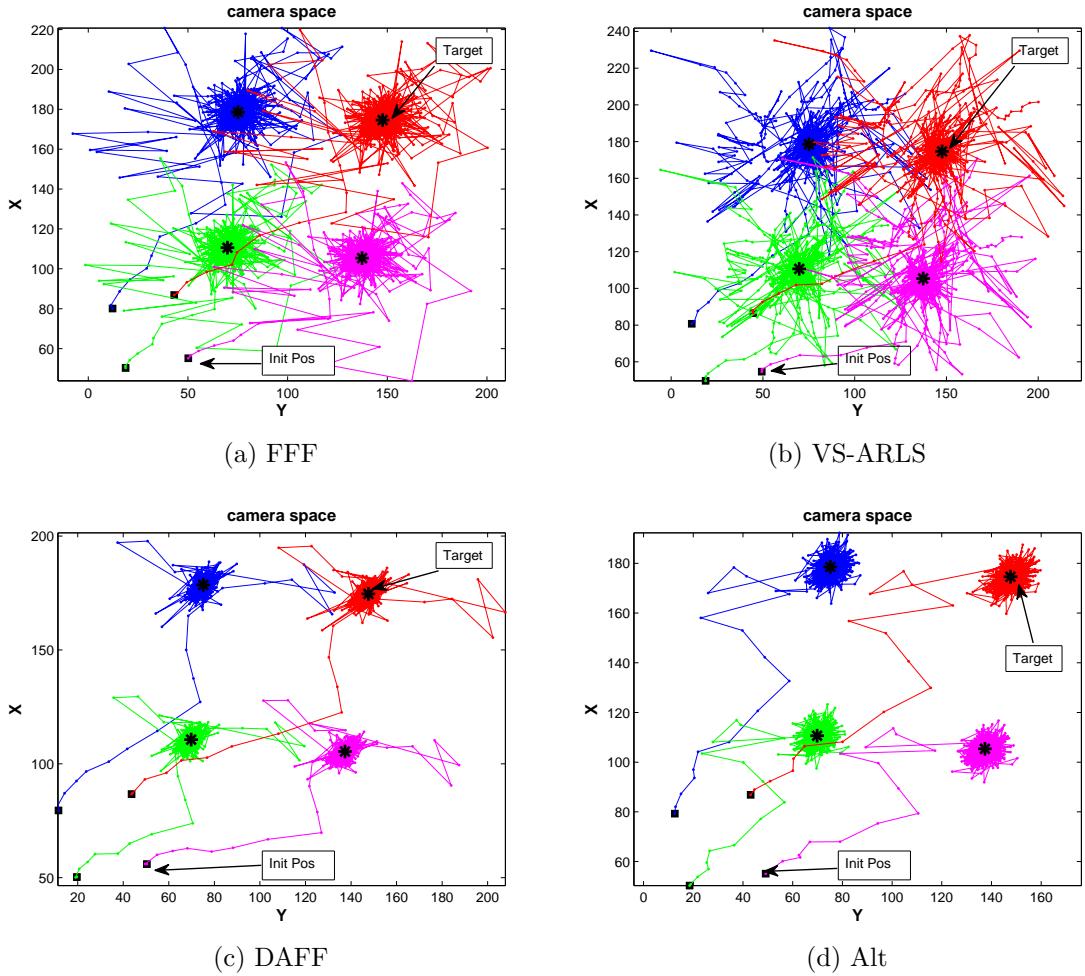


Figure 6.52: The camera space of the PUMA 560 manipulator with an eye-in-hand camera tracking four feature points of the circular target trajectory moving at $\omega = 0.45$ rad/s. Various VFF algorithms for λ_k calculation are implemented into the switching MBFGS-DB with $v = 0.5$. Uniform quantization noise of ± 1 pixel is added to the target feature points.

The DAFF and the Alt algorithms generate similar results as seen on Figure 6.52 and Figure 6.53. From Figure 6.54 the calculated λ_k values are similar though the DAFF algorithm yields more variation of λ_k with respect to the image error norm $\|f_k\|$. For the eye-in-hand case the image error of each feature point f_k varies within ± 5 pixels which is greater than expected. This indicates that the eye-in-hand case is more sensitive to noise disturbance as compared to the eye-to-hand cameras, possibly due to the fact that only one camera is used in the eye-in-hand case. However, the

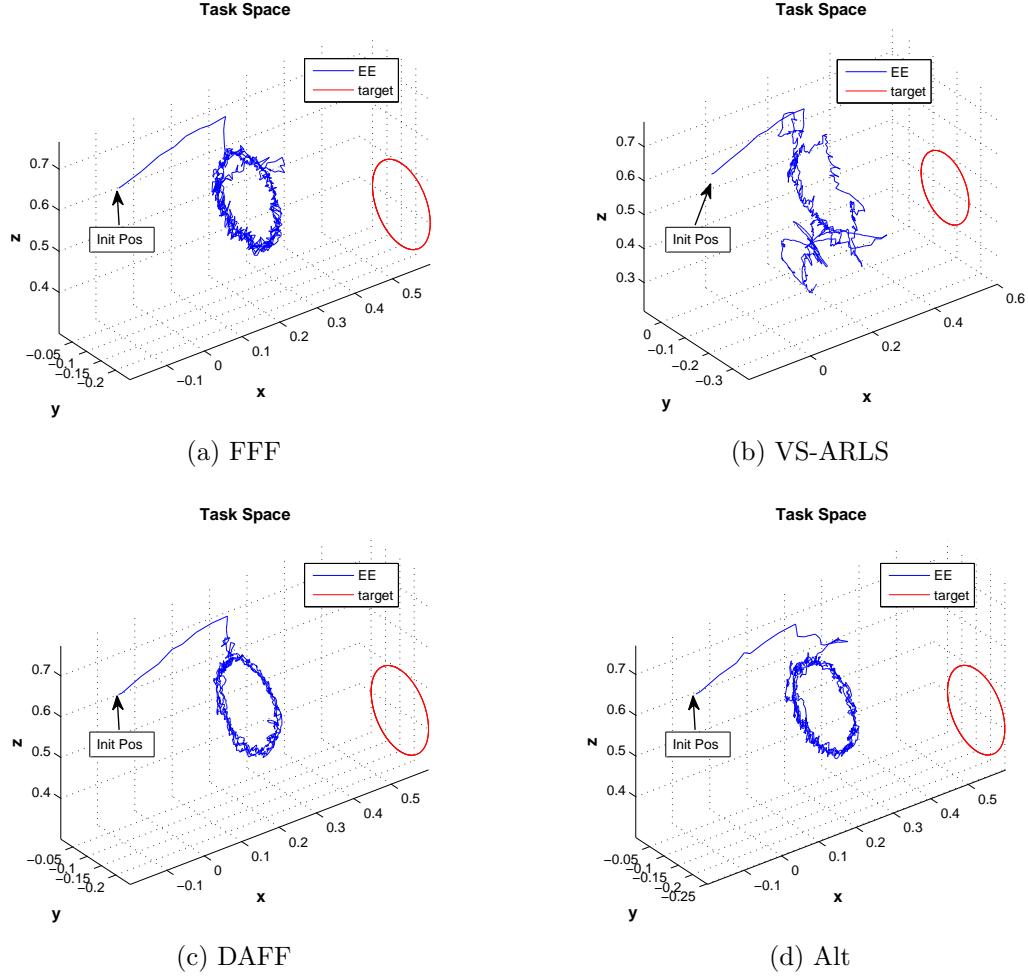


Figure 6.53: The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k calculation and $v = 0.5$. The robot is tracking four feature points of a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. Uniform quantization noise of ± 1 pixel is added to the target feature points.

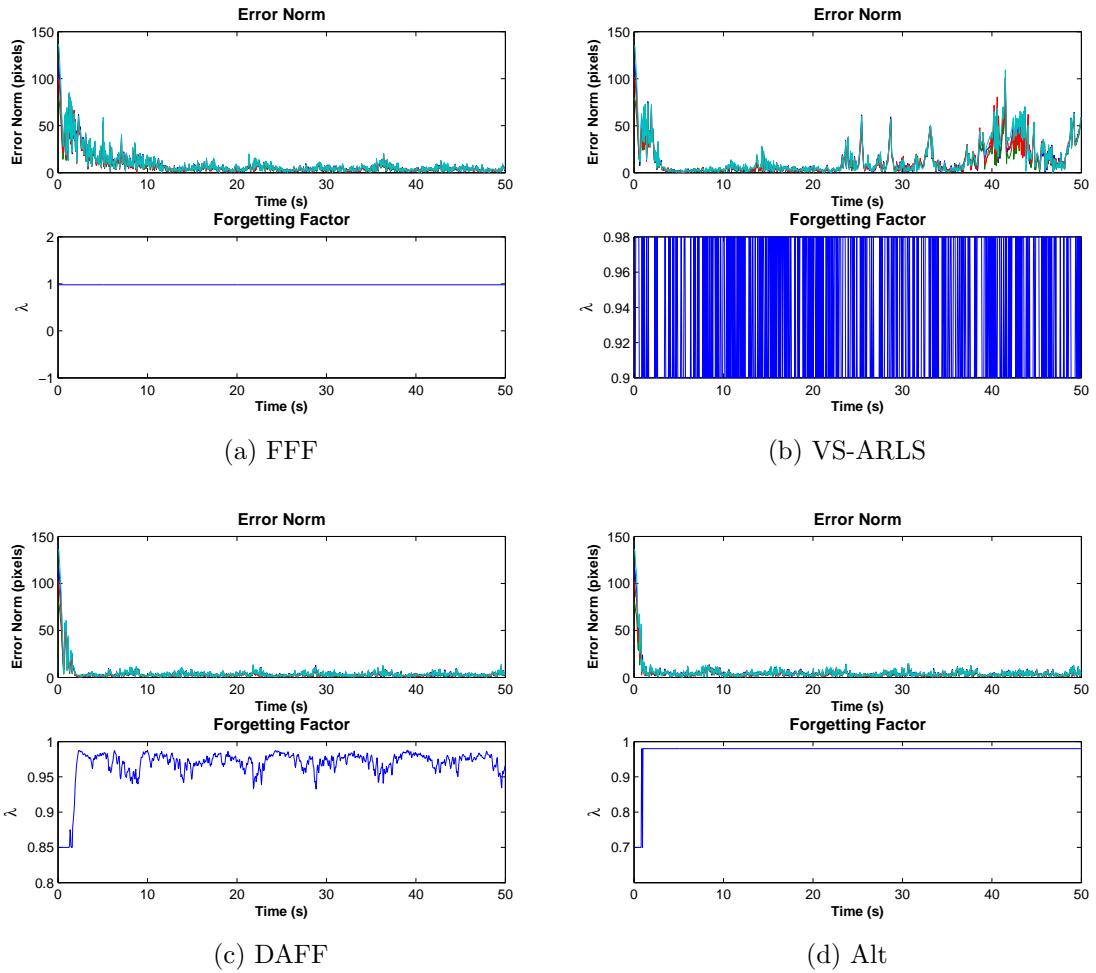


Figure 6.54: The error norm (top) and the forgetting factor λ_k (bottom) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. The robot is tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. Uniform quantization noise of ± 1 pixel is added to the target feature points.

motion of the EE is more accurate in term of maintaining the EE within the desired target plane.

A multiple camera case is also investigated for noise compensation. However, it is not feasible to arrange two eye-in-hand cameras perpendicular to one another. Consequently two nearly-parallel eye-in-hand cameras are used. Although a study of the number of cameras and arrangements on affecting tracking performance are beyond the focus of this research, a brief study of two eye-to-hand cameras is investigated.

Two eye-in-hand cameras

Two eye-in-hand cameras are used with the PUMA 560 robot. Each camera tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s. The cameras are placed ± 150 mm from the origin along the x axis of the EE frame. The optical axes are tilted by $\pm 10^\circ$ about the y axis of the EE frame. The switching MBFGS-DB with various VFF algorithms is tested with uniform quantization noise of ± 1 pixel added to the target feature points. The RMS error and t_s are summarized in Table 6.44 whereas the task space views showing one camera and one target point are shown in Figure 6.55. The switching criterion is $v = 0.5$ for all tests.

The DAFF and the Alt give similar performances that are better than the other algorithms. One interesting observation is that although the RMS value of the VS-ARLS algorithm in Table 6.44 is not much higher than the DAFF or the Alt results, the EE motion does not follow the desired trajectory as seen in Figure 6.55b. Even though the DAFF and the Alt algorithms using two eye-in-hand cameras gives a small RMS value improvement over one eye-in-hand case (Table 6.43), they provide smoother tracking as shown in Figure 6.56.

Square Trajectory

A more difficult square trajectory is used to evaluate each VFF scheme for corner velocity discontinuities that introduce transient behavior.

