Active Sensing with Applications to Mobile Robotics

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Ke Zhou

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Stergios I. Roumeliotis, Adviser

June, 2012

Acknowledgements

There are many people that have earned my gratitude for their contribution to my time in graduate school.

First of all, I would like to express my sincere gratitude to my academic and thesis adviser, Prof. Stergios Roumeliotis, for sharing his wisdom and knowledge without any reservation, spending countless time in valuable discussions, correcting my errors with great patience, and constantly encouraging and believing in me during the lows of my life. He has established a high academic standard which I have learned and benefited a lot from. He has created and provided me with excellent research opportunities which I greatly appreciate. He has devoted significant amount of time and efforts to improve the quality of every chapter of my thesis. This dissertation would not have been possible without his guidance, help, and encouragement. I also gratefully acknowledge financial support throughout my PhD study from the University of Minnesota Digital Technology Center (DTC), the National Science Foundation (NSF), the Air Force Office of Scientific Research (AFOSR), and the National Aeronautics and Space Administration (NASA) Mars Technology Program.

I would like to thank my preliminary and final oral committee members, Prof. Georgios Giannakis, Prof. Yousef Saad, Prof. Nikos Sidiropoulos, Prof. Zhi-Quan (Tom) Luo, and Prof. Tryphon Georgiou, for spending their valuable time and providing insightful comments, feedbacks, and suggestions to improve my work. I would also like to thank Linda Jagerson, who always provides prompt assistances whenever I have any questions regarding every procedure of the Electrical Engineering Ph.D. program completion.

I would like to acknowledge my colleagues, Anastasios Mourikis, Nikolas Trawny, Faraz Mirzaei, Xun (Sam) Zhou, Joel Hesch, Guoquan (Paul) Huang, Esha Nerurkar,

Igor Melnyk, Kejian Wu, Chao Guo, Dimitrios Kottas, Sean Bowman, Gian-Luca Mariottini, Paolo Stegagno, and Serena Pani, for all fruitful discussions, and offering their help without hesitation whenever I need it. I truly enjoy the time with your guys inside and outside labs.

My special thanks go to Har and Char Anderson, Doug and Linda Peacock, Charles and Barbara Welner, who treat us as their close relatives during our stay in USA. Their kindness and love towards my family make me feel warm even we are far away from China. I would also like to thank Pastor and Mrs. Malone, and all members from All Nations Baptist Church, who constantly provide help to my family, not only physically but also spiritually.

I am very grateful to my parents in China for their unconditional love. They have sacrificed a lot to provide me the opportunity to pursue the advanced degree overseas, and have endured many years of loneliness without any complaints, for these I am forever in debt.

My daughter Grace brings a lot of fun during my PhD career, and makes my life colorful. I would like to express my sincere love to my wife Xianhua Jiang, who as a PhD student, provides me supports in all areas of life, shares joys and tears with me during all the ups and downs throughout our marriage life, and takes good care of me and Grace, and makes my life complete and beautiful. I thank God for bringing her and Grace into my life.

Finally, I am forever grateful to my personal Savior Jesus Christ, who gives me hope and faithfully delivers me through all difficulties and hardships. Your grace is sufficient for me throughout these years (2 Corinthians 12:9), and nothing can separate us from the love of God in Christ Jesus our Lord (Romans 8: 35–39). May all glories return back to Him, who is worthy of our worship and praise.

Dedication

To Xianhua Jiang and Grace Yue Zhou.

Abstract

Compared to static sensors, mobile robots offer significant advantages primarily due to their ability to move and sense the world from multiple vantage points. However, they also pose significant challenges due to limitations on their resources. Specifically, moving a robot to a new location consumes energy, and thus, it is of paramount importance to design optimal motion strategies for mobile robots that maximize task performance while minimizing energy usage.

Active sensing seeks to maximize the efficiency of an estimation task by actively controlling the sensing parameters. Of particular interest for mobile robot teams is the case where active sensing is used for determining the locations at which the robots should move to in order to acquire the most informative measurements. By determining the optimal sensing locations that minimize the estimation uncertainty, active sensing enables a robot team to achieve the desired level of accuracy faster and more efficiently as compared to a random sensing strategy.

In this dissertation, we investigate active sensing algorithms for the problems of (i) leader-follower formation control, which is referred to as "active formation control"; and (ii) target tracking, which is termed as "active target tracking". Furthermore, since precise robot localization is a prerequisite for optimal active sensing, we next focus on reducing the complexity of cooperative localization.

The first part of this thesis investigates the problem of active formation control, where our objective is to determine the optimal trajectory for a robot in a leader-follower formation that minimizes its localization uncertainty. In particular, maintaining a perfect formation has been shown to increase the localization uncertainty (as compared to moving randomly), or even to lead to loss of observability when only bearing measurements are available and the robots move on parallel straight lines. To address this issue, we allow the follower to slightly deviate from its desired formation-imposed position and seek to find the next best location where it should move to in order to minimize the uncertainty about its relative pose (position and orientation) estimate, with respect to the leader. Our main contribution is that we formulate and analytically compute the global optimum for this constrained non-convex optimization problem.

The second part of this thesis focuses on active target tracking, where our objective is to select the best sensing locations of tracking robots so as to maximize the target-position-estimates' accuracy. More specifically, we introduce an algorithm that analytically computes the global optimal solution of the one-step-ahead (that is, determining the sensing location at the next time step) single-robot active target tracking problem. Furthermore, we show that the problem of one-step-ahead multi-robot active target tracking is NP-Hard. We then relax the original NP-Hard problem and propose a cyclic coordinate descent algorithm (also called nonlinear Gauss-Seidel relaxation), for determining the next sensing location for each robot, whose computational requirements scale only linearly in the number of robots. Finally, we investigate the problem of multi-step-ahead (that is, generating a sequence of optimal sensing locations over a finite time horizon) single-robot active target tracking, and introduce an efficient algorithm based on the nonlinear Gauss-Seidel relaxation, whose computational complexity is quadratic in the number of time steps considered.

The final part of this thesis focuses on reducing the computational complexity of multi-robot cooperative localization. In particular, we aim at reducing the processing requirements of the covariance update step when employing the extended Kalman filter (EKF). In contrast to the standard EKF, whose time complexity is quartic in the number of robots, we introduce an efficient algorithm, named the Modified Householder QR, which exploits the special sparse structure of the measurement (Jacobian) matrix, to reduce the processing cost to cubic.

In summary, by introducing active sensing algorithms for efficiently solving important problems that arise in robotics (leader-follower formation control, target tracking), and by reducing the computational complexity of cooperative localization, the research presented in this dissertation aims at optimizing resource utilization while minimizing the operational cost of mobile robots deployed in challenging real-world applications.

Contents

Acknowledgements **i**

Dedication **iii**

Abstract **iv**

List of Tables **xi**

List of Figures **xii**

1 Introduction **1**

1.1 Active sensing 1

1.1.1 Active sensing in mobile robotics 3

1.2 Research objectives and overview 4

1.2.1 Active formation control 4

1.2.2 Active target tracking 5

1.2.3 Complexity reduction in CL 8

1.3 Organization of the manuscript 9

2 Active Formation Control **11**

2.1 Introduction 11

2.2 Literature review 13

2.3 Problem formulation 15

2.3.1 System kinematic model 16

2.3.2 State and covariance propagation 17

2.3.3 Measurement update 18
2.3.4 Optimization problem formulation 19
2.4 Solution strategy 20
2.4.1 Inactive constraint 23
2.4.2 Active constraint 25
2.4.3 Globally optimal solution 27
2.5 Simulation results 27
2.6 Summary 29

3 Active Target Tracking: Multi-sensor Optimization 31
3.1 Introduction 31
3.2 Literature review 35
3.2.1 Multi-sensor target tracking: distance-only observations 35
3.2.2 Multi-sensor target tracking: distance-and-bearing observations . 36
3.2.3 Summary of related work 37
3.3 Problem formulation 38
3.3.1 State propagation 39
3.3.2 Measurement model 40
3.3.3 State and covariance update 44
3.3.4 Problem statement and reformulation 46
3.4 Complexity analysis 49
3.5 Single-sensor active target tracking 55
3.5.1 Case I: $0 < \rho \leq \|\mathbf{c}\| - r$ 58
3.5.2 Case II: $\sqrt{\|\mathbf{c}\|^2 - r^2} \leq \rho < \|\mathbf{c}\| + r$ 66
3.5.3 Case III: $\|\mathbf{c}\| - r < \rho < \sqrt{\|\mathbf{c}\|^2 - r^2}$ 68
3.5.4 Case IV: $\|\mathbf{c}\| + r \leq \rho$ 69
3.5.5 Summary: single-sensor solution algorithm 69
3.5.6 Extension to obstacle avoidance 69
3.6 Multiple-sensor active target tracking 71
3.7 Simulation results 73
3.7.1 Simulation setup 74
3.7.2 Target tracking with 2 sensors (homogeneous team) 75