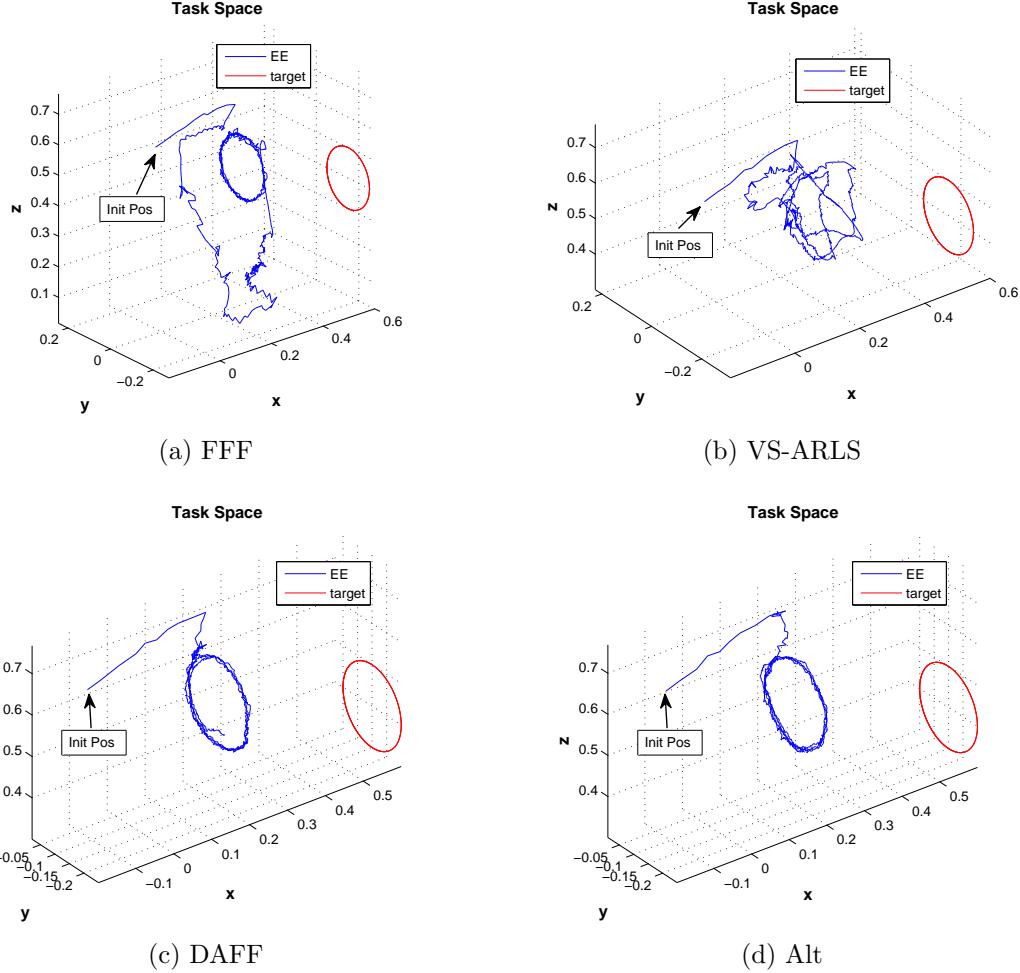


Figure 6.55: The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with two eye-in-hand cameras using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.5$. Each camera is tracking four feature points of a circular target trajectory moving at $\omega = 0.45 \text{ rad/s}$. Uniform quantization noise of ± 1 pixel is added to the target feature points.

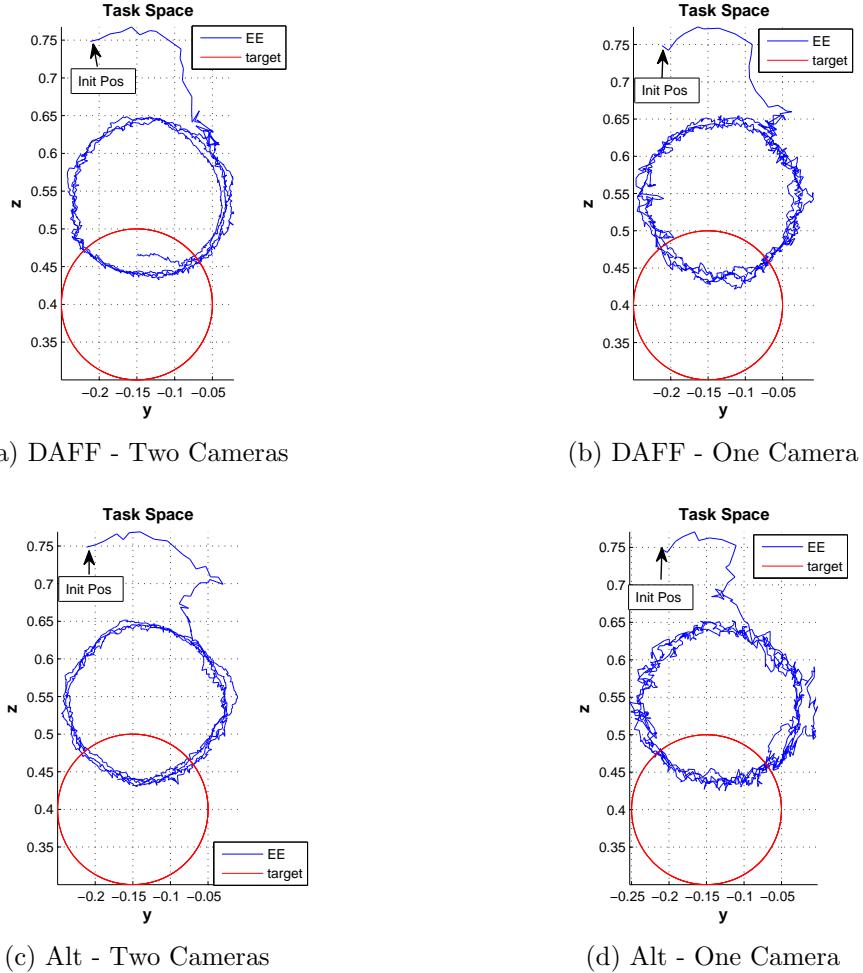


Figure 6.56: The task space in YZ view showing one target point comparison between the one and two eye-in-hand cameras used for the PUMA 560 manipulator using the switching MBFGS-DB with the DAFF and Alt algorithms ($v = 0.5$) with uniform quantization noise of ± 1 pixel added to the target feature points. Each camera is tracking four feature points of a circular target trajectory moving at $\omega = 0.45$ rad/s.

Table 6.44: The RMS error and the settling time comparison for VFF schemes using the PUMA 560 robot with two eye-in-hand cameras, each tracking four feature points of a circular target moving at the angular speed $\omega = 0.45$ rad/s. The switching criterion $v = 0.5$. Uniform quantization noise of ± 1 pixel is added to the target feature points.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
FFF $\lambda_k = 0.98$	5.5013	5.4841	14
VS-ARLS $\lambda_k \in [0.90, 0.98]$ $\eta_1 = 0.05$	4.9379	5.0554	5
DAFF $\lambda_k \in [0.85, 1]$ $\tau = 0.1$	3.0144	2.8989	2
Alt $\lambda_k = 0.70$ or $\lambda_k = 0.98$	3.0777	2.9405	2

The trajectory follows a 200 mm square at 50 mm/s. The target starts from the upper left corner and moves clockwise on the Y-Z plane. One eye-in-hand camera is used.

No noise added

The parameter values in Table 6.35 are used for each VFF algorithm. The task space view showing one target point and the image error norm with a plot of λ_k are shown in Figure 6.57 and Figure 6.58 respectively. The RMS tracking error and the settling time t_s are given in Table 6.45. The switching criterion $v = 0.3$ is used for all tests.

The error norm plots in Figure 6.58 show that the feature points converge to the desired locations and are perturbed as the target turns the corners. Along the sides of

the square trajectory the system exhibits a steady-state convergence. This phenomena is due to the nature of the Broyden estimator that only updates the Jacobian in the “direction” of the trajectory. Therefore, the current Jacobian \hat{J}_k cannot adequately describe the system where a discontinuity occurs until additional path information is received. In this situation, the forgetting factor λ_k should be reduced so that the current Jacobian (and the Hessian) calculation relies more on current information rather than averaging a number of past values. Once steady-state tracking along the edges is recovered, the forgetting factor λ_k is expected to restore its previous value for steady-state tracking. Only the DAFF method yields the expected behavior of λ_k as shown in Figure 6.58d and results in the lowest RMS values of the average transient errors occurring at each corner as shown in Figure 6.57. Furthermore, the DAFF algorithm has an RMS steady-state error (the linear path along the square sides) and t_s similar to the FFF and the Alt algorithms which is better than the VS-ARLS and the GN-VFF-RLS.

The effects of additional system noise

To investigate the effects of system noise, ± 1 mm uniform quantization noise is added to the EE location in addition to ± 1 pixel uniform quantization noise added to the target feature points.

From Table 6.46 the DAFF algorithm gives the best RMS tracking error and the settling time. Though other algorithms yield similar EE trajectories, the settling times are much longer. The DAFF and the Alt algorithms also yield the best repeatability of convergence. The eye-in-hand camera appears sensitive to noise. The average image error f_k of each feature point oscillates within ± 5 pixels even though only ± 1 pixel uniform noise is added to the target features. The oscillation f_k range appears to be approximately the same with uniform quantization ± 1 mm noise added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the

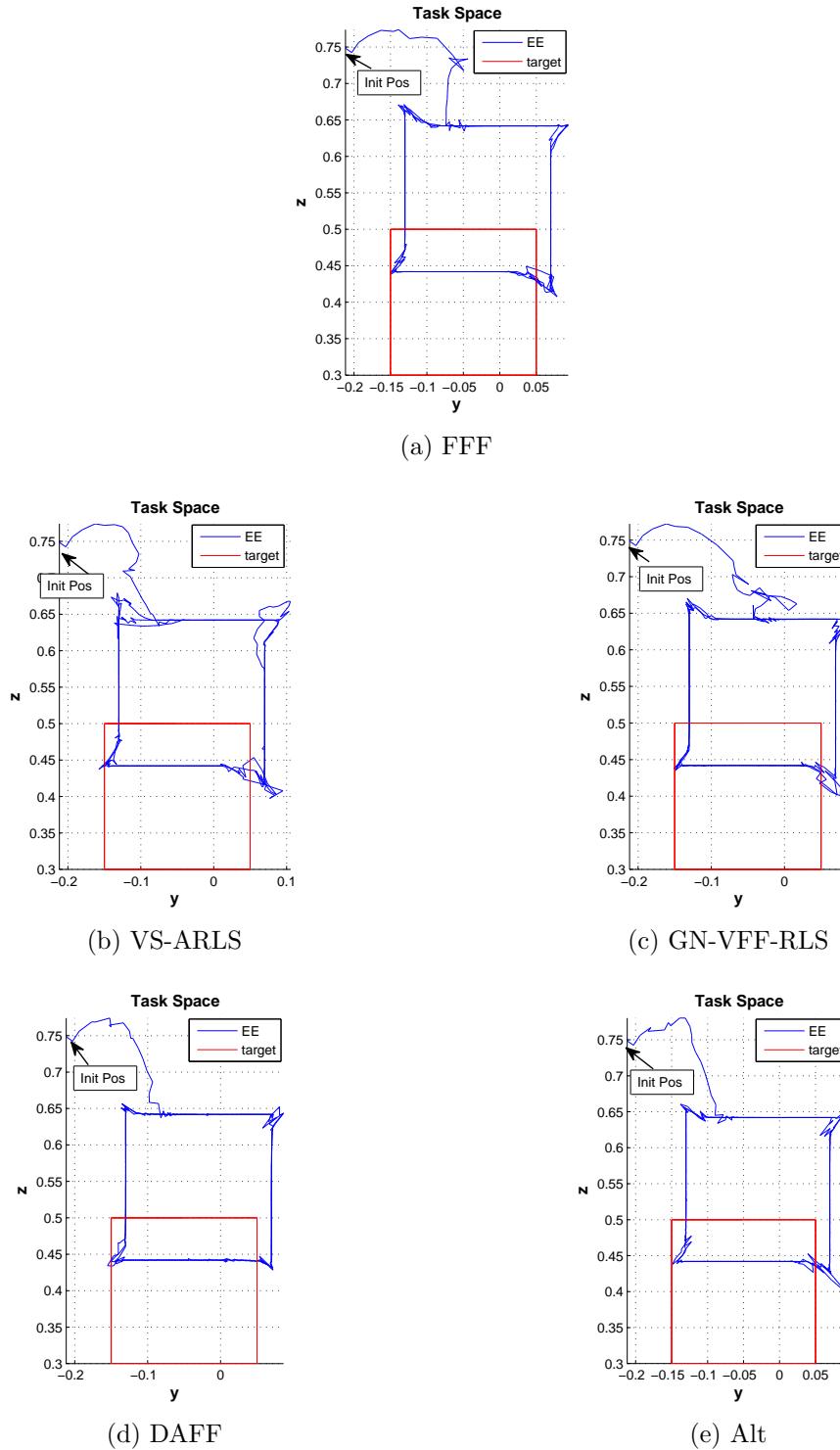
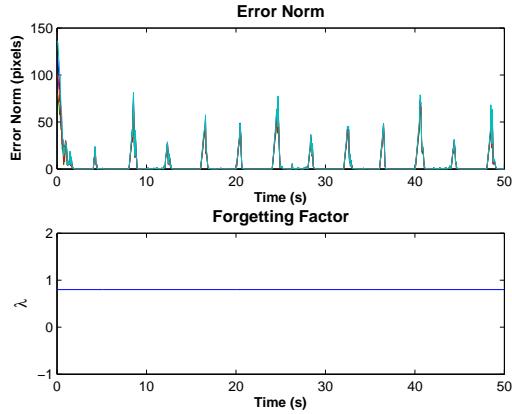
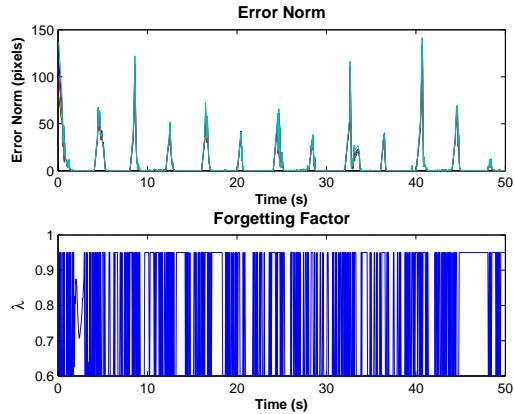


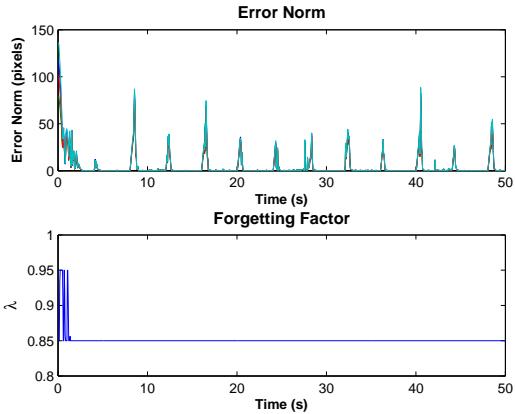
Figure 6.57: The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking four feature points of a square target trajectory moving at a speed 50 mm/s. No additional noise is added.



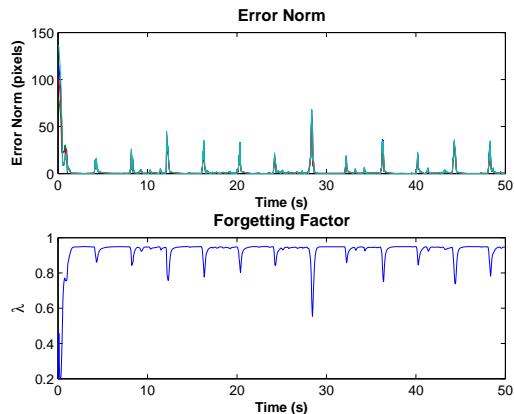
(a) FFF



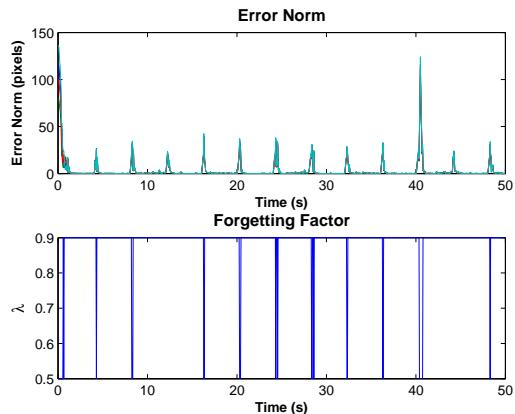
(b) VS-ARLS



(c) GN-VFF-RLS

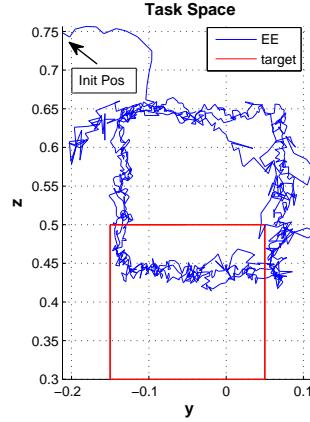


(d) DAFF

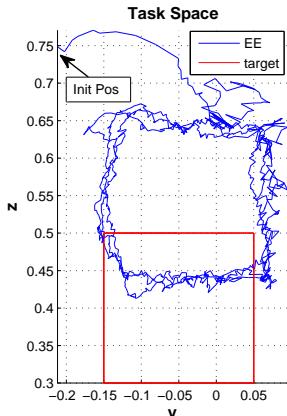


(e) Alt

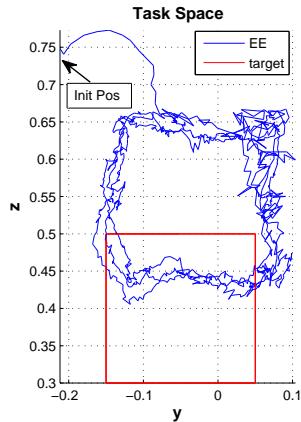
Figure 6.58: The error norm (top) and the forgetting factor λ_k (bottom) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking four feature points of a square target trajectory moving at a speed 50 mm/s. No noise is added.



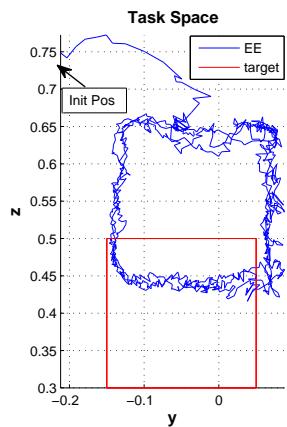
(a) FFF



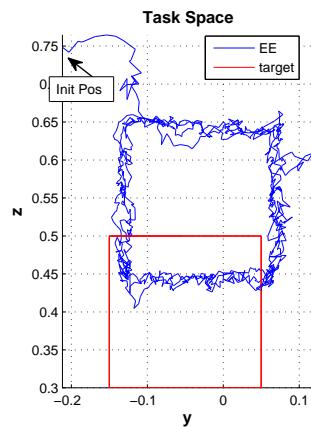
(b) VS-ARLS



(c) GN-VFF-RLS

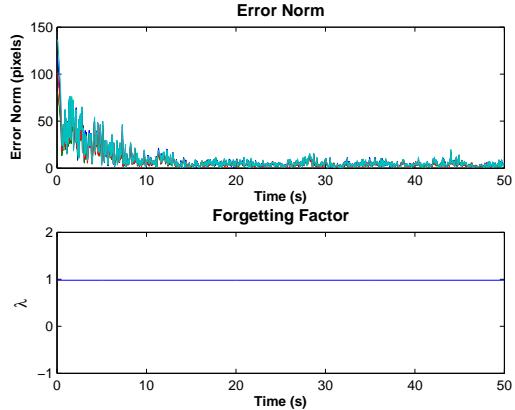


(d) DAFF

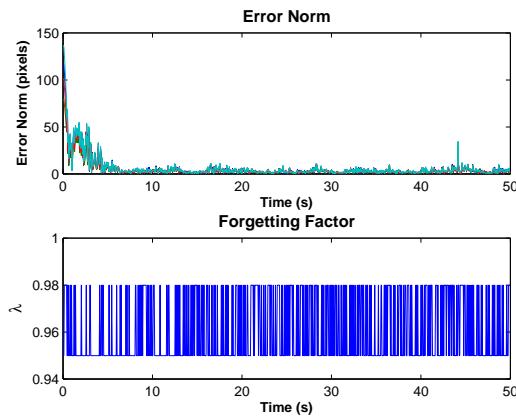


(e) Alt

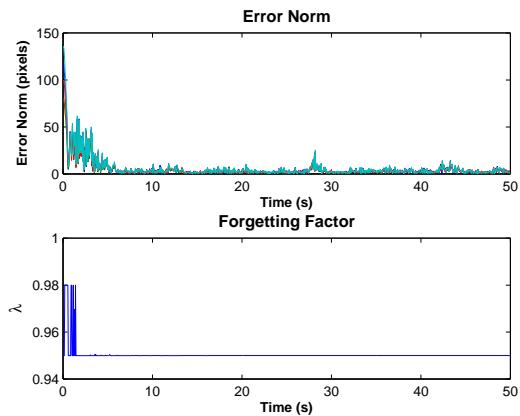
Figure 6.59: The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking four feature points of a square target trajectory moving at a speed 50 mm/s. ± 1 mm uniform quantization noise is added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.



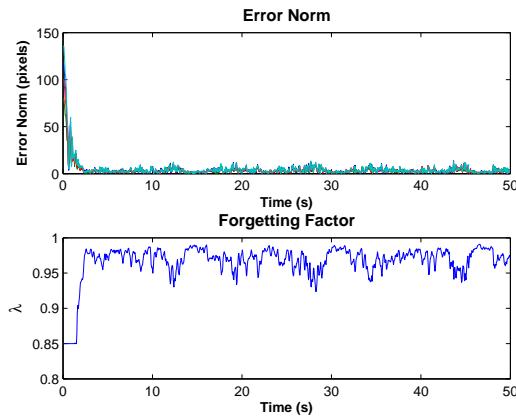
(a) FFF



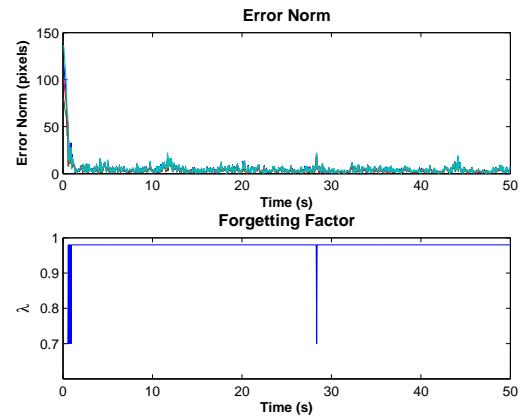
(b) VS-ARLS



(c) GN-VFF-RLS



(d) DAFF



(e) Alt

Figure 6.60: The error norm (top) and the forgetting factor λ_k (bottom) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms for λ_k and $v = 0.3$. The robot is tracking four feature points of a square target trajectory moving at a speed 50 mm/s. ± 1 mm uniform quantization noise is added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.

Table 6.45: The RMS error and the settling time comparison for various VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a square target moving at the speed 50 mm/s. The switching criterion $v = 0.3$. No additional noise is added.

Scheme	RMS Error Steady State (pixels)	RMS Error Transient (pixels)	t_s (sec)
FFF $\lambda_k = 0.8$	0.5409	51.4137	2
VS-ARLS $\lambda_k \in [0.60, 0.95]$ $\eta_1 = 0.05$	3.4937	73.4637	2
GN-VFF-RLS $\lambda_k \in [0.85, 0.95]$ $\eta_2 = 0.01$	1.3779	48.5655	3
DAFF $\lambda_k \in [0.2, 0.95]$ $\tau = 0.1$	0.8075	33.2126	1.5
Alt $\lambda_k = 0.50$ or $\lambda_k = 0.90$	0.5176	43.7836	1.5

target feature points. As a result, the two eye-in-hand camera case is investigated.

Two eye-in-hand cameras

Since the DAFF and Alt algorithms yield the fastest convergence time with similar RMS errors compared to the other algorithms in Table 6.46, only the DAFF and Alt schemes are used for the two eye-in-hand camera case. Uniform quantization ± 1 mm noise is added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points. Table 6.47 summarizes the RMS error and t_s . Figure 6.61 shows the task space view, the average error norm, and the λ_k of the DAFF and the Alt algorithms.

The DAFF algorithm again yields slightly smaller RMS error for both cameras.

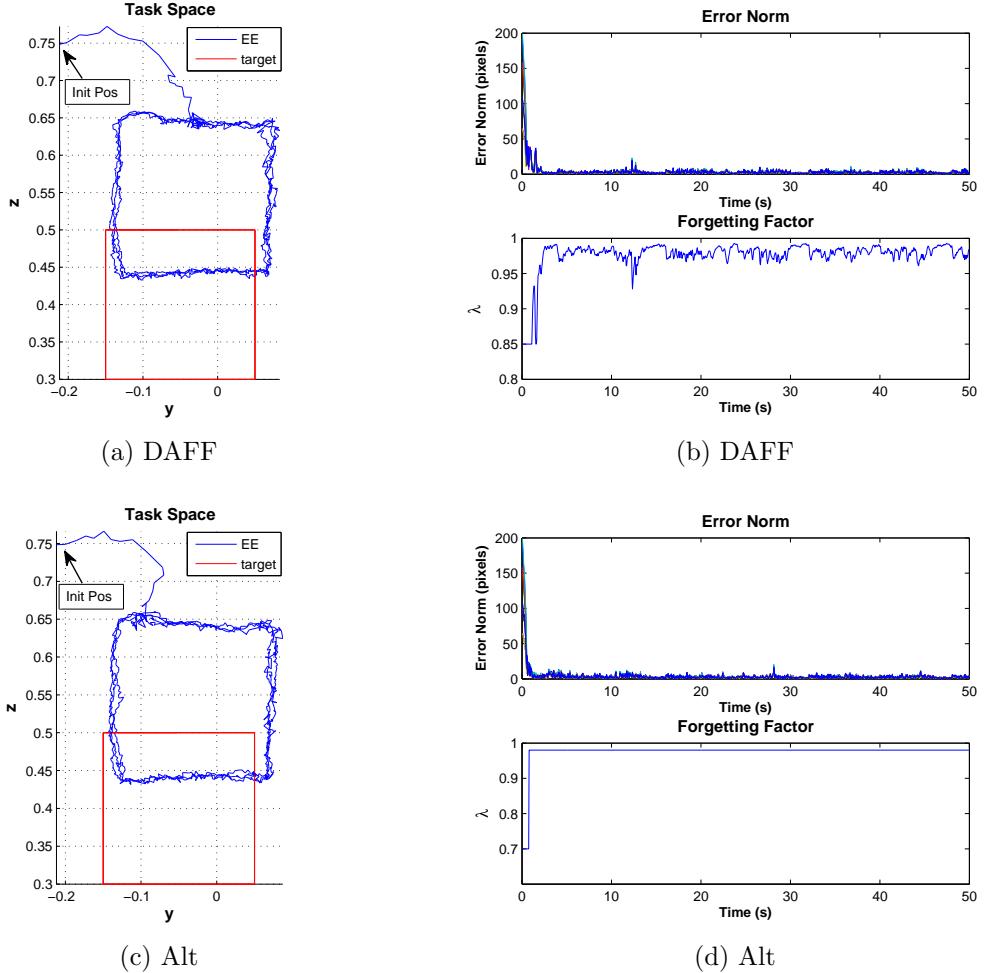


Figure 6.61: The task space view showing one camera and one target point (left column), the error norm and λ_k (right column) of the PUMA 560 manipulator with two eye-in-hand cameras using the switching MBFGS-DB with the DAFF and the Alt algorithms with $v = 0.3$. Each camera tracks four feature points of a square target trajectory moving at a speed 50 mm/s. Uniform quantization ± 1 mm noise is added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.

Table 6.46: The RMS error and the settling time comparison of various VFF schemes using the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a square target moving at the speed 50 mm/s. The switching criterion $v = 0.3$ is used with ± 1 mm uniform quantization noise added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.

Scheme	RMS Error Steady State (pixels)	t_s (sec)
FFF $\lambda_k = 0.98$	5.3445	10
VS-ARLS $\lambda_k \in [0.95, 0.98]$ $\eta_1 = 0.05$	3.6644	5
GN-VFF-RLS $\lambda_k \in [0.95, 0.98]$ $\eta_2 = 0.01$	3.5796	5
DAFF $\lambda_k \in [0.85, 1]$ $\tau = 0.1$	3.7950	2.5
Alt $\lambda_k = 0.70$ or $\lambda_k = 0.98$	4.6922	2.5

The EE motion of two eye-in-hand camera case in Figure 6.60 provides a smoother trajectory as compared to the one eye-in-hand camera case in Figure 6.59.

6.5.3 Comparison between the settling time t_s vs. the cycle time t_{cyc}

In comparison of the settling time t_s and the cycle time t_{cyc} required to complete one cycle of each trajectory, a summary of these quantities are shown in Table 6.48 for different trajectories and target speeds. Since the sampling time $T = 25$ ms is required to achieve tracking a fast cycloidal trajectory in (6.6), to confine the difference to only target speeds $T = 25$ ms is used for all tests. The switching MBFGS-DB with DAFF

Table 6.47: The RMS error and the settling time comparison of the DAFF and the Alt schemes using the PUMA 560 robot with two eye-in-hand cameras, each tracking four feature points of a square target moving at a speed 50 mm/s. The switching criterion is $v = 0.3$. Uniform quantization ± 1 mm noise is added to the EE location in addition to uniform quantization noise of ± 1 pixel added to the target feature points.