3.7.3 Target tracking with 3 sensors (heterogeneous team) 76
3.7.4 Scalability and run-time 80
3.8 Experimental results 81
3.9 Summary 84

4 Active Target Tracking: Multi-step-ahead Optimization 88
4.1 Introduction 88
4.2 Literature review 91
4.2.1 Analytical solutions for restricted target motion models 91
4.2.2 Numerical solutions for arbitrary target motion models 92
4.3 Problem formulation 94
4.3.1 Predicted target's state estimate and prior covariance 94
4.3.2 Predicted posterior covariance 95
4.3.3 Measurement variance inflation 97
4.3.4 Problem statement 98
4.3.5 Problem analysis 100
4.4 Solution strategy 102
4.4.1 Solution strategy for single-step optimization 102
4.4.2 Solution strategy for the multi-step optimization 107
4.5 Simulation results 109
4.5.1 Simulation setup 110
4.5.2 Target tracking over 3 time steps ahead 110
4.5.3 Target tracking over 5 time steps ahead 113
4.5.4 Root mean square error 115
4.6 Summary 117

5 Complexity Reduction in Cooperative Localization (CL) 119
5.1 Introduction 119
5.2 Literature review 122
5.2.1 MLE or MAP-based estimators 122
5.2.2 EKF-based estimators 124
5.3 Problem formulation 125
5.3.1 State propagation 125

5.3.2 Measurement model . . . 126
5.3.3 State and covariance update . . . 128
5.3.4 State and covariance update through QR factorization . . . 131
5.4 Standard Householder QR . . . 132
5.4.1 Householder reflections . . . 133
5.4.2 Description of the Standard Householder QR . . . 133
5.4.3 Complexity analysis of the Standard Householder QR . . . 136
5.5 Modified Householder QR . . . 144
5.5.1 Description of the Modified Householder QR . . . 145
5.5.2 Complexity analysis of the Modified Householder QR . . . 152
5.6 Simulation results . . . 154
5.7 Summary . . . 160

6 Concluding Remarks **161**
6.1 Summary of contributions . . . 161
6.2 Future research directions . . . 163

References **166**

Appendix A. Appendices for Chapter 3 **182**
A.1 Rational functions $f_0(\mathbf{s}^*)$ and $\nabla f_0(\mathbf{s}^*)$ . . . 182
A.2 Transforming (3.58) into (3.59) . . . 183
A.3 Coefficients of (3.59) for the case $\kappa_d = \kappa_\theta = 1, \lambda_1 > \lambda_2$ . . . 185
A.4 Coefficients of (3.59) for the case $\kappa_d = 0, \kappa_\theta = 1, \lambda_1 > \lambda_2$ . . . 185
A.5 Coefficients of (3.59) for the case $\kappa_d = 1, \kappa_\theta = 0, \lambda_1 > \lambda_2$ . . . 186
A.6 Cartesian coordinates for tangent points A & B . . . 187
A.7 Coefficients of (3.59) for the case $\lambda_1 = \lambda_2$ . . . 188
A.8 Transforming (3.74) into (3.75) . . . 189
A.9 Feasibility of the critical points . . . 190
A.10 Cartesian coordinates for intersections E & F . . . 190
A.11 Special cases of (A.13) . . . 191

Appendix B. Appendix for Chapter 5 **193**
B.1 Computational complexity of alternative QR decomposition algorithms . 193
B.1.1 Cholesky decomposition 193
B.1.2 Modified Gram-Schmidt 194
B.1.3 Givens QR 195

List of Tables

3.1 Computational time (sec) 82
4.1 $[q = 1]$ Target's position RMS error (m) 115
4.2 $[q = 10^{-6}]$ Target's position RMS error (m) 117
5.1 Time complexity per iteration of the Modified Householder QR 153
5.2 CPU runtime (sec) 157
5.3 Frobenius norm of $\mathbf{H}^{\mathrm{T}}\mathbf{H} - \mathbf{R}_{\mathbf{H}}^{\mathrm{T}}\mathbf{R}_{\mathbf{H}}$ 159

List of Figures

2.1 A leader, L, and a follower, F, are moving on parallel straight lines. The relative pose between the leader and follower is unobservable when the follower only measures the bearing β towards the leader, since the follower can be anywhere along the line of sight of the leader. 12

2.2 A leader-follower formation: the follower F is at the desired position $\mathbf{p}_0$ with respect to the leader L. The follower measures its bearing β to the leader. In order to preserve system observability and minimize the relative pose estimation uncertainty, the follower is allowed to move inside the circle centered at $\mathbf{p}_0$ with radius r. 16

2.3 The measurement Jacobian $\mathbf{h}_\beta$ is a function of d and θ, which represent the polar coordinates of the predicted relative position $\mathbf{p}$. 22

2.4 [Monte Carlo simulations] Average weighted trace of the follower's posterior covariance matrix in 50 experiments. 28

2.5 The estimated trajectories of the follower and the leader, when employing RAM, GBS, or MME as motion strategy. The ellipses denote the 3σ bounds for the follower's position uncertainty at the corresponding time steps. 29

3.1 (a) Suboptimal target tracking: The robot remains in the same location. (b) Optimal target tracking: The robot moves to the position that minimizes the uncertainty for the target's position along the horizontal direction. In both plots, the prior uncertainty (3σ) is denoted by a solid-line ellipse, the posterior by a dashed-line ellipse, while the measurement uncertainty is depicted as a circular ring (dotted-line) with center the location of the robot. 33

3.2 Illustration of the i-th sensor's and target's motion: Sensor-i moves in 2D with speed v_i, which is bounded by $v_{i\max}$. From time-step k to $k+1$, the sensor can only move *within* a circular region centered at its position at time-step k with radius $v_{i\max}\delta t$. Furthermore, to avoid collision with the target, sensor-i is prohibited to move inside a circular region centered at the target's position *estimate* at time-step $k+1$ with radius ρ_i. ${}^{S_i}\mathbf{p}_T$ is the target's position with respect to sensor-i. The distance measurement of sensor-i is the norm of ${}^{S_i}\mathbf{p}_T(k+1)$ plus noise, and the bearing measurement of sensor-i is $\theta_i(k+1)$ plus noise. 39

3.3 Geometric interpretation of the polar angle constraints: The circular constraint on $\mathbf{s}_i$ is transformed into an interval constraint on the polar angle $\bar{\theta}_i$, i.e., $\bar{\theta}_i \in [\bar{\theta}_{i\,\min}, \bar{\theta}_{i\,\max}], i = 1\ldots, M$. Also note that the objective function for the distance-only target tracking problem depends only on the polar angles $\bar{\theta}_i, i = 1, \ldots, M$. In other words, the cost function remains the same for arbitrary point $\mathbf{s}_i$ located along the ray with angle $\bar{\theta}_i$, $i = 1, \ldots, M$. 50

3.4 Geometric interpretation of the optimal motion strategy problem: The $M+1$ vectors shown have fixed lengths $\bar{\lambda}_i, i = 0, \ldots, M$. The vector $\mathbf{v}_0$ is affixed to the negative x axis, while the direction of each of the vectors $\mathbf{v}_i,\ i = 1, \ldots, M$, can change, within the interval denoted by the enclosing dashed lines, based on the motion of the corresponding sensor. The objective is to find the directions of the vectors $\mathbf{v}_i,\ i = 1, \ldots, M$ – directly related to the optimal polar angles of $\mathbf{s}_i,\ i = 1, \ldots, M$ – that minimize the Euclidean norm of $\sum_{i=0}^{M} \mathbf{v}_i$. 52

3.5 Geometric illustration of Lemma 13. The global optimal solution resides only in Θ, i.e., the portion of the boundary of the feasible set $\bar{\Omega}$ (depicted by the red-colored curve ADB), defined by the two tangent lines OA and OB, which is closest to O. 57