Scheme	RMS Error Camera 1 (pixels)	RMS Error Camera 2 (pixels)	t_s (sec)
DAFF $\lambda_k \in [0.85, 1]$ $\tau = 0.1$	3.2880	3.0011	2
Alt $\lambda_k = 0.70$ or $\lambda_k = 0.98$	3.6061	3.2723	2

scheme is used with $v = 0.3$ for all cases except the fast cycloidal trajectory in (6.6) which uses $v = 1.3$ to avoid divergence.

The settling time t_s is nearly constant for a variety of target speeds and trajectories. Though visual guided servoing is a nonlinear system, the robot joint angles θ_k are solved by the Jacobian \hat{J}_k and the Hessian \hat{H}_k that are approximated using an affine model of the system. Thus this system can be somewhat described as a linear dynamic system in which the homogeneous solution is dependent on physical properties of the system, while the particular solution is based on system inputs. Since the homogeneous solution determines the transient behavior the settling time t_s remains nearly constant.

6.5.4 Conclusion

The RRR robot with two eye-to-hand cameras, and the PUMA 560 with one and two eye-in-hand cameras are used to evaluate tracking performance of the switching MBFGS-DB with a variety of VFF methods for adaptively calculating λ_k . For the

Table 6.48: The cycle time (t_{cyc}) and the settling time (t_s) comparison of the switching MBFGS-DB with DAFF method for tracking various trajectories with varying target speed using v selected to ensure convergence.

Trajectory	speed (rad/s)	t_{cyc} (sec)	t_s (sec)
Circle $v = 0.3$	$\omega = 0.01$	628.3185	0.8
	$\omega = 0.05$	125.6637	0.7
	$\omega = 0.15$	41.8879	0.8
	$\omega = 0.45$	13.9626	0.8
	$\omega = 0.90$	6.9813	0.8
	$\omega = 1.35$	4.6542	0.6
	$\omega = 1.80$	3.4907	0.8
Cycloid (slow) $v = 0.3$	$\omega = 1$	6.2832	1
	$\omega = 2$	3.1416	1.3
Cycloid (fast) $v = 1.3$	$\omega = 3$	2.0944	2
Square $v = 0.3$	$v = 10$ (mm/s)	80	0.8
	$v = 50$ (mm/s)	16	0.7
	$v = 100$ (mm/s)	8	1
	$v = 150$ (mm/s)	5.3333	0.6
	$v = 300$ (mm/s)	2.6667	1

RRR robot case, the DAFF algorithm provides the fastest convergence with the smallest RMS error tracking for most cases. However, tracking performance is degraded in the presence of measurement and system noise. The EE motion in Cartesian space significantly deviates out of the desired target plane though the RMS errors on the image space are not substantial. This problem can be improved if the cameras are rearranged so that both X-Y and Y-Z tracking planes can be seen by each camera.

The DAFF and Alt algorithms yield similar results that outperform other VFF algorithms for the PUMA 560 robot with an eye-in-hand camera tracking four feature points of a circular trajectory. DAFF offers slightly lower RMS errors and settling

times, especially for a faster angular speed and in the presence of added noise. However, the eye-in-hand PUMA 560 is more sensitive to noise than the RRR robot with the two eye-to-hand cameras. A brief study using two eye-in-hand cameras shows a potential for improving noise compensation with smoother EE motion.

For the square trajectory, the DAFF algorithm also offers better RMS errors and settling times as compared to other VFF algorithms. In addition, it provides a smaller RMS transient error due to velocity discontinuities at the corners. In the presence of measurement and system noise, the DAFF algorithm yields a better RMS tracking error as compared to the Alt method. The two eye-in-hand cameras also help improve the RMS error while generating a smoother EE motion in the task space.

6.6 Performance Evaluation of the Switching MBFGS-DB with various VFF algorithm and with/without LMA

Even though the LMA shows insignificant improvement over the switching MBFGS-DB without the LMA in Section 6.4, those cases only use a fixed forgetting factor without noise. The DAFF and the Alt algorithms are used with the switching MBFGS-DB and adaptive λ_k schemes to test the LMA in the presence of noise. Since the DGN-PBM algorithm uses the FFF originally, the FFF is also used with the MBFGS-DB to evaluate the LMA.

In this section the same PUMA 560 robot with an eye-in-hand camera is used for tracking the cycloidal trajectory in (6.5) with uniform quantization noise of ± 1 pixel added to the target feature points. The starting robot joint angles are $\theta_0 = [15.73^\circ, 132.5^\circ, -135.6^\circ, -4.27^\circ, -108.75^\circ, 14.27^\circ]^T$ for all tests. Due to the trajectory complexity, using a 6 degrees-of-freedom robot, and using one eye-in-hand camera, this is the most difficult case, especially in the presence of the measurement noise. A sampling time $T = 15$ ms is used to facilitate convergence. The switching criterion $v = 0.3$, the range of $\lambda_k \in [0.85, 1]$, and $\tau = 0.1$ are used for the switching MBFGS-DB with the DAFF algorithm whereas λ_k is alternated between $\lambda_{low} = 0.7$ and

$\lambda_{high} = 0.98$ for the Alt algorithm and $\lambda = 0.98$ is applied for the FFF algorithm.

For convenience the switching MBFGS-DB with the DAFF, the FFF, or the Alt algorithms are referred as the MBFGS-DB-DAFF, MBFGS-DB-FFF, and MBFGS-DB-Alt algorithms respectively. To validate the feasibility of the switching MBFGS, the DGN-PBM is also implemented with the DAFF, FFF, and Alt to calculate λ_k .

Table 6.49: The RMS error and the settling time comparison of the switching MBFGS-DB algorithm with/without LMA and a variety of the VFF algorithms. The PUMA 560 robot with an eye-in-hand camera is used for tracking four feature points of a cycloidal target. The switching criterion $v = 0.3$ is used with uniform quantization noise of ± 1 pixel added to the target feature points.

VFF Scheme	LMA	RMS Error (pixels)	t_s (sec)
FFF $\lambda_k = 0.98$	Yes	5.2830	3
	No	4.2146	3
DAFF $\lambda_k \in [0.85, 1]$ $\tau = 0.1$	Yes	2.9022	1
	No	2.5548	1.2
Alt $\lambda_k = 0.70$ or $\lambda_k = 0.98$	Yes	3.3687	1
	No	5.2643	3

Table 6.49 and Table 6.50 show summaries of the RMS tracking error and the settling t_s for the switching MBFGS-DB and the DGN-PBM with various VFF algorithms and with/without the LMA respectively. For the switching MBFGS-DB algorithm the DAFF without the LMA gives the best RMS tracking error and the settling time as shown in Table 6.49. Although the EE trajectories in the camera space are similar for all algorithms in which the EE initially moves away from the desired target, the EE motion of the DAFF algorithm with/without LMA gives a smaller deviation from the desired target points as seen in Figure 6.62. The EE motion in the task space of the DAFF with/without LMA is similar to the Alt with/without

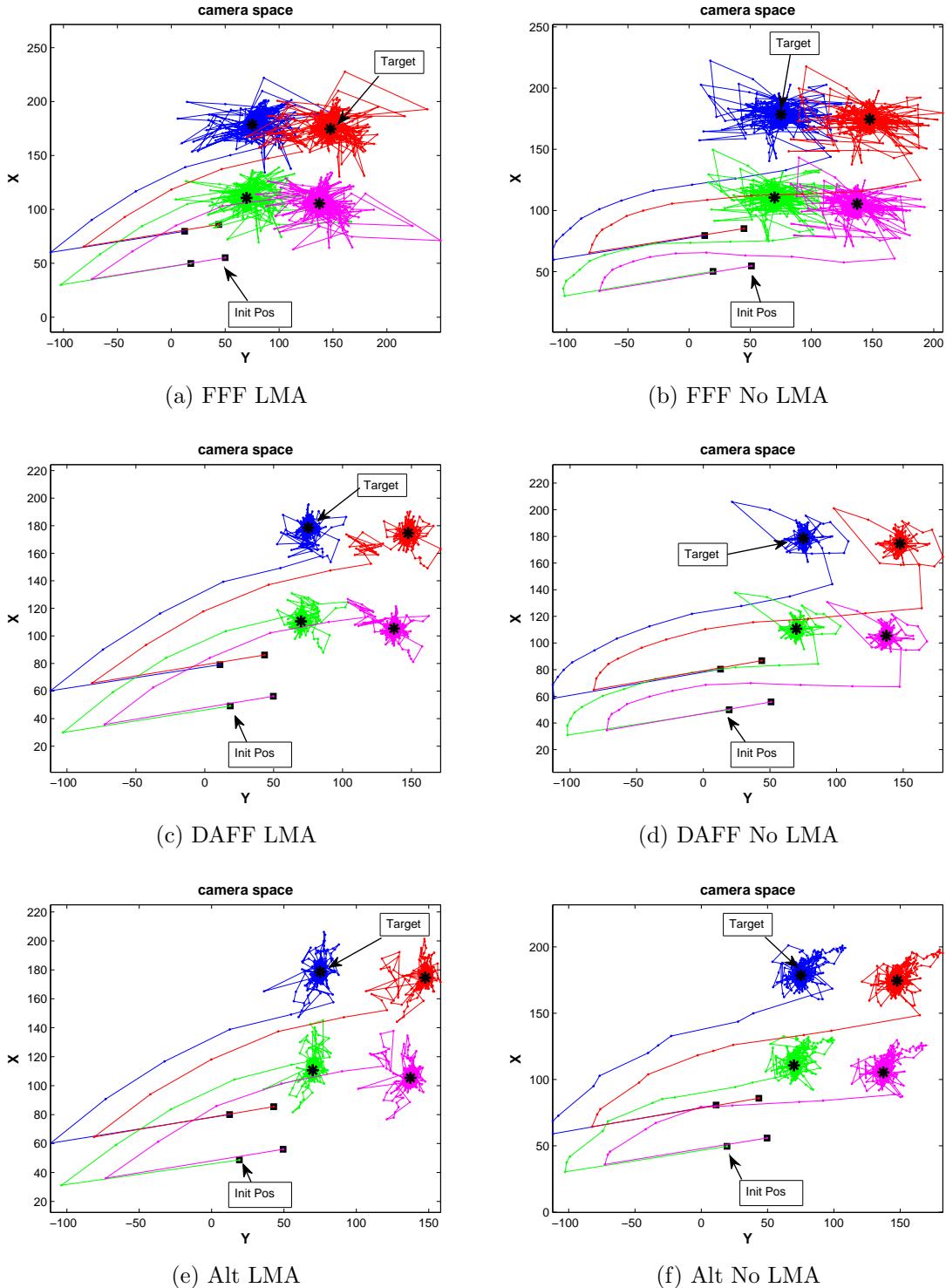


Figure 6.62: The camera space of the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.

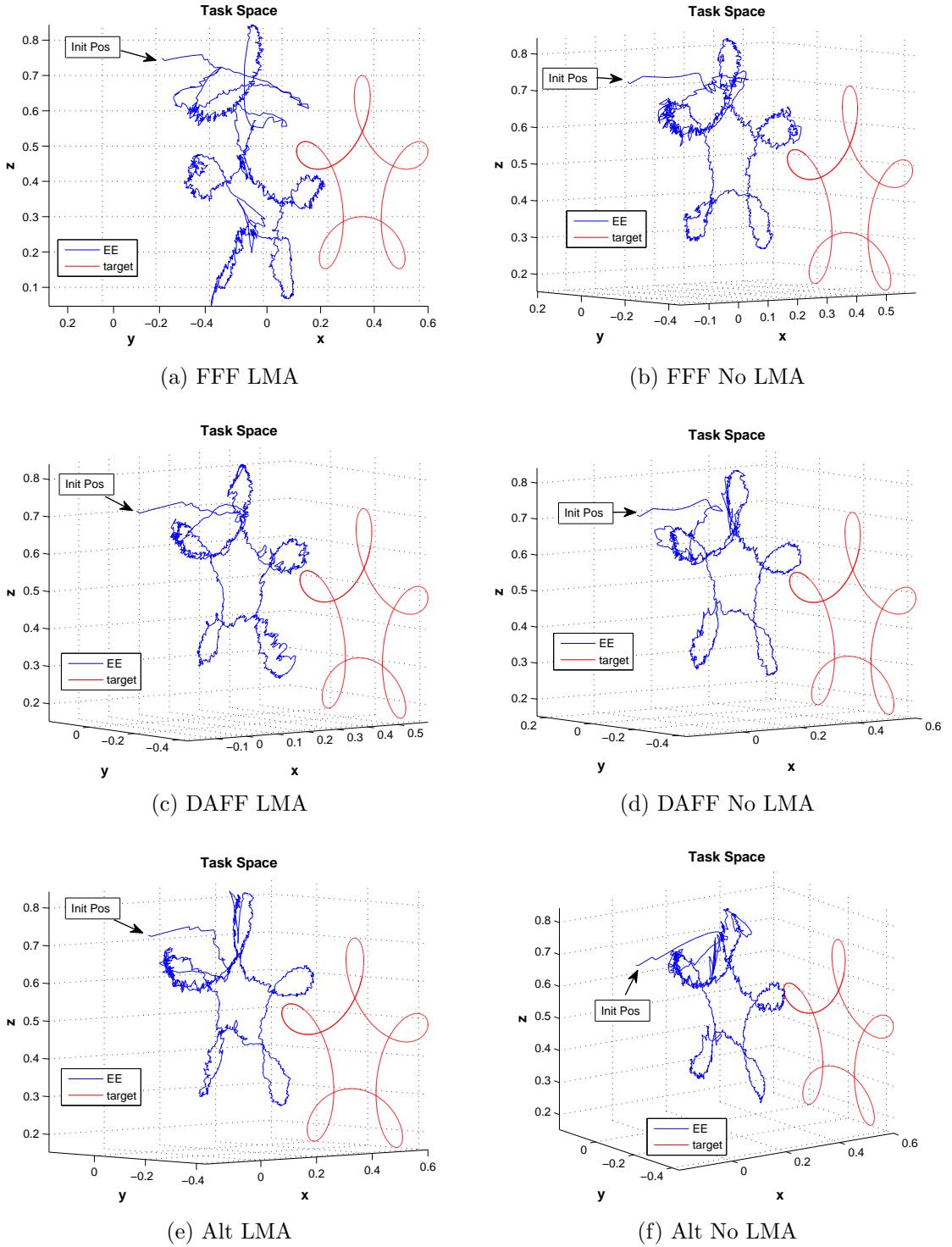


Figure 6.63: The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB with various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.

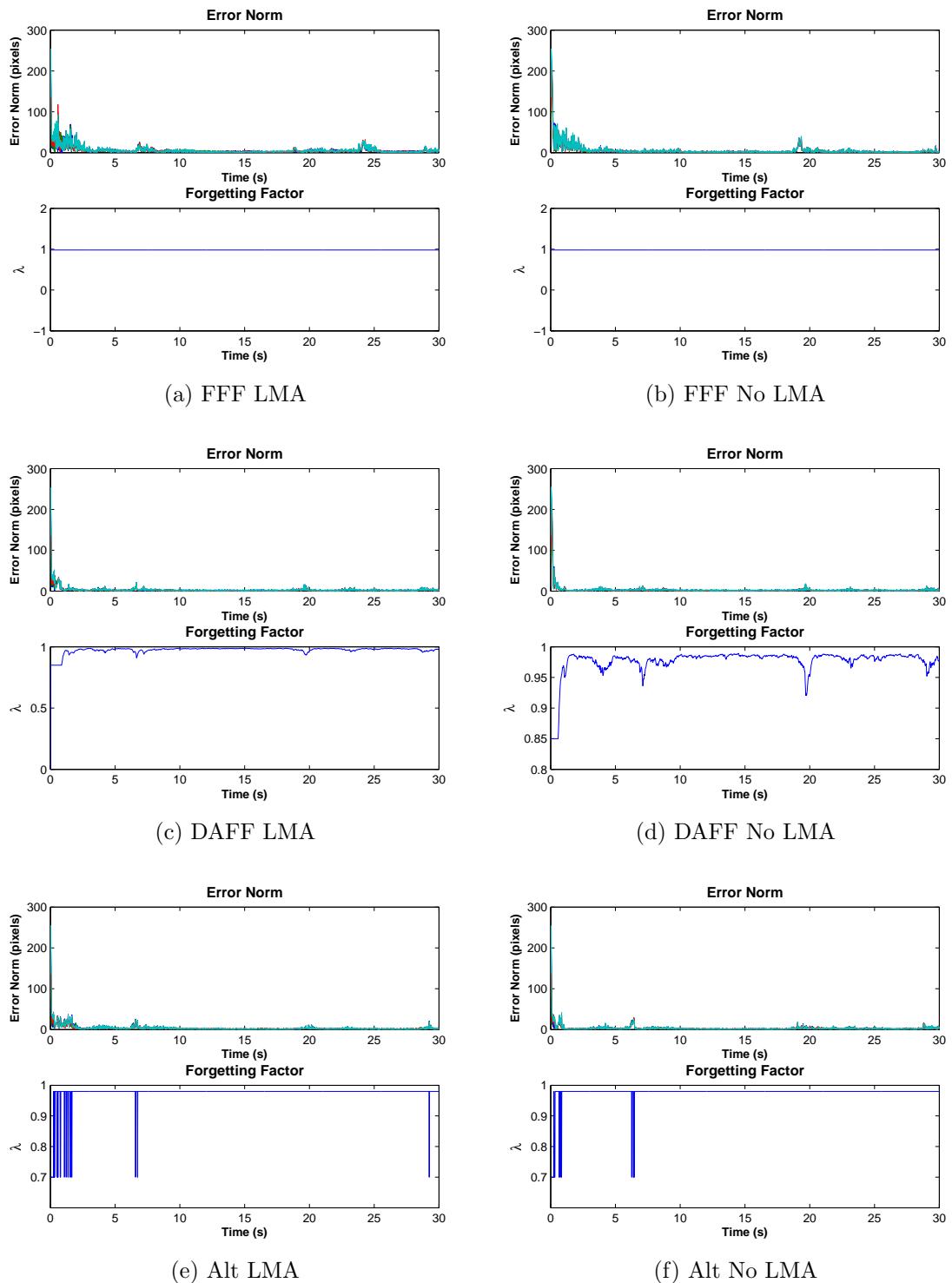


Figure 6.64: The error norm and λ_k plots of the PUMA 560 manipulator with an eye-in-hand camera using the switching MBFGS-DB for various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal target trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.

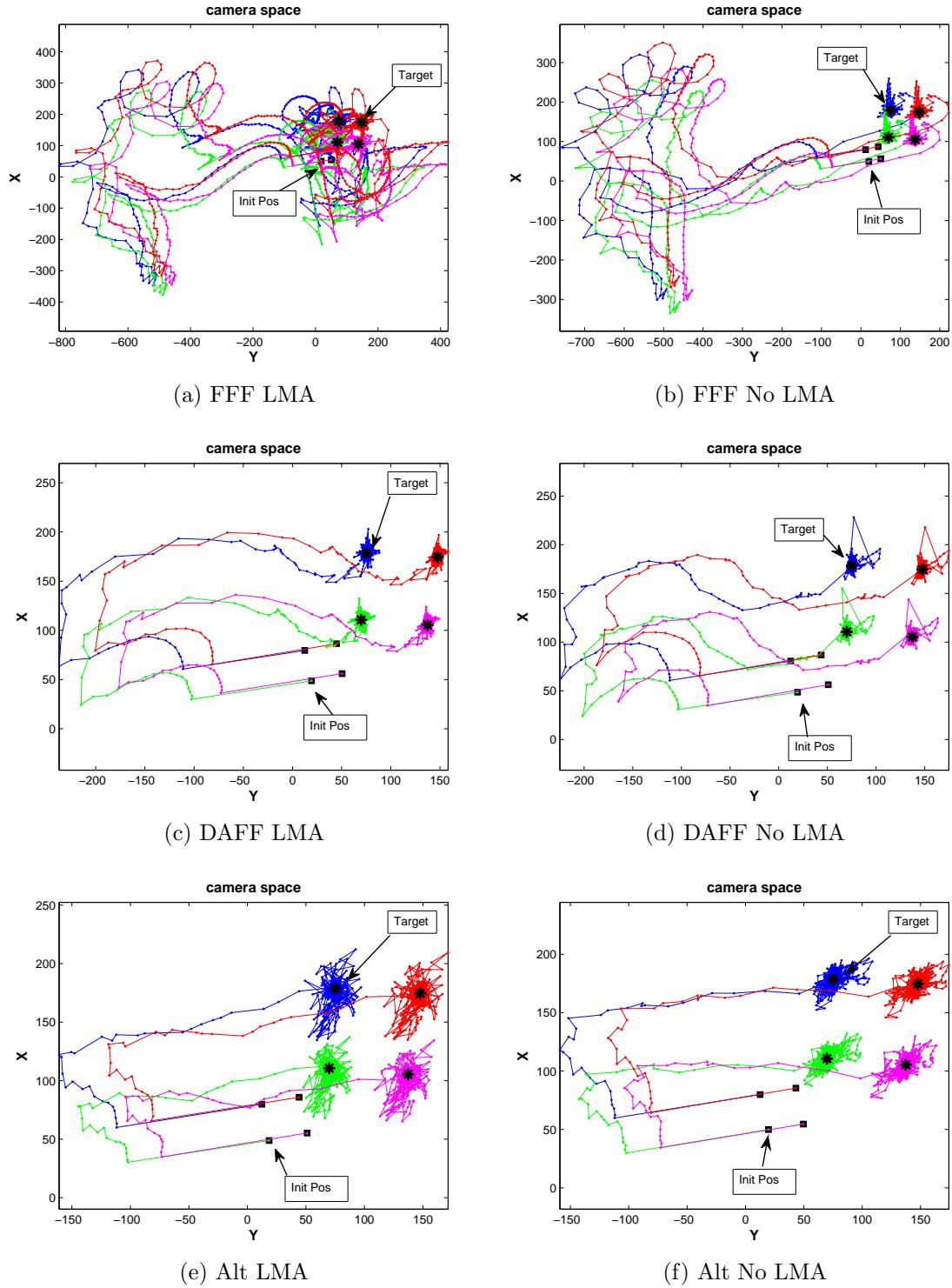


Figure 6.65: The camera space of the PUMA 560 manipulator with the eye-in-hand camera using the DGN-PBM for various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.

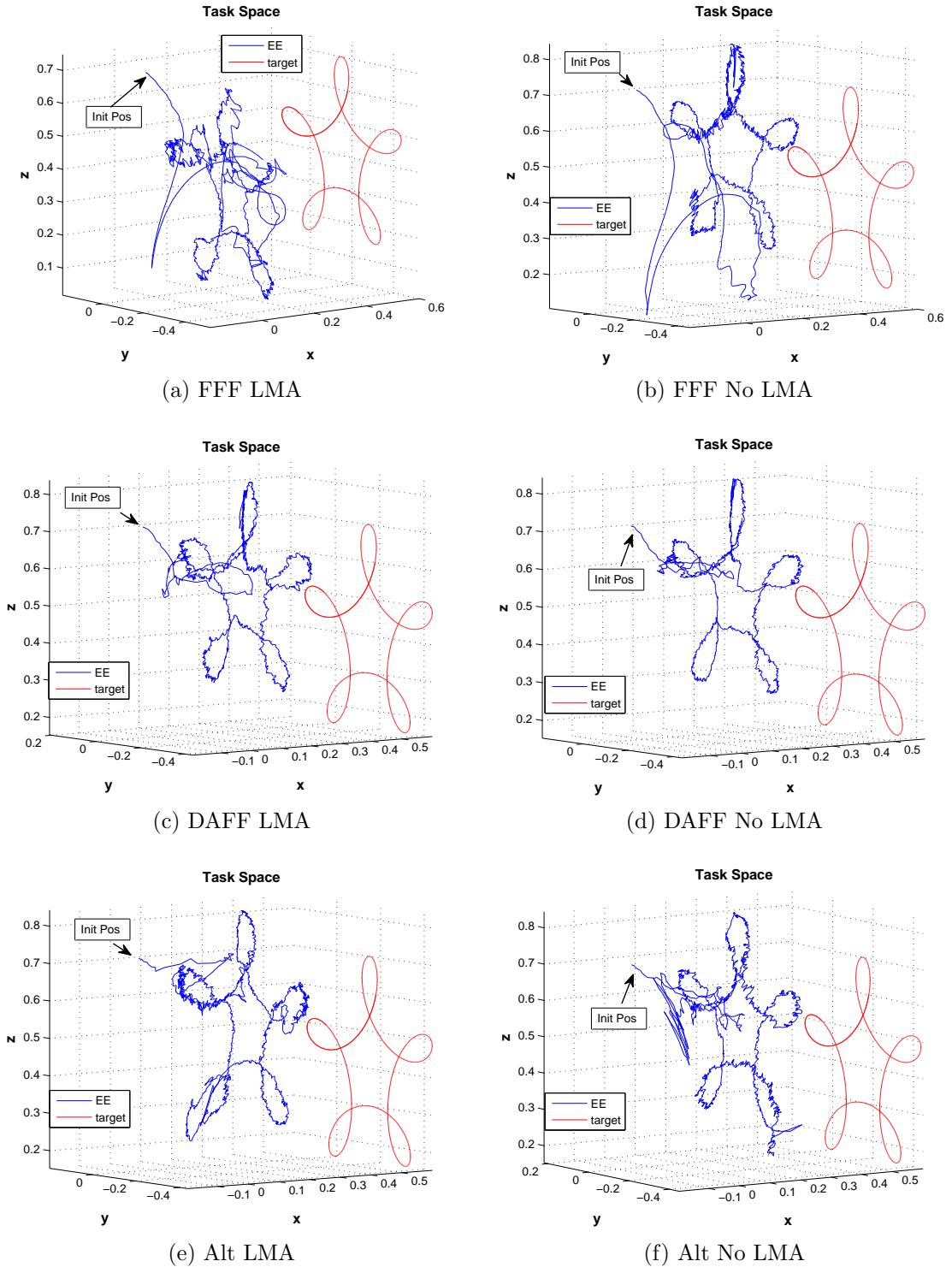


Figure 6.66: The task space view showing one target point for clarity (the others are similar) for the PUMA 560 manipulator with an eye-in-hand camera using the DGN-PBM for various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.