3.6 Four cases of the feasible set $\bar{\Omega}$. (a) Case I: $0 < \rho \leq \|\mathbf{c}\| - r$. (b) Case II: $\sqrt{\|\mathbf{c}\|^2 - r^2} \leq \rho < \|\mathbf{c}\| + r$. (c) Case III: $\|\mathbf{c}\| - r < \rho < \sqrt{\|\mathbf{c}\|^2 - r^2}$. (d) Case IV: $\|\mathbf{c}\| + r \leq \rho$, which corresponds to the feasible set $\bar{\Omega}$ being empty. In the first three cases (a)-(c), the global optimal solution resides in a subset Θ of the boundary of $\bar{\Omega}$, which is depicted by the red-colored curve ADB in Case I, EGF in Case II, $AEGFB$ in Case III, respectively. In the above plots, O is the origin; C is the center of the circle $\|\mathbf{s}-\mathbf{c}\| = r$; A and B are the two tangent points residing in the circle $\|\mathbf{s} - \mathbf{c}\| = r$; E and F are the intersection points of the two circles $\|\mathbf{s} - \mathbf{c}\| = r$ and $\|\mathbf{s}\| = \rho$; the ray starting from O and passing through C intersects the circle $\|\mathbf{s}\| = \rho$ at G, and the circle $\|\mathbf{s} - \mathbf{c}\| = r$ at D and D'. Finally C' is the midpoint between O and C. 59

3.7 Critical points for single-sensor target tracking with distance-only observations. (a) $\max(|c_1|, |c_2|) \leq r$: There exist six critical points, A, B, I, I', J, J'. (b) $|c_2| \leq r \leq |c_1|$: The four critical points are A, B, I, I'. (c) $|c_1| \leq r \leq |c_2|$: The four critical points are A, B, J, J'. (d) $\min(|c_1|, |c_2|) \geq r$: Only A and B are real critical points, and there exists no real solution satisfying $\xi_2(x, y) = 0$ and $f_2(x, y) = 0$ simultaneously. . 64

3.8 [Two-sensor case] Trace of the target's position posterior covariance matrix. Comparison between GBES, GDC, GSR, and RM. 75

3.9 [Two-sensor case, Monte Carlo simulations] Average trace of the target's position posterior covariance matrix in 50 experiments. Comparison between GBES, GDC, GSR, and RM. 76

3.10 [Two-sensor case] Trajectories of the two sensors, and the actual and estimated trajectories of the target, when employing as motion strategy (a) GBES, (b) GDC, (c) GSR, and (d) RM. The ellipses denote the 3σ bounds for the target's position uncertainty at the corresponding time steps. 77

3.11 [Two-sensor case] 2-norm of the actual error between the target's position estimate and its true value. Comparison between GBES, GDC, GSR, and RM. 78

3.12 [Three-sensor case] Trace of the target's position posterior covariance matrix. Comparison between GBES, GDC, GSR, and RM. 78
3.13 [Three-sensor case] Trajectories of the three sensors, and the actual and estimated trajectories of the target, when employing as motion strategy (a) GBES, (b) GDC, (c) GSR, and (d) RM. The ellipses denote the 3σ bounds for the target's position uncertainty at the corresponding time steps. 79
3.14 [Three-sensor case] 2-norm of the actual error between the target's position estimate and its true value. Comparison between GBES, GDC, GSR, and RM. 81
3.15 CPU time vs. number of sensors, when employing GSR as motion strategy. 82
3.16 [Two-sensor case, experimental setup] Three Pioneer robots, each with a pattern board attached on its top. The target is located at the bottom right of the image, while the other two robots act as tracking sensors. . 84
3.17 [Two-sensor case, experimental result] Trace of the target's position posterior covariance matrix, when employing GSR as motion strategy. . . . 85
3.18 [Two-sensor case, experimental result] 2-norm of the error of the target's position posterior estimates, when employing GSR as motion strategy. . 85
3.19 [Two-sensor case, experimental result] Real trajectories of the two sensors, and the actual and estimated trajectories of the target, when employing GSR as motion strategy. The ellipses denote the 3σ bounds for the target's position uncertainty at the corresponding time steps. 86
4.1 (a) The predicted target's positions over multiple time steps can significantly differ from its true positions if the target motion model is inaccurate. (b) Under small process noise variance, the predicted target's positions over a finite time horizon can be accurate enough to carry out the optimization over multiple time steps. In both plots, the target's true trajectory is shown by the blue solid line, while its predicted trajectory is denoted by the dotted black line. The target's true positions are shown as square boxes, while its predicted positions are denoted by circles, whose associated 3σ uncertainties are shown by the dashed line ellipses. 89

4.2 Illustration of the multi-step-ahead single-sensor active target tracking problem: The target's true trajectory is shown by the blue solid line, while its predicted trajectory is denoted by the dotted black line. Due to process noise, the predicted target's position at time-step k, $\hat{\mathbf{p}}_T(k)$, denoted by a circle, always differs from its true value $\mathbf{p}_T(k)$, shown as a square box. Additionally, the 3σ ellipse computed from the predicted prior covariance $\mathbf{P}_k^{\ominus}$, which characterizes the uncertainty of the estimate $\hat{\mathbf{p}}_T(k)$, is ever growing due to the process noise covariance $\mathbf{Q}_j$, $0 \leq j \leq k-1$. Our objective is to determine the optimal trajectory (shown by the red dashed line) for the tracking sensor over a sliding window of fixed length K (i.e., $\mathbf{p}_S(k)$, $k = 1, \ldots, K$, shown as solid circles or disks), so as to minimize the target's position uncertainty at the Kth time step, subject to the sensor's mobility constraints. 95

4.3 [Three time steps ahead; Monte Carlo simulations without measurement noise variance inflation] Average trace of the target's position posterior covariance matrix. Comparison between GBES, GSR, and OSA. 111

4.4 [Three time steps ahead; Monte Carlo simulations without measurement noise variance inflation] Average 2-norm of the error of the target's position posterior estimates. Comparison between GBES, GSR, and OSA. . 111

4.5 [Three time steps ahead; Monte Carlo simulations with measurement noise variance inflation] Average trace of the target's position posterior covariance matrix. Comparison between GBES, GSR, and OSA. 112

4.6 [Three time steps ahead; Monte Carlo simulations with measurement noise variance inflation] Average 2-norm of the error of the target's position posterior estimates. Comparison between GBES, GSR, and OSA. . 113

4.7 [Five time steps ahead; Monte Carlo simulations with measurement noise variance inflation] Average trace of the target's position posterior covariance matrix. Comparison between GSR and OSA. 114

4.8 [Five time steps ahead; Monte Carlo simulations with measurement noise variance inflation] Average 2-norm of the error of the target's position posterior estimates. Comparison between GSR and OSA. 114

4.9 [Five time steps ahead; single simulation with measurement noise variance inflation] The trajectory of the sensor, and the actual and estimated trajectories of the target, when employing as motion strategy (a) GSR and (b) OSA. The ellipses denote the 3σ bounds for the target's position uncertainty at the corresponding time steps. 116
5.1 Sparsity patterns of (a) measurement matrix $\mathbf{H}$ and (b) $\mathbf{H}^{\mathrm{T}}\mathbf{H}$ for $N = 11$, where the blue dots denote the non-zero elements. Note that due to the structure of $\mathbf{H}$, the square matrix $\mathbf{H}^{\mathrm{T}}\mathbf{H}$ is not necessarily sparse, even though $\mathbf{H}$ is sparse. Therefore, the upper triangular matrix $\mathbf{R}$ computed from the QR decomposition of $\mathbf{H}$ is generally dense. 121
5.2 [Monte Carlo simulations] Average CPU runtime of QR decomposition of $\mathbf{H}$ over 120 trials. Comparison between SuiteSparseQR (SS-QR) and the Modified Householder QR (MH-QR). 156
5.3 [Monte Carlo simulations] Average Frobenius norm of $\mathbf{H}^{\mathrm{T}}\mathbf{H} - \mathbf{R}_{\mathbf{H}}^{\mathrm{T}}\mathbf{R}_{\mathbf{H}}$ over 120 trials. Comparison between SuiteSparseQR (SS-QR) and the Modified Householder QR (MH-QR). 158

Chapter 1

Introduction

1.1 Active sensing

Active sensing is an important skill that humans learn while interacting with their environment. The most obvious, but often unnoticed, fact is that human perception is not a passive, but an active process [1]. For example, we often move our head or body in order to obtain a complete view of a perceived object or scene. Additionally, our eyes are capable of adjusting to the illumination level of the environment [1]. Similarly, animals also actively seek information from the environment using their biological sensors. For example, animals use their sense of smell actively in order to deduce the location of the source of food.

One of the earliest definitions of active sensing, also known as adaptive sensing or active perception, introduced in the context of computer vision by Bajcsy around the late 1980's, was stated as "a problem of controlling strategy applied to the data acquisition process which will depend on the current state of data interpretation and the goal or the task of the process" [2]. Bajcsy further illustrated the above definition through various applications in computer vision, such as object recognition [2].