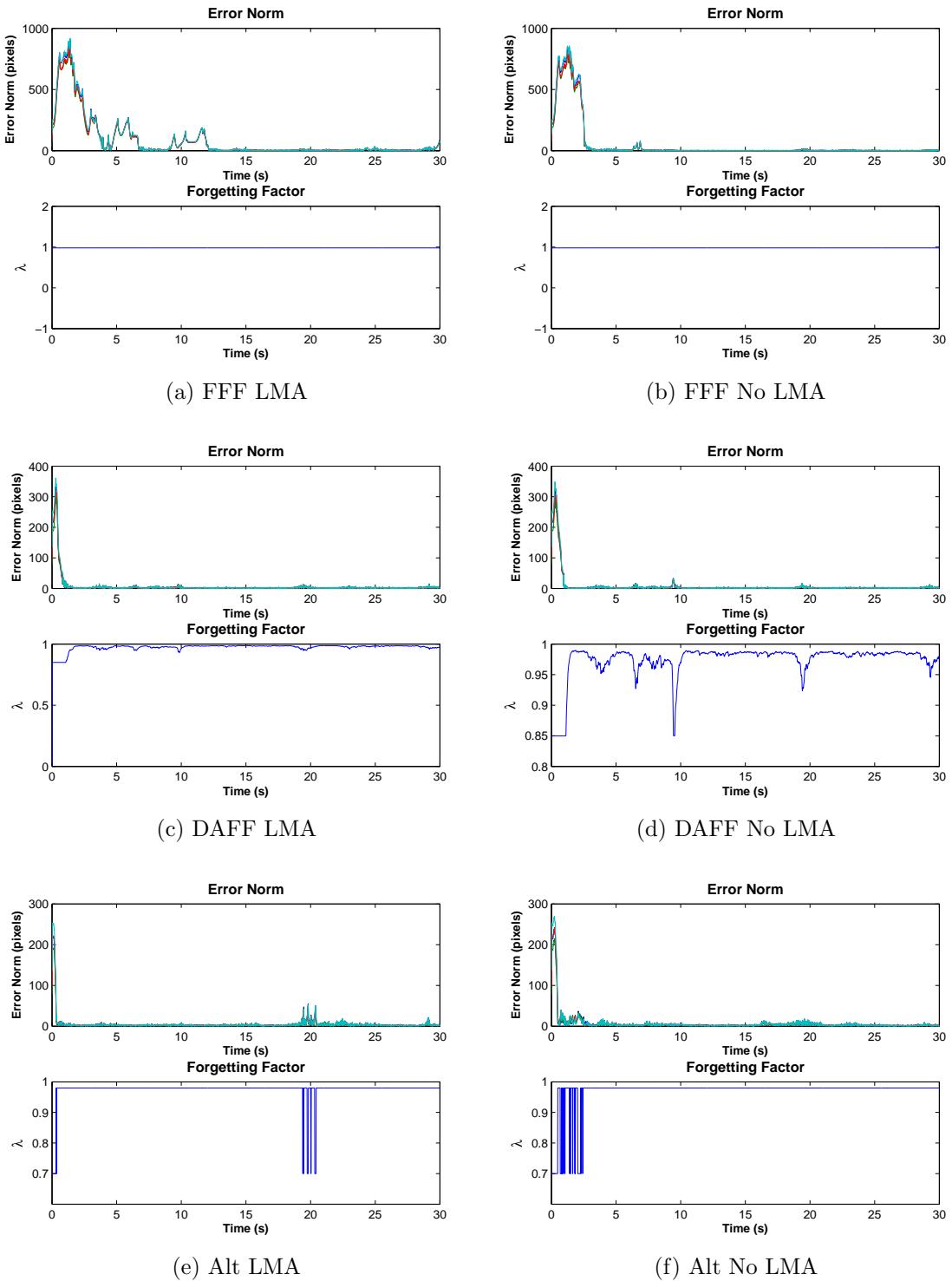


Figure 6.67: The error norm and λ_k plots of the PUMA 560 manipulator with an eye-in-hand camera using the DGN-PBM for various VFF algorithms implemented with the LMA (left column) and without the LMA (right column). The robot is tracking four feature points of a cycloidal target trajectory. Uniform quantization noise of ± 1 pixel is added to the target feature points.

Table 6.50: The RMS error and the settling time comparison of the DGN-PBM algorithm with/without LMA and a variety of the VFF algorithms. The PUMA 560 robot with an eye-in-hand camera is used for tracking four feature points of a cycloidal target. The switching criterion $v = 0.3$ is used with uniform quantization noise of ± 1 pixel added to the target feature points.

VFF Scheme	LMA	RMS Error (pixels)	t_s (sec)
FFF $\lambda_k = 0.98$	Yes	9.6868	12
	No	4.0848	3
DAFF $\lambda_k \in [0.85, 1]$ $\tau = 0.1$	Yes	2.6756	1.5
	No	3.1855	1.2
Alt $\lambda_k = 0.70$ or $\lambda_k = 0.98$	Yes	4.9357	1
	No	3.8042	3

LMA. However, Figure 6.63 shows that the DAFF without LMA yields the most direct path from the initial robot position to reach the desired trajectory and is best able to follow the target. The λ_k calculated from the DAFF algorithm is adaptively changed with respect to the error norm $\|f_k\|$ but the DAFF without LMA is more sensitive to $\|f_k\|$ compared to the LMA as shown in Figure 6.64.

For the DGN-PBM algorithm with various VFF algorithms the EE trajectories as seen in the camera space move significantly away from the target, shown in Figure 6.65, compared to the switching MBFGS-DB algorithm. The DGN-PBM with FFF algorithm with/without LMA yields an undesirable motion of the EE moving from its initial position to the desired target. Though the Alt without LMA gives better tracking on the Y-Z plane, its motion from the initial robot configuration is not as direct as the DAFF with/without LMA as seen on Figure 6.66. Figure 6.67 shows the average error norm and the λ_k plot of various VFF algorithms with/without LMA that are similar to the switching MBFGS-DB algorithm. The DAFF without LMA

yields λ_k that is more sensitive the image error norm $\|f_k\|$ than with the LMA.

6.6.1 Conclusion

For all VFF algorithms, both the MBFGS-DB and the DGN-PBM without LMA offers better EE trajectory than the LMA. The DAFF algorithm offers the best performance for both the switch MBFGS-DB and the DGN-PBM algorithms and the LMA has little effect on the performance of DAFF. Overall, the LMA algorithm only marginally improves tracking performance of a complex trajectory in the presence of noise.

To improve noise compensation more cameras can be added similar to Section 6.5.2.2. An alternative is to combine an eye-to-hand with an eye-in-hand camera, which is beyond scope of this study.

6.7 Summary

This chapter simulates the proposed switching algorithms with VFF schemes for large residual tracking. They are also studied with and without implementation of the LMA for avoiding ill-conditioning of the Hessian approximation. A summary of the algorithms proposed to improve on the dynamic quasi-Gauss Newton algorithms is presented in Table 6.1.

The objective is to establish a basic understanding of the proposed algorithms to improve tracking performance. They are compared with the original DBM-RLS and the DGN-PBM algorithms using a fixed forgetting factor. Although only a representative number of simulations are tested, they reveal the potential of the presented switching schemes with VFF to improving tracking performance with a variety of manipulator DOF, camera configurations, and trajectories.

Section 6.1 presents overviews of the organization of this chapter. A summary of the robot systems, the camera systems, and assumptions used for simulations are reviewed in Section 6.2.

In Section 6.3 a 3 DOF and a 6 DOF robot with a variety target trajectories

are used for large residual tracking. The proposed residual \hat{S}_k approximations are reviewed in Figure 6.4. In this section the effect of noise is deferred and a fixed forgetting factor λ is used.

For RRR robot with two eye-to-hand cameras tracking a circular trajectory target, the switching DFN-BFGS-DB and the MBFGS-DB algorithms are the most effective methods. The results are further improved if the NP-Jacobian approximation is utilized. A heuristic selection of the switching criterion v demonstrates better RMS tracking error and settling time as compared to other existing algorithms (Scheme 1 and 2).

Then a Puma 560 manipulator with an eye-in-hand camera tracking is used with a variety of target trajectories. For a circular trajectory with the fastest angular speed the switching DFN-BFGS-DB and MBFGS-DB outperform the other switching algorithms. However, the switching MBFGS-DB algorithm yields the smallest RMS error with better stability. For the helical trajectory, the switching MBFGS-DB algorithm provides the best result for the fastest speed.

In Section 6.4 the switching algorithms are implemented with the LMA but with a fixed λ and no noise. First, the RRR robot with two eye-to-hand cameras tracking a circular trajectory are used for simulations. Generally, without LMA yields faster convergence but higher RMS errors for all switching algorithms. For various starting robot locations, including LMA only shows slightly improvement.

Second, for the PUMA 560 robot using an eye-in-hand camera tracking a circular trajectory all switching algorithms with LMA yield slightly faster convergence with a trade-off of slightly increased RMS errors. For different robot starting positions and trajectories, the LMA only marginally improves on convergence and tracking stability for a more complex cycloidal trajectory.

In Section 6.5 different VFF algorithms implemented with the switching algorithms are investigated.

For the RRR robot with a circular trajectory in the presence of noise, the switching MBFGS with the DAFF algorithm yields the fastest convergence and RMS tracking errors. However, the EE motion substantially deviates from the target plane for all algorithms. Alternatively, a perpendicular camera arrangement discussed in Section 6.5 significantly minimizes the out of plane the EE motions.

For the PUMA 560 robot with a circular trajectory the DAFF offers slightly better RMS error and t_s , especially for a faster angular speed and in the presence of noise. For a square trajectory, the DAFF algorithm gives the best RMS steady-state and RMS transient tracking errors as the target turns the corners resulted in the velocity discontinuities. In the presence of noise, the DAFF algorithm consistently offers the best RMS tracking error. A brief study of using two eye-in-hand cameras shows a potential for slightly improving RMS errors and smoother EE trajectories in the presence of measurement and system noise.

In Section 6.6 presents the effect of the LMA implemented with the switching MBFGS and the DGN-PBM for various VFF algorithms using the PUMA 560 robot tracking a complex cycloidal trajectory. In the presence of noise, the DAFF algorithm gives the most desirable performance when implemented with either the switching MBFGS-DB or the DGN-PBM algorithm in regardless of LMA inclusion. For all cases, the switching MBFGS-DB-DAFF without the LMA generates the smallest RMS error and the fastest convergence. The LMA algorithm only marginally improves tracking performance of a complex trajectory with the presence of noise. A similar two eye-in-hand cameras may be used for noise compensation.

6.7.1 Conclusion

Considering the various robot degrees-of-freedom, camera configurations, trajectories, and target speeds the switching MBFGS-DB algorithm with the DAFF method for adaptively calculating λ_k most consistently outperforms the other proposed switching

and VFF algorithms. As compared to the original DGN-PBM algorithm with a fixed forgetting factor, the switching MBFGS-DAFF algorithm improves tracking performance for large residual problems while the DAFF algorithm significantly offers better tracking accuracy with fast convergence, especially in the presence of noise. Although the LMA seems to marginally improve tracking performance in the presence of noise, this is only a limited case being studied and a more thoroughly investigation should be pursued. Multiple cameras and camera arrangements significantly improve tracking performance, especially in the presence of noise. These simulation results indicate the effectiveness of the switching MBFGS-DB algorithm with the DAFF scheme to improve tracking performance for large residual problems in the presence of noise.

CHAPTER VII

CONCLUDING REMARKS

This study investigates visual servoing with adaptive forgetting factor algorithms for large residual problems. Various novel residual approximations are introduced, namely the *dynamic BFGS (DBFGS)*, the *modified BFGS (MBFGS)*, and the *dynamic full Newton method with BFGS (DFN-BFGS)* algorithms. Since residual approximation is utilized only when a large error occurs, or a switching algorithm is introduced that is a combination of the (full) Newton method and the dynamic quasi-Gauss-Newton algorithm in [57, 59]. Unlike other approaches in [17, 49], the algorithm includes or excludes the residual approximation based on a heuristically selected switching criterion v .

The switching algorithm performance is dependent on a proper selection of a forgetting factor λ_k . Consequently, a novel adaptive forgetting factor called the *Dynamic Adaptive Forgetting Factor (DAFF)* is developed which is a heuristic approach to approximate λ_k with respect to the image error norm $\|f_k\|$. Compared to a number of existing Variable Forgetting Factor (VFF) algorithms from adaptive filtering area, the DAFF method is shown to consistently provide the best RMS tracking errors and convergence times in the presence of noise.

For a variety of robot DOF, camera systems, and trajectories investigated in simulation, the switching MBFGS-DB with DAFF algorithm is shown to offer superior tracking performance for large residual tracking problems, especially in the presence of noise.

7.1 Contributions

This study built on the work by Piepmeyer et al. [57, 59], Fu et al. [25], and Hao et al. [29]. Piepmeyer et al. introduced dynamic quasi-Gauss-Newton algorithms with two implementations, with or without partitioning for Broyden’s method (the DBM-RLS and the DGN-PBM algorithms respectively), that are shown to provide stable and convergent tracking in uncalibrated visual servoing. However, they are limited to only the zero- or small-residual cases where the Gauss-Newton method is applicable. Consequently, Fu et al. introduce a residual approximation integrated with the DGN-PBM algorithm with a trust region and the LMA to track a static target for large residual problems. This work pointed the way to the development of the MBFGS and DBFGS residual approximations. Hao et al. propose an algorithm to adaptively calculate λ_k to be integrated with the DGN-PBM algorithm. Simulation results show a potential to improve tracking performance motivating the DAFF method development. While the proposed algorithm relates to the aforementioned work, the switching MBFGS-DB with DAFF algorithm is distinct from that work due to the following attributes:

- The switching MBFGS-DB with DAFF algorithm provides stable uncalibrated visual servoing tracking for large residual problems with a moving target that had not been previously addressed.
- The DAFF algorithm effectively adapts λ_k with respect to the image error norm $\|f_k\|$ and results in significant RMS error improvement in the presence of noise for large residual problems.

In addition this thesis make the specific contributions that differentiate it from the previous work:

1. A theoretical development of the DBFGS and the MBFGS algorithms to recursively approximate residual \hat{S}_k term.

- The implementation of the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm to approximate the residual \hat{S}_k with a dynamic quasi-Gauss-Newton method resulted in the DFN-BFGS algorithm for large residual problems.
 - The DBFGS and the MBFGS methods for approximating the residual \hat{S}_k are derived.
 - Convergence analysis of the MBFGS in analogy to the BFGS algorithm in [8] is presented.
2. A fundamental basis for applying a switching algorithm for large residual tracking problems.
- A switching method combining a residual \hat{S}_k approximation with the dynamic quasi-Gauss-Newton algorithms with or without partitioning for Broyden's method is developed. To determine a proper switching mechanism, a switching criterion is heuristically selected.
 - Various switching algorithms including the switching MBFGS-DB, the switching DFN-BFGS-DB, and the switching DBFGS-DB algorithms are investigated.
 - Implementations of Scheme 1 and Scheme 2, alternatively combining the dynamic quasi-Newton and dynamic quasi-Gauss-Newton methods, for large residual problems are investigated.
 - Implementation of the residual approximation presented in NL2SOL [17, 25] with the proposed switching method (the switching Fu-DB) for large residual tracking problem is studied.
3. A fundamental development of the novel DAFF algorithm to adaptively calculate λ_k
- A novel DAFF method to heuristically select λ_k is developed.

- Implementation of the VS-ARLS, the GN-VFF-RLS, and the VFF algorithms (originally presented in an adaptive filtering context) with the proposed switching algorithms are investigated for large residual problems.