In the context of detection and estimation, which is the focus of this dissertation, we define active sensing as follows:

Definition 1 (Active sensing)**.** *Active sensing can be broadly defined as a strategy whose goal is to reduce the uncertainty about an observed process* (*e.g., target tracking, object*

identification, etc.) *by actively controlling the sensors' sensing parameters* (*e.g., the sensor's location, orientation, sensing range, speed, and operating frequency*).[1]

Remark 2. *Based on Definition 1, active sensing becomes meaningful when the following two conditions are satisfied:*

- *The sensors' parameters are not fixed, and we have control over them.*
- *The uncertainty of the observed process (or equivalently, the information about it) explicitly depends on the sensors' parameters.*

The first assumption also implies that the exact values of the control variables (i.e., the sensors' parameters) are known. However, this assumption may not always be satisfied in practice. For example, in mobile robotics, the robots' true states (or parameters), such as positions and orientations, in most cases are either unknown or approximately known. For this reason, estimators are often used to determine these parameters by fusing additional observations from on-board sensors. Active sensing algorithms then use these parameter estimates (instead of their true values) as inputs for generating control actions. Large estimation errors usually produce sub-optimal control actions, which in turn deteriorate the overall performance of active sensing algorithms. Therefore, accurate estimation of the sensors' parameters is a prerequisite for ensuring optimal performance of active sensing algorithms.

The second assumption makes it possible to achieve better sensing performance by deliberately adjusting and modifying the sensors' parameters. In fact, viewed as an optimization problem, active sensing directly seeks to minimize the detection/estimation uncertainty (or equivalently, maximize the estimation accuracy) by optimizing over these design parameters. Therefore, by applying active sensing algorithms, it is possible to reduce both the time and the number of measurements required to achieve the desired level of accuracy. This will subsequently result in faster task execution and energy

[1] Note that the adjective "active" has different meanings in the two terms "active sensing" and "active sensor". "Active sensor" often refers to a sensor that emits energy, usually in the form of a signal, and receives and measures the reflected signal (e.g., laser scanners). In contrast, "passive" sensors, such as cameras, only perceive signals emitted by other sources (e.g., the sun or lamps) and reflected by objects within its field of view. Based on our definition, however, a camera can be used to perform active sensing, if its viewing direction (i.e., one of its sensing parameters) can be modified to maximize the acquired information.

savings. In contrast, sensors whose parameters are fixed or modified in an *ad hoc* way, can not always guarantee reduction in detection/estimation uncertainty.

Since its introduction by Bajcsy in 1988, active sensing has received considerable attention. For example, active sensing has been applied to computer vision and machine perception [2, 3, 4, 5, 6], objective recognition [7, 8, 9, 10, 11, 12], object-shape recovery [9, 13], and 3-D scene reconstruction [14, 15]. Recently, active sensing has also been extended to sparse signal recovery [16], laser-beam-position estimation [17], odor identification and its source localization [18], cognitive radio [19], dynamic sampling [20], vehicle driving [21, 22], environment monitoring [23], and illumination control [24]. Additionally, over the past few decades, disciplines such as neurobiology have devoted significant research efforts towards improving our understanding of active sensing mechanisms in both humans and animals. However, this subject still remains an active area of research with many important questions unanswered and key problems unsolved.

1.1.1 Active sensing in mobile robotics

Mobile sensors, or robots,[2] have evolved rapidly in recent years due to technology innovations and advancements in micro-mechanics, micro-processors and various sensing modalities. In contrast to static sensors, whose density and sensing locations are fixed, the mobility of robots offers substantial advantages, for example, the flexibility to adapt their spatial configurations/topologies to changes in the environment. However, mobility is achieved at the cost of higher energy consumption. Since the energy resources of robots are inevitably limited, it is necessary to devise active sensing algorithms that optimize their utilization.

Active sensing has been extensively studied in the context of various robotics applications [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]. For example, the idea of optimally choosing the mobile sensors' locations in order to maximize information gain has been applied to the problems of single or multi-robot localization within a known environment [36, 37, 38, 39], multi-robot cooperative localization in an unknown environment [40, 41], simultaneous localization and mapping (SLAM) [42, 43, 44], chemical parameter estimation [45, 46], ocean sampling [47], and target tracking [48, 49, 50, 51,

[2] In the rest of this dissertation, "robot" and "(mobile) sensor" are used interchangeably. Additionally, the two words "measurement" and "observation" share the same meaning.

52, 53, 54, 55, 56, 57, 58, 59].

In this dissertation, we present novel active sensing algorithms for two important problems arising in robotics: (i) robot formation (see Chapter 2) and (ii) target tracking (see Chapters 3 & 4). Furthermore, since our active target tracking algorithms are developed under the assumption that the true pose (position and orientation) of each tracking sensor is perfectly known, we also investigate methods for exploiting all measurements available to a robot team in order to improve their localization accuracy. Specifically, we focus on the extended Kalman filter (EKF)-based multi-robot cooperative localization (CL) and introduce a method for reducing the computational complexity of the EKF-based CL (see Chapter 5).

1.2 Research objectives and overview

1.2.1 Active formation control

Multi-robot (or multi-vehicle) formation control is a well-studied problem, and various methods have been proposed for solving it. It is well-known that the performance of any algorithm employed for formation control depends heavily on the accuracy and reliability of the robots' relative pose estimates. However, the accuracy of pose estimation degrades when robots move in formation. In particular, when robots move on parallel straight lines and only record relative bearing measurements, Mariottini *et al.* have shown that the system becomes unobservable [60, 61]. To address this issue, Mariottini *et al.* proposed an active control strategy where a perturbation is introduced to the nominal trajectory of the follower robot, in a leader-follower formation, so as to ensure observability of the system [61]. However, [61] does not assess the quantitative impact of the perturbation on the relative-pose estimate's accuracy, and neither seeks to find the optimal perturbation that minimizes the estimation uncertainty.

Similar to SLAM [42, 43, 44], the sensing locations where robots in formation measure each other have a significant impact on the accuracy of the relative poses' estimates. This motivates us to develop active sensing algorithms for the problem of formation control and seek to find optimal motion strategies for the follower robots. We formulate this as a constrained optimization problem and our key contribution is to develop a constant-time algorithm that calculates the global minimum in closed-form.

More specifically, we consider a leader-follower formation, where the leader moves on a straight line while the follower measures its relative bearing towards the leader and maintains a desired formation. As mentioned before, the overall system becomes unobservable and the estimation uncertainty of the relative pose increases unboundedly when the follower robot maintains a perfect formation [60]. To address this issue, we exploit the idea of perturbation introduced in [61] and allow the follower to deviate from the exact formation, but at the same time confine it within a disk centered at its desired location. The size of the disk is chosen to balance the trade-off between maintaining the formation and improving the localization accuracy. We seek the best sensing location inside the aforementioned disk where the follower should move to in order to minimize the relative pose estimation uncertainty. Even though this is a non-convex optimization problem, we are able to leverage tools from algebraic geometry and analytically compute its globally optimum solution (i.e., the location where the follower should move to so as to collect the most informative relative bearing observation). In contrast to [61], where the proposed random perturbation can only ensure the system observability, our optimal strategy not only preserves observability, but also achieves the maximum accuracy for a particular deviation from the desired formation.

In conclusion, by providing highly-reliable relative pose estimates as control inputs, our work can improve the performance of various formation control algorithms. Furthermore, it is possible to employ robots equipped with less accurate bearing-sensors that follow optimal trajectories computed by our algorithm, and achieve performance comparable to that of robots equipped with more accurate (and often expensive) bearing-sensors that either seek to maintain the exact formation or randomly deviate from it.

1.2.2 Active target tracking

Target tracking using mobile robots has recently attracted significant interest in the research community because of its importance in a variety of applications, such as environmental monitoring, surveillance, human-robot interaction, as well as defense applications [62, 63]. In most cases, robots track a moving target by measuring and fusing distance and/or bearing observations to the target. Since these measurements are non-linear functions of the relative positions between the target and the robots, it has been

well understood that the sensing locations of the robots can significantly affect the target's state estimation accuracy. Active target tracking seeks to improve the estimation accuracy of the target state by actively controlling the sensing locations of tracking robots, and it is particularly preferable in tracking applications when the time needed for determining the trajectory of a target is critically important (e.g., when tracking an evading enemy). In these scenarios, sensors that actively pursue a target can minimize the estimation uncertainty about the target's state much faster as compared to a random motion strategy. Due to this advantage, over the past decade, numerous active target tracking algorithms have been developed for generating sensing locations at the next time step, or over a finite time horizon, for a single or multiple sensors. The majority of work, however, has focused on either (i) analytically determining the optimal sensing trajectories for a specific, deterministic target motion model; or (ii) numerically computing the sensing locations for arbitrary target motion models by employing general nonlinear programming techniques, such as gradient descent and grid-based exhaustive search. As a result, the issue of the computational complexity of the underlying optimization problem of active target tracking with respect to the problem size (i.e., the number of robots and the number of time steps considered), as well as the impact of each robot's mobility constraints on the computational complexity, have received little attention in literature. Furthermore, the structure of the optimization problem has not been fully exploited in developing efficient algorithms.

In the second part of this dissertation (see Chapters 3 & 4), we address the aforementioned limitations. Our objective is to determine the optimal sensing locations, subject to the robots' mobility constraints, where the robots will collect the most informative distance and/or bearing observations. The most informative observations are defined in the sense that after fusing these measurements, the trace of the posterior covariance matrix that characterizes the target's position-estimate uncertainty is minimized. Our main contributions are: (i) we show that the multi-sensor active target tracking problem is NP-Hard; and (ii) we propose two polynomial-time algorithms that compute approximate solutions for (a) the multi-sensor and (b) the multi-step-ahead active target tracking problems, respectively. Most importantly, we demonstrate that our computationally-efficient algorithms achieve comparable tracking performance as that of the grid-based exhaustive search, whose complexity is exponential both in the

number of sensors and in the number of time steps considered.

More specifically, we begin (see Chapter 3) by addressing the problem of one-step-ahead multi-sensor active target tracking, and prove that by imposing maximum-speed constraints on the mobile sensors, the resulting constrained nonlinear optimization problem is NP-Hard. This indicates that the optimal solution of the one-step-ahead multi-sensor active target tracking problem is computationally intractable. In order to develop efficient methods for computing an approximate solution to this NP-Hard problem, we first study the problem of single-sensor active target tracking using either distance-only, bearing-only, or distance-and-bearing observations. In particular, we derive an analytical solver (or algorithm) applicable to arbitrary target motion models that guarantees finding the global minimum, for each type of measurements. We then leverage the analytical solution for the single-sensor case, relax the original multi-sensor NP-Hard problem, and apply nonlinear-Gauss-Seidel relaxation to compute an approximate solution for the multi-sensor case. Our algorithm exploits the separable feasible sets [64] typically assumed in the multi-sensor target tracking problem, has computational requirements that scale linearly in the number of robots, and significantly reduces the time needed for localizing a target as compared to that of a random motion strategy.

In Chapter 4, we investigate the problem of multi-step-ahead single-sensor active target tracking. In this case, our objective is to generate a sequence of optimal sensing locations over a finite time horizon, instead of seeking only the next best sensing location, for a tracking sensor using distance-only observations. In our problem formulation, we compensate for the increasing uncertainty on the predicted (over a finite time horizon) target-position estimates by appropriately inflating the measurement noise variances at the corresponding future time steps. In order to efficiently solve the multi-step-ahead problem, we develop a cyclic block-coordinate descent algorithm, applicable to arbitrary target motion models, to calculate an approximate solution, whose computational complexity is quadratic in the number of time steps considered. Monte Carlo simulations demonstrate that as the process noise variance reduces, the multi-step-ahead motion strategy computed based on our proposed algorithm, indeed achieves better tracking accuracy as compared to that of the one-step-ahead strategy.

An advantage of our proposed low-complexity algorithms is that they can be implemented on commercial off-the-shelf (COTS) processors, such as those found on mobile

robots and cell phones, and hence can be used as part of real-time absolute and relative positioning systems.

1.2.3 Complexity reduction in CL

In active target tracking, we solely focus on the design of optimal motion strategies for mobile sensors under the assumption that the pose of each tracking sensor is perfectly known. In Chapter 5, we relax this assumption and seek to optimize, in terms of processing requirements, existing approaches to accurate multi-robot localization.

Localization of mobile robots is typically achieved by fusing noise-corrupted measurements from on-board proprioceptive sensors (e.g., odometers and inertial measurement units [IMUs]) and exteroceptive sensors (e.g., cameras, sonars, and laser scanners). Robot localization based solely on the integration of proprioceptive measurements, also known as dead reckoning, is unreliable, due to the presence of noise and biases in the proprioceptive sensors' signals. To reduce the estimation errors, exteroceptive sensors are often used for providing additional positioning information. For example, when teams of robots operate within unknown and GPS-denied environments, each robot can use its on-board exteroceptive sensors to measure its relative distance and/or bearing to its neighboring robots. By processing all these observations, the team can jointly estimate the poses of all robots so as to improve the robots' pose estimation accuracy. This process is known as CL. Indeed, CL can be readily carried out in multi-sensor target tracking tasks, and improve their performance if each sensor uses its exteroceptive sensors to not only track the moving target, but also measure its relative position with respect to its teammates. In order, however, to take advantage of the improved positioning accuracy afforded by CL, one would need to cope with the increasing demands on both processing and communication resources. Previous work has revealed that the computational complexity of the EKF-based CL for a team of N robots, in the worst case where each robot can measure all other robots in the team, is of $\mathcal{O}(N^4)$. This high processing cost clearly prohibits real-time implementation of the EKF-based CL for large N.

Previous efforts to reduce the computational complexity of CL often introduce approximations (i.e., they ignore the cross-correlations among robots), which can lead to overly optimistic and inconsistent estimates. In contrast, in this thesis, we develop

an optimal (exact) algorithm that significantly reduces the processing requirements of the EKF-based CL by taking advantage of the previously unexploited sparsity of the measurement (Jacobian) matrix in CL. Specifically, we consider a team of N robots performing CL in a 2-D environment, when the underlying relative measurement graph is fully connected, and thus the total number of relative observations is $(N-1)N$ per time step. For this formulation, we develop an efficient numerical algorithm that reduces the processing cost of the EKF-based CL, from $\mathcal{O}(N^4)$ to $\mathcal{O}(N^3)$. The key idea behind our approach is to perform dimensionality reduction of the measurement equations by applying QR factorization on the measurement matrix $\mathbf{H}$. However, the traditional column-pivoted Householder QR decomposition algorithm, which is usually adopted in practice, inevitably destroys the sparsity of $\mathbf{H}$ and results in high processing cost of $\mathcal{O}(N^4)$. On the other hand, all existing software packages for general-purpose sparse matrix computations do not take full advantage of the special sparse pattern of $\mathbf{H}$ and requires high memory usage. To address these issues, we develop a novel Modified Householder QR algorithm, which fully exploits the sparsity of $\mathbf{H}$ and achieves QR factorization with minimum computational overhead. As a result, the EKF-based CL when employing our proposed Modified Householder QR has a proven worst-case computational complexity of $\mathcal{O}(N^3)$.

By reducing the computational complexity of the EKF-based CL by an order of magnitude, our research enables real-time implementation of CL for teams of significantly large number of robots. Furthermore, for time-critical applications, such as search and rescue missions, our work ensures that the robot localization algorithm operates efficiently in the background using only a small portion of the computational resources while achieving the maximum localization accuracy. This, in turn, enables the robot team to devote most of its processing capabilities to higher priority tasks of critical importance.

1.3 Organization of the manuscript

Chapter 2 addresses the active formation control problem, and presents the algorithm derived for global minimization of the localization uncertainty over the follower robot's sensing locations. Results from simulations demonstrate that our proposed optimized

motion strategy leads to significant localization accuracy improvement as compared to alternative approaches. Chapters 3 & 4 presents the analytical expressions for the global optimum solution of the one-step-ahead single-sensor active target tracking problem. Computationally-efficient iterative algorithms for calculating approximate solutions for (i) multi-sensor and (ii) multi-step-ahead active target tracking are described in Chapters 3 & 4, respectively. Chapter 5 presents the Modified Householder QR algorithm and analyzes its computational complexity when used for factorizing the sparse measurement matrix $\mathbf{H}$ in CL. Simulations verify the complexity reduction of the EKF-based CL, and demonstrate the superior performance of the Modified Householder QR both in terms of accuracy and CPU runtime, as compared to the current state-of-the-art sparse QR algorithm. Finally, Chapter 6 summarizes the main contributions of this dissertation and highlights future research directions.

Chapter 2

Active Formation Control

2.1 Introduction

Multiple robots are often required to move and maintain certain formations in order to successfully execute a task. For example, in military missions, aerial vehicles fly in formations for mutual defense and reconnaissance; in transportation, vehicle platooning saves fuel and increases the throughput of transportation networks [65]; and in cooperative object manipulation tasks, multiple robots form a formation in order to move a large payload [66].