4. Simulation validation of the proposed switching algorithms with DAFF method

- The switching MBFGS-DB algorithm is found to effectively improve RMS tracking error and settling time t_s compared to the DBM-RLS and the DGN-PBM algorithms originally applied with a fixed forgetting factor for large residual problems.
- Implementation of the LMA with the various switching algorithms and VFF schemes shows only little improvement.
- The DAFF method is found to outperform the other existing VFF algorithms in the presence of noise by offering the best RMS tracking errors, convergence times, and stability.

To put the above contributions into context, a summary of this thesis is presented in Section 7.2.

7.2 *Summary*

Chapter 1 discusses the motivation that leads to the objective of this research, the literature review of uncalibrated visual servoing, and organization of this study.

Chapter 2 reviews the theoretical background of Newton and quasi-Newton methods for solving unconstrained optimization problems. Since in this study visual servoing is formulated as a data driven nonlinear optimization problem, a quasi-Newton method is used for finding a solution. A variety of Newton and quasi-Newton methods and techniques are used to improve various difficulties present in the methods.

Chapter 3 presents a dynamic Broyden's method to estimate a compound Jacobian proposed by Piepmeyer et al. [57, 59, 60]. The important contribution of these

algorithms is their novel development of an uncalibrated visual servoing algorithm using a robot manipulator to track a moving target without an a priori model of either the robot or the camera. The algorithm is used to generate robot joint commands via a dynamic quasi-Newton method that solves a nonlinear least squares problem. This chapter reviews the fundamental development of the *dynamic Broyden's method using recursive least-squares estimation* (DBM-RLS) for a stationary camera and the *dynamic Gauss-Newton algorithm with partitioned Broyden's method* (DGN-PBM) for an eye-in-hand camera. A few major limitations of these algorithms are discussed including a) initial large errors, b) utilizing a fixed forgetting factor, c) ill-conditioned Jacobian matrix. This thesis addresses a) and b) in Chapter 4 and Chapter 5 respectively. A brief study of the LMA for solving c) is investigated in Chapter 6.

Chapter 4 presents derivations of the DBFGS, the DFN-BFGS, and the MBFGS for approximating the residual S for large residual problems. A major disadvantage of the DBM-RLS and the DGN-PBM algorithms are their applicability to the zero- or small-residual cases. This is the nature of the Gauss-Newton based algorithm that neglects the residual S_k in the Hessian H matrix. For large residual problems, the residual S_k is not negligible. Despite the fact that the residual S_k is usually difficult to analytically determine, various algorithms are used to solve large-residual cases. Since the goal is to approximate the Hessian matrix, one solution is to approximate the whole Hessian (the DBFGS algorithm) and the other solution is to approximate the residual \hat{S}_k (the MBFGS and the DFN-BFGS algorithms) by assuming that $J_k^T J_k$ is already available. The derivation of the DBFGS algorithm is analogous to the BFGS derivation for the whole Hessian approximation \hat{H}_k . The major significance of the DBFGS is due to the inclusion of the time-dependent term, $\frac{\partial f_k}{\partial t} h_t$, in the secant equation used to approximate \hat{H}_k . Unlike the DBFGS algorithm, the DFN-BFGS and MBFGS methods attempt to only approximate the residual \hat{S}_k . The DFN-BFGS

method is derived by applying the BFGS algorithm to estimate the residual \hat{S}_k , while the MBFGS algorithm is derived by modifying a denominator of a term in the DFN-BFGS formula. This modification basically enforces the curvature information of the function F_k into the residual S_k approximation. In addition, for the case that \hat{S}_k is relatively large compared to the term $J_k^T J_k$, the MBFGS method is the same as the DFN-BFGS method. Otherwise, the MBFGS method generates distinct results from the unmodified BFGS method.

Since the DBFGS and the DFN-BFGS algorithms use the BFGS method, which provides superlinear convergence under reasonable assumptions, a convergence proof is only required to validate the MBFGS algorithm. The convergence analysis of the MBFGS algorithm is studied in analogy to the convergence analysis of Broyden's class formula presented in [8]. The novel MBFGS algorithm assumably yields superlinear convergence if the specified assumptions hold.

A hybrid between the DGN-PBM algorithm and a proposed residual approximation is developed and is referred to as the *switching method*. The switching method is a heuristic approach to switch from using the full quasi-Newton method (inclusion of \hat{S}_k into \hat{H}_k where \hat{S}_k is estimated using a proposed residual approximation) to the quasi-Gauss-Newton method (neglect \hat{S}_k or the DGN-PBM algorithm). Switching occurs if the image error norm $\|f_k\|$ is less than the switching criterion v , which is heuristically selected. The switching methods presented for residual approximation schemes are the switching DBFGS-DB, the MBFGS-DB, and the DFN-BFGS-DB algorithms.

Chapter 5 discusses the derivation of the DAFF algorithm, a heuristic method for adaptive λ_k calculation. The selection of λ_k is critically influences tracking performance so λ_k must be properly selected. As a result, various existing variable forgetting factor (VFF) algorithms, widely studied in RLS adaptive filtering, are discussed. However, these algorithms are complex, require a number of constant selections, and

do not appear effective for uncalibrated visual servoing, hence, the DAFF method is developed.

Due to the difficulty in obtaining λ_k analytically, a heuristic method is explored. By observation, the image error is typically large in the transient state, especially at the beginning of the tracking process, and becomes relatively small during steady state. Since the inverse of $1 - \lambda$ roughly represents memory, it can be hypothesized that the forgetting factor λ should be small (less memory) when the image error is large during transience. On the contrary, λ should get closer to unity (more memory) when the image error becomes relatively small in steady-state tracking. Thus, the image error norm $\|f_k\|$ and λ_k are somewhat inversely related. This behavior of λ_k is in analogy to a step response of a first-order differential system. An analogous transfer function for which the input function is the image error norm and the output function is the inverse of the memory is developed that ultimately leads to the DAFF algorithm.

Chapter 6 first discusses the proposed switching algorithms, the switching MBFGS-DB, DBFGS-DB, and DFN-BFGS-DB algorithms, on improving tracking performance for large residual problems in simulation. A RRR robot with two eye-to-hand cameras and a PUMA 560 robot with an eye-in-hand camera tracking a variety target trajectories are used. For all tests the starting robot position is located far away from the desired target to ensure a initial large error, no noise is added, and a fixed λ_k is used. The results are then compared with the DBM-RLS and the DGN-PBM algorithms. The switching DFN-BFGS-DB and the MBFGS-DB algorithms are the most effective methods with NP-Jacobian approximation for the RRR robot case. However, for the PUMA 560 robot case the switching MBFGS-DB algorithm provides the best results in terms of RMS errors, t_s , and stability for a complex trajectory or a fast target speed.

The different VFF algorithms presented in Chapter 5 are investigated to validate tracking improvement. Due to parameter selection required for each algorithm, their variation is investigated with the PUMA 560 robot using an eye-in-hand camera tracking a circular trajectory. The performance of each VFF algorithm is in fact insensitive to the variation of parameters so the tracking performance of each VFF algorithm can be reasonably compared. The switching MBFGS with the DAFF algorithm consistently yields the fastest convergence and the smallest RMS tracking errors in the presence of measurement and system noise for both the RRR robot with two eye-to-hand cameras and the PUMA 560 with an eye-in-hand camera. This consistency is also true for a variety of trajectories and target speeds. However, the EE motions substantially deviate from the desired target planes for the RRR robot case. This problem can be minimized if the cameras are arranged perpendicularly. Although the PUMA 560 robot case does not have much of a deviation problem, it is more sensitive to the noise disturbance compared to the RRR robot case. Since the perpendicular camera arrangement is not applicable for the eye-in-hand case, two nearly paralleled eye-in-hand cameras are used. This camera arrangement generates smoother EE trajectories.

The effect of the MBFGS-DB and the DGN-PBM with various VFF algorithms on two implementations, with or without the LMA, is studied. The DAFF gives the best performance as compared to other VFF algorithms for both MBFGS-DB and the DGN-PBM and the LMA has little affect on DAFF performance. In fact the LMA algorithm only marginally improves tracking performance for all tested cases. The switching MBFGS-DB-DAFF without the LMA generates the smallest RMS error and the fastest convergence.

In summary, the switching MBFGS-DB algorithm with the DAFF method consistently yields the best results for a variety of robot degrees-of-freedom, camera

configurations, trajectories, and target speeds. The DAFF algorithm is shown to significantly offer the best overall tracking accuracy and convergence, especially in the presence of noise. The number of cameras and the camera arrangement are shown to significantly affect noise compensation. Although the cases studied in this situation are limited, these results validate the effectiveness of the switching MBFGS-DB algorithm with the DAFF scheme to improve tracking performance for large residual problems in the presence of noise.

7.3 Future Work

Although the switching MBFGS-DB algorithm with the DAFF method is shown to improve tracking performance, there exists a few issues on further improving the effectiveness of this novel algorithm.

One issue is to further improve the Jacobian approximation \hat{J}_k . The proposed residual \hat{S}_k approximations are derived with an assumption that \hat{J}_k is available and approximately represents the actual Jacobian J_k . Since \hat{J}_k is used to approximate \hat{S}_k , this quantity is critical and directly affects how well \hat{S}_k is approximated. Two promising methods have been proposed by Farahmand et al. [22]. In the K-nearest neighborhood method Jacobians are calculated from sensor data using a recursive least squares method. It is also estimated from a database of previously stored Jacobians which are “nearest” to it in configuration space which are used for a weighted interpolation. Using heuristics the best estimate is taken and a decision is made if the stored Jacobian should be updated. In the second method the tip location data is stored as the robot moves in the workspace. To determine the Jacobian at any given state, a hyperplane is fit to the stored location data. Good results have been reported for both methods using an industrial robot.

In the presence of noise tracking performance significantly degrades. The higher

DOF robot with an eye-in-hand case is shown to be very sensitive to noise disturbances. Though the switching MBFGS-DB algorithm with the DAFF method offers the best RMS error and settling time, the EE motion oscillates at least in the range of the added noise in the best cases, i.e., the RRR robot with two eye-to-hand cameras. As a result, a noise filtering algorithm such as a Kalman filter may be implemented, which is routinely used in a variety applications such as adaptive filters, system identification, or sensor fusion. In fact the RLS algorithm can be cast as a special case of the Kalman filtering [32] and the relationship between them is discussed in [32]. Since the switching MBFGS-DB algorithm with the DAFF method utilizes the dynamic Broyden's method to recursively approximate \hat{J}_k , there exists a possibility to augment the Kalman filter into the Jacobian \hat{J}_k approximation.

Due to the potential of using multiple cameras for noise compensation, multiple cameras and camera arrangements can be further investigated to optimally improve tracking. Since the perpendicular camera arrangement for the RRR robot cases considerably minimizes the EE deviation out of the target plane, one or more eye-to-hand cameras may be added to the eye-in-hand PUMA 560 case to improve tracking performance.

Experimental verification of the switching MBFGS-DB algorithm with the DAFF method is required for a practical validation. This study does not address singularity avoidance or the situation where the target exits the robot workspace or camera view and must be considered for experimental realization.

The switching MBFGS-DB algorithm with the DAFF method significantly improves on difficulties encountered in large residual tracking problems. This work is the first successfully developed uncalibrated visual guided control to handle large residual problems for moving target tracking with fast convergence and desirable accuracy.

REFERENCES

- [1] AGHEKSANTERIAN, A., "Line search methods in unconstrained optimization: A new inexact method." UMBC An Honors University in Maryland <http://www.math.umbc.edu/~aa5/articles/fall2006.pdf>, October 18 2006.
- [2] BARTLETT, M. S., "An inverse matrix adjustment arising in discriminant analysis," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 107–111, 1951. J
- [3] BETTS, J. T., "Solving the nonlinear least squares problem: Application of a general method," *Journal of Optimization Theory and Applications*, vol. 18, pp. 469–483, April 1976. J
- [4] BILEN, H., HOCAOGLU, E., OZGUR, M., and SABANOVIC, A., "A comparative study of conventional visual servoing schemes in microsystem applications," in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (San Diego, CA, USA), pp. 1308–1313, Oct 29 - Nov 2 2007. C
- [5] BONKOVIC, M., HACE, A., and JEZERNIK, K., "Population-based uncalibrated visual servoing," *Transactions on Mechatronics, IEEE/ASME*, vol. 13, no. 3, pp. 393–397, 2008. J
- [6] BROYDEN, C. G., "Quasi-newton methods and their application to function minimization," *Mathematics of Computation*, vol. 21, pp. 368–381, 1967. J
- [7] BROYDEN, C. G., DENNIS, J. E., and MORÉ, J. J., "On the local and super-linear convergence of quasi-newton methods," *IMA Journal of Applied Mathematics*, vol. 12, no. 3, pp. 223–245, 1973. J
- [8] BYRD, R. H., NOCEDAL, J., and YUAN, Y. X., "Global convergence of a class of quasi-newton methods on convex problems," *SIAM Journal on Numerical Analysis*, vol. 24, no. 5, pp. 1171–1190, 1987. J
- [9] CONN, A. R., GOULD, N. I. M., and TOINT, P. L., *Trust-Region Methods*. MPS-SIAM Series on Optimization, Philadelphia: SIAM, 2000. J
- [10] CORKE, P. I., "A robotics toolbox for matlab," *IEEE Robotics and Automation Magazine*, vol. 3, no. 1, pp. 24–32, 1996. J
- [11] CORKE, P., "Machine vision toolbox," *IEEE Robotics and Automation Magazine*, vol. 12, pp. 16–25, Nov. 2005. J
- [12] DAVIDON, W. C., "Variance algorithm for minimization," *The Computer Journal*, vol. 10, no. 4, pp. 406–410, 1968. J