Due to its broad applications, formation control has been extensively studied, and there exist various control algorithms for maintaining formation with high accuracy, such as behavior-based [67], virtual structure [68], and leader-follower-based methods [69]. All these methods inevitably rely on on-board exteroceptive sensors to provide information about the robots' relative poses (positions and orientations), so as to reduce the deviations from the desired relative poses. However, exteroceptive sensors usually do not directly measure the relative pose, but instead only measure the noise-corrupted distance or bearing from one robot to its neighbors. Hence, an estimator is necessary for estimating the relative pose by fusing inter-robot distance or bearing measurements. Clearly, regardless of the control algorithm employed for maintaining a formation, accurate relative pose estimates are necessary for successful formation control.

Interestingly, the robots' motions can significantly affect the accuracy of the relative pose estimates. In particular, early experimental results by Rekleitis *et al.* demonstrate

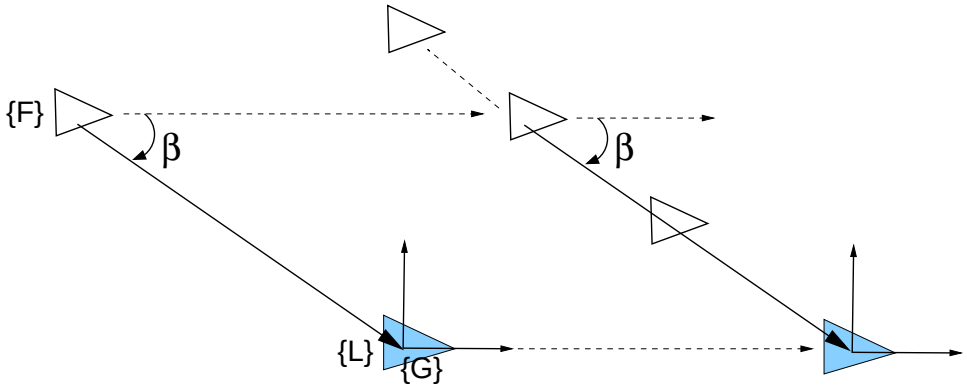


Figure 2.1: A leader, L, and a follower, F, are moving on parallel straight lines. The relative pose between the leader and follower is unobservable when the follower only measures the bearing β towards the leader, since the follower can be anywhere along the line of sight of the leader.

that when the robots move randomly, they achieve higher relative-pose estimation accuracy as compared to when maintaining an exact formation [70]. In addition, Trawny and Barfoot also confirmed that the optimal motion strategy is not to move in formations [41].

The estimation accuracy of the relative pose in robot formations degrades because the system becomes unobservable when the robots only record either relative distance or bearing measurements between each other. This is due to the fact that since their relative poses do not vary, distance or bearing measurements of a constant state do not provide sufficient information to determine the robots' relative poses (see Figure 2.1). However, the system immediately becomes observable when the robots move arbitrarily and deviate from their exact formations [60, 61, 71, 72], since extra knowledge about each robot's motion provides additional information about their relative poses. Consequently, random trajectories often result in better estimation performance than moving in and maintaining exact formations. Thus, we conclude that there is a trade-off between keeping the formation and improving the estimation accuracy.

To analyze this trade-off, we seek the optimal next-time-step follower position in a leader-follower formation by processing robot-to-robot relative bearing observations. In order to balance the requirement of moving in formation and improving the estimation

accuracy, the follower is allowed to deviate away from the desired position but within a predefined range. The specific estimator employed in this work is the extended Kalman filter (EKF), and the optimization criterion used is the weighted trace of the posterior covariance matrix. Our main contribution is to develop an algorithm for calculating the global optimal solution of this constrained non-convex optimization problem. Specifically, we firstly analytically compute all critical points (i.e., those points satisfying the Karush-Kuhn-Tucker (KKT) necessary optimality conditions), and secondly select as global minimum the one that yields the lowest objective value [73].

The remainder of this chapter is organized as follows. In Section 2.2, we describe related localization approaches in robot formation. In Section 2.3, we formulate the optimal motion strategy problem and describe our approach for computing the global optimum in Section 2.4. Simulation results for validating our proposed algorithm are provided in Section 2.5. Finally, concluding remarks are drawn in Section 2.6.

2.2 Literature review

Current research on localization in robot formations mainly focuses on designing specific formation patterns that yield the highest localization accuracy for the whole team. More specifically, Zhang *et al.* investigate the impact of the robots' spatial configuration on the localization accuracy for *static* robots that receive absolute position measurements, as well as robot-to-robot relative distance and/or bearing observations [74]. They establish the necessary and sufficient conditions for the team of robots to be completely localized, and show that the covariance matrix characterizing the localization uncertainty is a function of the relative robot poses. A robot formation that minimizes the trace of the aforementioned covariance matrix is computed through gradient descent optimization techniques. However, due to the non-convexity of the objective function, the proposed optimization algorithm can not guarantee global optimality.

Kurazume and Hirose propose a leapfrogging strategy for a team of robots comprised of one master and two slave robots [40]. In particular, the master and slave robots alternate in moving forward on straight paths toward a destination, while the slave robots act as portable landmarks for localizing the master robot. The authors study the effect of the master-slave relative position on localization accuracy, and select three

configurations that demonstrate superior performance, by numerically minimizing a weighted least squares cost function. However, the requirement of at least one slave robot to remain stationary at each time step is often undesirable.

Contrary to [40], in [41] all three robots are allowed to move continuously and simultaneously toward a target configuration while measuring relative distance to each other. In order to generate the optimal trajectories that maximize localization accuracy, the authors employ a gradient-based optimization algorithm that minimizes the determinant of the covariance matrix (associated with the localization uncertainty) at the target configuration. Even though the gradient-based algorithm can only converge to a local optimum, the numerical experiments vindicate that localization accuracy is improved when the robots deviate from exact formations.

The effects of formation geometry on the localization accuracy is addressed in [75], where the robots record both relative distance and bearing measurements. Additionally, the robots' orientations are assumed perfectly known. The optimality criterion is the steady-state position uncertainty of the robot team. A genetic algorithm is employed for determining the optimal relative robot positions. This algorithm is shown to be a suitable tool for the addressed problem due to the existence of multiple local minima of the cost function. Their results reveal that the optimal formation geometry comprises of adjacent equilateral triangles.

The works mentioned above have focused on observable systems, where the robots have sufficient measurements (e.g., relative position, orientation, and/or absolution position) for determining their relative poses. However, due to cost considerations in practice, the robots might only have access to a limited type of measurements, such as inter-robot range or bearing observations. In this case, the relative poses become unobservable if the robots move in formation on straight lines, as illustrated in Figure 2.1, and the trade-off between maintaining the formation and ensuring system observability has generally been overlooked with one exception.

Mariottini *et al.* consider the problem of loss of observability in robot formations with bearing-only measurements, and propose a switching active-sensing strategy to preserve observability and maintain the formation [61]. Specifically, a standard leader-follower controller drives the follower to its desired location (to keep exact formation) when the leader does not move on straight paths and the system is observable. When the

system becomes unobservable, the standard controller is switched to an active-formation controller that guides the follower to a position which is slightly different from the desired location in order to ensure system observability. However, no optimality, in terms of the estimation accuracy, is claimed by the proposed active controller.

Our work [73] differs from the aforementioned papers, in that our objective is to balance the trade-off between maintaining the formation and improving the estimation accuracy of the relative pose. Most importantly, our method is able to analytically calculate the next best sensing location of the follower that deviates from its desired formation within a predefined region. In contrast to [61], our optimal motion strategy not only preserves system observability, but also minimizes the uncertainty of the relative pose estimates.

2.3 Problem formulation

In a leader-follower formation, the leader moves on a desired trajectory, while the follower keeps its position $\mathbf{p} = [x\ y]^{\mathrm{T}}$ and orientation ϕ constant with respect to the leader.[1] We define the state vector as the follower's position and orientation with respect to the leader, i.e., $\mathbf{x} = [\mathbf{p}^{\mathrm{T}}\ \phi]^{\mathrm{T}} = [x\ y\ \phi]^{\mathrm{T}}$, and denote the desired follower state (i.e., desired formation vector) as $\mathbf{x}_0 = [\mathbf{p}_0^{\mathrm{T}}\ \phi_0]^{\mathrm{T}} = [x_0\ y_0\ \phi_0]^{\mathrm{T}}$. Furthermore, the follower measures the relative bearing β towards the leader (see Figure 2.2), and estimates its state by fusing these bearing measurements. Due to the trade-off between maintaining the formation and improving the estimation accuracy of the relative pose $\mathbf{x}$ as discussed before, we allow the follower's position to deviate from its desired position $\mathbf{p}_0$ within a predefined distance r. Our objective is to determine where the follower should move to at the next time step to record a relative bearing measurement, such that the estimate uncertainty of $\mathbf{x}$ is minimized.

In the following sections, we will first present the system kinematic model,[2] the measurement model, and then formulate the optimization problem that needs to be solved.

[1] In our analysis, we only consider the case with one follower. The same algorithm can be applied to the case of multiple followers.

[2] Note that our optimization algorithm can be applied to any system kinematic model. The specific one presented below is used in our simulations to evaluate the performance of our method.

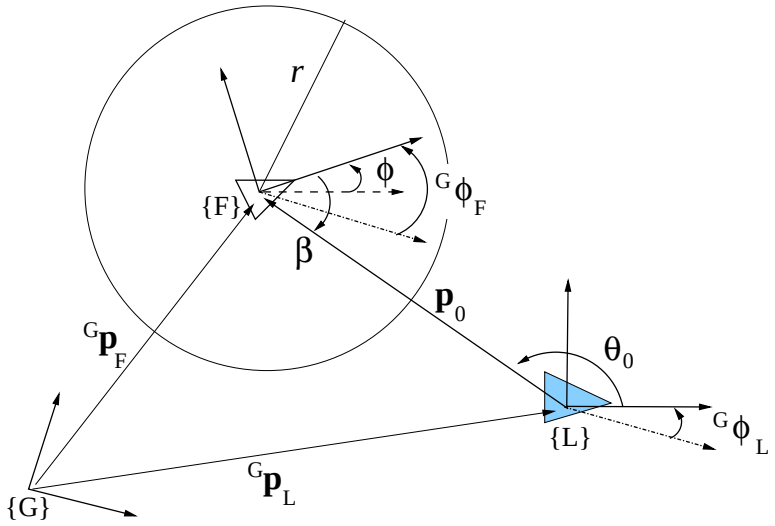


Figure 2.2: A leader-follower formation: the follower F is at the desired position $\mathbf{p}_0$ with respect to the leader L. The follower measures its bearing β to the leader. In order to preserve system observability and minimize the relative pose estimation uncertainty, the follower is allowed to move inside the circle centered at $\mathbf{p}_0$ with radius r.

2.3.1 System kinematic model

We assume that both the leader and the follower are non-holonomic vehicles whose linear and rotational velocities are (v_L, ω_L) and (v_F, ω_F), respectively. In addition, we focus on the case where the leader is moving on a straight line with constant v_L, and $\omega_L = 0$, since most likely this is the dominant motion when a team of robots is moving towards a distant location. The directions of the linear velocities are along the x-axes of the vehicles' body frames. Hence, these linear velocities expressed in a fixed global frame $\{G\}$ are described by

$$ {}^{G}\dot{\mathbf{p}}_L = {}^{G}_{L}\mathbf{C} \cdot \mathbf{e}_1 v_L, \quad \text{and} \quad {}^{G}\dot{\mathbf{p}}_F = {}^{G}_{F}\mathbf{C} \cdot \mathbf{e}_1 v_F. \tag{2.1} $$

where $\mathbf{e}_1 = [1\ 0]^{\mathrm{T}}$, ${}^{G}_{L}\mathbf{C}$ and ${}^{G}_{F}\mathbf{C}$ are the rotation matrices that project vectors expressed in frames $\{L\}$ and $\{F\}$ to frame $\{G\}$, respectively.

The rotation matrices ${}^{G}_{L}\mathbf{C}$, ${}^{G}_{F}\mathbf{C}$, and ${}^{L}_{F}\mathbf{C}$ satisfy the following geometric relationship:

$$ {}^{G}_{F}\mathbf{C}({}^{G}\phi_F) = {}^{G}_{L}\mathbf{C}({}^{G}\phi_L) \cdot {}^{L}_{F}\mathbf{C}({}^{L}\phi_F) \quad \Rightarrow \quad {}^{G}\phi_F = {}^{G}\phi_L + {}^{L}\phi_F. \tag{2.2} $$

Differentiating (2.2) with respect to time, yields the kinematic model for the orientation, i.e.,

$$\dot{\phi} = {}^{L}\dot{\phi}_F = {}^{G}\dot{\phi}_F - {}^{G}\dot{\phi}_L = \omega_F - \omega_L = \omega_F. \tag{2.3}$$

Similarly, we derive the kinematic model for the robots' relative position $\mathbf{p}$ by taking the time derivative of the following geometric constraint

$$\mathbf{p} = {}^{G}_{L}\mathbf{C}^{\mathrm{T}}({}^{G}\mathbf{p}_F - {}^{G}\mathbf{p}_L) \quad \Rightarrow \quad \dot{\mathbf{p}} = {}^{G}_{L}\mathbf{C}^{\mathrm{T}}({}^{G}\dot{\mathbf{p}}_F - {}^{G}\dot{\mathbf{p}}_L). \tag{2.4}$$

Substituting (2.1) and (2.2) in (2.4), we obtain

$$\dot{\mathbf{p}} = {}^{L}_{F}\mathbf{C} \cdot \mathbf{e}_1 v_F - \mathbf{e}_1 v_L. \tag{2.5}$$

Rearranging terms in (2.3) and (2.5), yields the following continuous-time kinematic model:

$$\begin{aligned} \dot{x} &= \cos(\phi) v_F - v_L, \\ \dot{y} &= \sin(\phi) v_F, \\ \dot{\phi} &= \omega_F. \end{aligned} \tag{2.6}$$

2.3.2 State and covariance propagation

The discrete-time kinematic equations of the follower's pose are obtained by discretizing (2.6) using Euler's forward method, i.e.,

$$\begin{aligned} x_{k+1} &= x_k + \cos(\phi_k) v_{F_k} \delta t - v_L \delta t, \\ y_{k+1} &= y_k + \sin(\phi_k) v_{F_k} \delta t, \\ \phi_{k+1} &= \phi_k + \omega_{F_k} \delta t. \end{aligned} \tag{2.7}$$

We employ the EKF to propagate the estimated follower's pose using the control inputs (odometry measurements) $v_{F_{m,k}}$ and $\omega_{F_{m,k}}$, i.e.,[3]

$$\begin{aligned} \hat{x}_{k+1|k} &= \hat{x}_{k|k} + \cos(\hat{\phi}_{k|k}) v_{F_{m,k}} \delta t - v_L \delta t, \\ \hat{y}_{k+1|k} &= \hat{y}_{k|k} + \sin(\hat{\phi}_{k|k}) v_{F_{m,k}} \delta t, \\ \hat{\phi}_{k+1|k} &= \hat{\phi}_{k|k} + \omega_{F_{m,k}} \delta t. \end{aligned} \tag{2.8}$$

[3] In the rest of the chapter, the "hat" symbol "$\hat{}$" denotes the estimated value of a quantity. The subscript $\ell|j$ refers to the estimate of the quantity at time-step ℓ after incorporating all measurements up to time-step j. The error between the actual value of the quantity and its estimate is denoted by the "tilde" symbol "$\tilde{}$". The relationship between the actual value $\mathbf{x}$ and the estimate $\hat{\mathbf{x}}$ is $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$.

By linearizing (2.8), the error-propagation equation for the follower's pose is readily derived as:

$$\begin{bmatrix}\tilde{x}_{k+1|k}\\ \tilde{y}_{k+1|k}\\ \tilde{\phi}_{k+1|k}\end{bmatrix} = \begin{bmatrix}1 & 0 & -\sin(\hat{\phi}_{k|k})v_{F_{m,k}}\delta t\\ 0 & 1 & \cos(\hat{\phi}_{k|k})v_{F_{m,k}}\delta t\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\tilde{x}_{k|k}\\ \tilde{y}_{k|k}\\ \tilde{\phi}_{k|k}\end{bmatrix} + \begin{bmatrix}\cos(\hat{\phi}_{k|k})\delta t & 0\\ \sin(\hat{\phi}_{k|k})\delta t & 0\\ 0 & \delta t\end{bmatrix}\begin{bmatrix}w_{v_{F_k}}\\ w_{\omega_{F_k}}\end{bmatrix}$$

$$\Leftrightarrow \quad \tilde{\mathbf{x}}_{k+1|k} = \mathbf{\Phi}_k\tilde{\mathbf{x}}_{k|k} + \mathbf{G}_k\mathbf{w}_k. \tag{2.9}$$

where $w_{v_{F_k}}$ and $w_{\omega_{F_k}}$ are white, zero-mean, Gaussian noise sequences with variances $\sigma^2_{v_F}$ and $\sigma^2_{\omega_F}$, affecting the linear and rotational velocity inputs, respectively.