- [13] DAVIDON, W. C., "Optimally conditioned optimization algorithms without line searches," *Mathematical Programming*, vol. 9, no. 1, pp. 1–30, 1975.
- [14] DEMENTHON, D. and DAVIS, L., "Model-based object pose in 25 lines of code," *International Journal of Computer Vision*, vol. 15, pp. 123–141, June 1995.
- [15] DENNIS, JR., J. E., "Some computational techniques for the nonlinear least squares problem," in *Numerical Solution of Systems of Non-linear Algebraic Equations* (HALL, G. D. B. and A, C., eds.), pp. 157–183, New York: Academic *in-B* Press, 1973.
- [16] DENNIS, JR., J. E., GAY, D. M., and WELSCH, R. E., "An adaptive nonlinear least squares algorithm," tech. rep., NBER, August 1977. *T Report*
- [17] DENNIS, JR., J. E., GAY, D. M., and WELSCH, R. E., "An adaptive nonlinear least-squares algorithm," *ACM Transactions on Mathematical Software (TOMS)*, vol. 7, pp. 348–368, September 1981a.
- [18] DENNIS, JR., J. E. and MORÉ, J. J., "A characterization of superlinear convergence and its application to quasi-newton methods," *Mathematics of Computation*, vol. 28, pp. 549–560, April 1974.
- [19] DENNIS, JR., J. E. and SCHNABEL, R. B., *Numerical methods for unconstrained optimization and nonlinear equation (classics in applied mathematics)*. Philadelphia: SIAM, 1996. *J*
- [20] DENNIS, J. E., J., GAY, D. M., and WELSCH, R. E., "Algorithm 573: NL2sol - an adaptive nonlinear least-squares algorithm [e4]," *ACM Transactions on Mathematical Software (TOMS)*, vol. 7, pp. 369–383, September 1981b.
- [21] DIXON, L. C. W., "Variable metric algorithms: Necessary and sufficient conditions for identical behavior on nonquadratic functions," *Journal of Optimization Theory and Applications*, vol. 10, pp. 34–40, July 1972. *J*
- [22] FARAHMAND, A. M., A., S., and JAGERSAND, M., "Global visual-motor estimation for uncalibrated visual servoing," in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (San Diego, CA, USA), pp. 1969–74, 2007. *C*
- [23] FLETCHER, R., *Practical methods of optimization, Second Edition*. John Wiley & Sons, 1987. *B*
- [24] FRANKLIN, G. F. and POWELL, J. D., *Digital Control of Dynamic Systems*. Philippines: Addison-Wesley, 1981. *B*
- [25] FU, Q., ZHANG, Z., and SHI, J., "Uncalibrated visual servoing using more precise model," in *2008 IEEE Conference on Robotics, Automation, and Mechatronics*, (Chengdu, China), pp. 916–921, 2008. *C*

- [26] GILL, P. E. and MURRAY, W., "Numerically stable methods for quadratic programming," *Mathematical Programming*, vol. 14, pp. 349–372, 1978. J
- [27] GILL, P. E., MICHAEL, and LEONARD, M. W., "Reduced-hessian quasi-newton methods for unconstrained optimization," *SIAM Journal on Optimization*, vol. 12, no. 1, pp. 209–237, 2001. J
- [28] GRIEWANK, A. and TOINT, P. L., "Local convergence analysis of partitioned quasi-newton updates," *Numerische Mathematik*, vol. 39, October 1982. J
- [29] HAO, M., DEUFLHARD, P., SUN, Z., and FUJII, M., "Model-free uncalibrated visual servoing using recursive least squares," *Journal of Computers*, vol. 3, pp. 42–50, November 2008. J
- [30] HAO, M., SUN, Z., FUJII, M., and SONG, W., "Uncalibrated eye-in-hand visual servoing using recursive least squares," in *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, (Montreal, QC, Canada), pp. 64–69, October 7-10 2007. C
- [31] HARVILLE, D. A., *Matrix Algebra From a Statistician's Perspective*. New York: Springer, corrected ed., 2008. B
- [32] HAYKIN, S., *Adaptive filter theory*. Englewood Cliffs, NJ: Prentice-Hall, 1996. B
- [33] HILL, J. and PARK, W. T., "Real time control of a robot with a mobile camera," in *Proceedings of the 9th ISIR*, pp. 233–246, March 1979. C
- [34] HOSODA, K. and ASADA, M., "Versatile visual servoing without knowledge of true jacobian," in *IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, (Munich, Germany), pp. 186–193, 1994. C
- [35] HOSODA, K., IGARASHI, K., and ASADA, M., "Adaptive hybrid control for visual servoing and force servoing in an unknown environment," *IEEE Robotics and Automation Magazine*, vol. 5, no. 4, pp. 39–43, 1998. J
- [36] HUTCHINSON, S., HAGER, G. D., and CORKE, P. I., "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 651–670, October 1996. J
- [37] JAGERSAND, M., FUENTES, O., and NELSON, R., "Experimental evaluation of uncalibrated visual servoing fro recision manipulation," in *Proceedings of International Conference on Robotics and Automation*, (Albuquerque, NM), pp. 2874–80, 1997. C
- [38] KIM, G. W. and LEE, B. H., "Adaptive regulation of robot joint velocity in uncalibrated visual servoing," in *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, (New Orleans, LA), pp. 739–744, 2004. C

- [39] KIM, G. W. and LEE, B. H., "Target tracking using the efficient estimation of the image jacobian with large residual," *Robotica*, vol. 24, pp. 325–327, 2006. J
- [40] KIM, G. W., LEE, B. H., and KIM, M. S., "Uncalibrated visual servoing technique using large residual," in *Proceedings of the 2003 IEEE International Conference on Robotics & Automation*, (Taipei, Taiwan), pp. 3315–3320, 2003. C
- [41] LEUNG, S.-H. and SO, C. F., "Gradient-based variable forgetting factor rls algorithm in time-varying environments," *IEEE Transactions on Signal Processing*, vol. 53, no. 8, pp. 3141–3150, 2005. J
- [42] LEVENBERG, K., "A method for the solution of certain non-linear problems in least squares," *The Quarterly of Applied Mathematics*, vol. 2, pp. 164–168, 1944. J
- [43] LOURAKIS, M. I. A., "A brief description of the levenberg-marquardt algorithm implemented by levmar. foundation for research and technology," 2005.
- [44] LV, X. and HUANG, X., "Fuzzy adaptive kalman filtering based estimation of image jacobian for uncalibrated visual servoing," in *Proceeding of the 2006 IEEE/RSJ International Conference on intelligent Robots and Systems*, (Beijing, China), pp. 2167–2172, Oct 9-15 2006. C
- [45] MARQUARDT, D., "An algorithm for least-squares estimation of nonlinear parameters," *SIAM Journal on Applied Mathematics*, vol. 11, pp. 431–441, 1963. J
- [46] MIURA, K., HASHIMOTO, K., GANGLOFF, J., and MATHELIN, M. D., "Visual servoing without jacobian using modified simplex optimization," in *Proceeding of the 2003 IEEE International Conference on Robotics and Automation*, pp. 3315–3320, September 2005. C
- [47] MORÉ, J. J., "The levenberg-marquardt algorithm: Implementation and theory," *Lecture Notes in Mathematics*, vol. 630, pp. 105–116, 1978. J
- [48] NAZARETH, J. L., "A hybrid least squares method," tech. rep., Argonne National Laboratory, 1975. T. Report
- [49] NAZARETH, L., "Some recent approaches to solving large residual nonlinear least squares problems," *SIAM Review*, vol. 22, no. 1, pp. 1–11, 1980. J
- [50] NOCEDAL, J. and WRIGHT, S. J., *Numerical optimization*. New York, NY: Springer, 1999. B
- [51] OREN, S. S., "Self-scaling variable metric algorithms without line search for unconstrained minimization," *Mathematics of Computation*, vol. 27, no. 124, pp. 873–885, 1973. J
- [52] OZGUR, E. and UNEL, M., "Positioning the trajectory following tasks in microsystems using model free visual servoing," in *The 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON)*, (Taipei, Taiwan), pp. 886–891, Nov 5-8 2007. C

- [53] PALEOLOGU, C., BENESTY, J., and CIOCHINĂ, S., “A robust variable forgetting factor recursive least-squares algorithm for system identification,” *IEEE Signal Processing Letters*, vol. 15, pp. 597–600, 2008. J
- [54] PARK, D. J. and JUN, B. E., “Self-perturbing rls algorithm with fast tracking capability,” *Electronics Letters*, vol. 28, pp. 558–559, March 12 1992. J
- [55] PARK, D. J., JUN, B. E., and KIM, J. H., “Fast tracking rls algorithm using novel variable forgetting factor with unity zone,” *Electronics Letters*, vol. 27, no. 23, pp. 2150–2151, 1991. J
- [56] PEARSON, J., “Variable metric methods of minimization,” *Computer Journal*, vol. 12, pp. 171–178, 1969. J
- [57] PIEPMEIER, J. A., *A Dynamic quasi-Newton method for model independent visual servoing*. PhD thesis, Georgia Institute of Technology, 1999. thesis
- [58] PIEPMEIER, J. A., “Experimental results for uncalibrated eye-in-hand visual servoing,” in *IEEE Southeastern Symposium on System Theory*, (Morgantown, WV), pp. 335–339, 2003. J
- [59] PIEPMEIER, J. A. and LIPKIN, H., “Uncalibrated eye-in-hand visual servoing,” *The International Journal of Robotics Research*, vol. 22, pp. 805–819, 2003. J
- [60] PIEPMEIER, J. A., McMURRAY, G. V., and LIPKIN, H., “Uncalibrated dynamic visual servoing,” *IEEE Transactions on Robotics and Automation*, vol. 20, pp. 143–147, Feb. 2004. J
- [61] POWELL, M. J. D., “A new algorithm for unconstrained optimization,” in *Nonlinear Programming* (ROSEN, J. B., MANGASARIAN, O. L., and RITTER, K., eds.), pp. 31–65, New York: Academic Press, 1970. B
- [62] POWELL, M. J. D., “Some global convergence properties of a variable metric algorithm for minimization without exact line searches,” in *Nonlinear Programming, SIAM-AMS Proceedings* (COTTLE, R. W. and LEMKE, C., eds.), Philadelphia: Scociety for Industrial and Applied Mathematics, 1976. C
- [63] QIAN, J. and SU, J., “Online estimation of image jacobian matrix by kalman-bucy filter for uncalibrated stereo vision feedback,” in *Proceedings ICRA’02 IEEE International conference on Robotics and Automation*, vol. 1, pp. 562–567, Aug 2002. C
- [64] RALIS, S. J., B., V., and NELSON, B. J., “Micropositioning of a weekly calibrated microassembly system using coarse-to-fine visual servoing strategies,” *IEEE Transactions on Electronics Packaging Manufacturing*, vol. 23, pp. 123–131, Apr 2000. J

- [65] RENNIE, J. D. M., "Relating the trace and frobenius matrix norms." <http://people.csail.mit.edu/jrennie/writing/traceFrobenius.pdf>, August 31 2005. X
- [66] RITTER, K., "Local and superlinear convergence of a class of variable metric methods," *Computing*, vol. 23, pp. 287–297, 1979. J
- [67] SEBER, G. A. F. and WILD, C. J., *Nonlinear Regression*. Wiley series in probability and mathematical statics, New York, US: John Wiley & Sons, 1989. B
- [68] SHERMAN, J. and MORRISON, W. J., "Adjustment of an inverse matrix corresponding to a change in one element of a given matrix," *The Annals of Mathematical Statistics*, vol. 21, no. 1, pp. 124–127, 1950. J
- [69] SLOCK, D. T. M. and KAILATH, T., "Fast transversal filters with data sequence weighting," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 3, pp. 346–359, 1989. J
- [70] SO, C. F., NG, S. C., and LEUNG, S. H., "Gradient based variable forgetting factor rls algorithm," *Signal Processing*, vol. 83, pp. 1163–1175, 2003. J
- [71] SONG, S., LIM, J.-S., BAEK, S., and SUNG, K.-M., "Gauss newton variable forgetting factor recursive least squares for time varying parameter tracking," *Electronics Letters*, vol. 36, no. 11, pp. 988–990, 2000. J
- [72] STACHURSKI, A., "Superlinear convergence of broyden's bounded θ -class of methods," *Mathematical Programming*, vol. 20, pp. 196–212, December 1981. J
- [73] STEELE, J., *The Cauchy-Schwarz Master Class: An Introduction to the Art of Mathematical Inequalities (Maa Problem Books Series.)*. Cambridge, United Kingdom: Cambridge University Press, 2004. B
- [74] STOER, J., "On the convergence rate of imperfect minimization algorithms in broyden's beta-class," *Mathematical Programming*, vol. 9, pp. 313–335, 1975. J
- [75] STRANG, G., *Linear algebra and its applications*. Philadelphia, PA: Brooks/Cole, third edition ed., 1988. B
- [76] VANDEN BERGHEN, I. F., "Trust region algorithms." Kranf Site <http://www.applied-mathematics.net/index.html>. X
- [77] VANDEN BERGHEN, I. F., *CONDOR: a constrained, non-linear, derivative-free parallel optimizer for continuous, high computing load, noisy objective functions*. Phd thesis, University of Brussels, 2004. Thesis
- [78] WANG, H. and H., L. Y., "Dynamic visual servoing of robots using uncalibrated eye-in-hand visual feedback," in *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Beijing, China), pp. 3797–3802, Oct 9 -15 2006. C

- [79] WEISS, L. E., SANDERSON, A. C., and NEUMAN, C. P., "Dynamic sensor-based control of robots with visual feedback," *IEEE Journal of Robotics and Automation*, vol. RA-3, pp. 404–417, October 1987.
- [80] WILSON, W. J., HULLS, C. C. W., and BELL, G. S., "Relative end-effector control using cartesian position-based visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 684–696, Oct 1996.

5

5

VITA

Jomkwun Munnae was born in Phetchaburi, a province in the central of Thailand. In March 1997 she graduated from Satriwittaya School in Bangkok, Thailand. She was awarded a Thai scholarship to pursue higher education in the United States in April 1997. After spending one year at Dana Hall School in Wellesley, MA, she attended the University of Wisconsin at Madison for her Bachelor's degree in Mechanical Engineering. After graduating in 2002, she pursued her graduate study and was awarded a M.S. Mechanical Engineering degree from the Georgia Institute of Technology in May of 2004. There she developed her interests in robotics control. She worked as a graduate research assistant at the ATAS Laboratory, a unit of the Georgia Technology Research Institute, providing engineering solutions to a variety of industries across the nation. This experience bridged her academic experience with solving real-world technical problems.