From equation (2.9), the error-state covariance is propagated as:

$$\mathbf{P}_{k+1|k} = \mathbf{\Phi}_k\mathbf{P}_{k|k}\mathbf{\Phi}_k^{\mathrm{T}} + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^{\mathrm{T}}, \tag{2.10}$$

where $\mathbf{Q}_k = \mathrm{diag}(\sigma^2_{v_F}, \sigma^2_{\omega_F})$.

2.3.3 Measurement update

The follower employs the inter-robot relative bearing measurements to perform pose updates in the EKF. The relative bearing measurement at time step k can be described by the following nonlinear model (see Figure 2.2)

$$z_{k+1} = \beta_{k+1} + n_{\beta_{k+1}} = \theta_{k+1} - \phi_{k+1} + \pi + n_{\beta_{k+1}} = \arctan\left(\frac{y_{k+1}}{x_{k+1}}\right) - \phi_{k+1} + \pi + n_{\beta_{k+1}},$$

where $n_{\beta_{k+1}}$ is zero-mean Gaussian noise with variance σ^2_β.

In the EKF, we employ linearization to obtain the measurement error equation, i.e.,

$$\tilde{z}_{k+1|k} = \mathbf{h}^{\mathrm{T}}_{\beta_{k+1}}\tilde{\mathbf{x}}_{k+1|k} + n_{\beta_{k+1}},$$

where the measurement (Jacobian) matrix $\mathbf{h}_{\beta_{k+1}}$, a column vector of dimension 3, has the following structure:

$$\mathbf{h}_{\beta_{k+1}} = \begin{bmatrix}\frac{(\mathbf{J}\hat{\mathbf{p}}_{k+1|k})^{\mathrm{T}}}{\hat{\mathbf{p}}^{\mathrm{T}}_{k+1|k}\hat{\mathbf{p}}_{k+1|k}} & -1\end{bmatrix}^{\mathrm{T}} = \frac{1}{\hat{d}_{k+1|k}}\begin{bmatrix}-\sin(\hat{\theta}_{k+1|k}) & \cos(\hat{\theta}_{k+1|k}) & -\hat{d}_{k+1|k}\end{bmatrix}, \tag{2.11}$$

where the predicted relative position $\hat{\mathbf{p}}_{k+1|k}$ at time-step $k+1$ is defined as $\hat{\mathbf{p}}_{k+1|k} = [\hat{x}_{k+1|k}\ \hat{y}_{k+1|k}]^{\mathrm{T}}$, and $\hat{d}_{k+1|k} = \|\hat{\mathbf{p}}_{k+1|k}\|_2$, $\hat{\theta}_{k+1|k} = \arctan\left(\frac{\hat{y}_{k+1|k}}{\hat{x}_{k+1|k}}\right)$ [see Figure 2.3]. The

constant matrix $\mathbf{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$. Note that the measurement matrix $\mathbf{h}_{\beta_{k+1}}$ is time varying and depends on the relative position $\hat{\mathbf{p}}_{k+1|k}$ [see (2.11)]. Thus, the location where the robot records the bearing measurement will affect the accuracy of the state estimates.

Once a relative bearing measurement z_{k+1} becomes available, the state estimate and its covariance can be updated as

$$\begin{aligned} \hat{\mathbf{x}}_{k+1|k+1} &= \hat{\mathbf{x}}_{k+1|k} + \frac{\mathbf{P}_{k+1|k}\mathbf{h}_{\beta_{k+1}}}{\mathbf{h}_{\beta_{k+1}}^{\mathrm{T}}\mathbf{P}_{k+1|k}\mathbf{h}_{\beta_{k+1}} + \sigma_{\beta}^{2}}\tilde{\mathbf{z}}_{k+1|k}, \\ \mathbf{P}_{k+1|k+1} &= \mathbf{P}_{k+1|k} - \frac{\mathbf{P}_{k+1|k}\mathbf{h}_{\beta_{k+1}}\mathbf{h}_{\beta_{k+1}}^{\mathrm{T}}\mathbf{P}_{k+1|k}}{\mathbf{h}_{\beta_{k+1}}^{\mathrm{T}}\mathbf{P}_{k+1|k}\mathbf{h}_{\beta_{k+1}} + \sigma_{\beta}^{2}}. \end{aligned} \tag{2.12}$$

It is evident from (2.12) that the posterior covariance $\mathbf{P}_{k+1|k+1}$, indicating the accuracy of the estimate of relative pose $\hat{\mathbf{x}}_{k+1|k+1}$, is an explicit function of $\hat{\mathbf{p}}_{k+1|k}$ through the measurement matrix $\mathbf{h}_{\beta_{k+1}}$. Therefore it is possible to optimize the follower's motion to achieve higher estimation accuracy. In the next section, we will formulate the follower's motion strategy as an optimization problem. The optimization variable is the predicted relative position $\hat{\mathbf{p}}_{k+1|k} = [\hat{x}_{k+1|k}\ \hat{y}_{k+1|k}]^{\mathrm{T}}$, or equivalently, its polar coordinate representation $(\hat{d}_{k+1|k}, \hat{\theta}_{k+1|k})$ [see Figure 2.3].

2.3.4 Optimization problem formulation

The cost function is defined as the weighted sum of the diagonal elements of the posterior covariance $\mathbf{P}_{k+1|k+1}$, i.e., $\mathrm{tr}(\mathbf{W}\mathbf{P}_{k+1|k+1}\mathbf{W})$, where the weighting matrix $\mathbf{W} = \mathrm{diag}(1, 1, d_0)$ with $d_0 = \|\mathbf{p}_0\|_2$.

Since $\mathbf{\Phi}_k$ [see (2.9)] is itself a function of the control input $v_{F_{m,k}}$, the prior covariance $\mathbf{P}_{k+1|k}$ [see (2.10)] is also a function of the optimization variable $\hat{\mathbf{p}}_{k+1|k}$, which makes solving the optimization problem very challenging. However, since the robots are required to move in formation, the follower can only deviate at most from its nominal position $\mathbf{p}_0$ by a small distance r. Therefore we approximate $\mathbf{P}_{k+1|k}$ by a constant matrix $\mathbf{P}$ which is chosen to be the prior covariance when the follower moves to its desired position $\mathbf{p}_0$. To simplify notation, in the rest of the chapter we will drop the subscripts for the state, covariance, and measurement matrix, when the meaning of each quantity is clear in the context, e.g., $\mathbf{p} = \hat{\mathbf{p}}_{k+1|k}$, $\mathbf{P} = \mathbf{P}_{k+1|k}$ and $\mathbf{h}_{\beta} = \mathbf{h}_{\beta_{k+1}}$.

Since $\mathbf{P}_{k+1|k}$ is approximated by a constant matrix $\mathbf{P}$, based on (2.12), the objective function can be simplified by

$$\arg\min_{\mathbf{p}} \operatorname{tr}\left(\mathbf{W}\mathbf{P}_{k+1|k+1}\mathbf{W}\right) = \arg\min_{\mathbf{p}} \; -\frac{\operatorname{tr}\left(\mathbf{P}\mathbf{h}_\beta\mathbf{h}_\beta^{\mathrm{T}}\mathbf{P}\right)}{\mathbf{h}_\beta^{\mathrm{T}}\mathbf{P}\mathbf{h}_\beta+\sigma_\beta^2} = \arg\min_{\mathbf{p}} \; -\frac{\mathbf{h}_\beta^{\mathrm{T}}\mathbf{P}^2\mathbf{h}_\beta}{\mathbf{h}_\beta^{\mathrm{T}}\mathbf{P}\mathbf{h}_\beta+\sigma_\beta^2}.$$

Defining $f_\beta = -\frac{\mathbf{h}_\beta^{\mathrm{T}}\mathbf{P}^2\mathbf{h}_\beta}{\mathbf{h}_\beta^{\mathrm{T}}\mathbf{P}\mathbf{h}_\beta+\sigma_\beta^2}$, we formulate the formation optimization problem as

- Optimization Problem 1 (Π_1)

$$\begin{aligned} \underset{\mathbf{p}}{\text{minimize}} \quad & f_\beta(\mathbf{p}) = -\frac{\mathbf{h}_\beta^{\mathrm{T}}\mathbf{P}^2\mathbf{h}_\beta}{\mathbf{h}_\beta^{\mathrm{T}}\mathbf{P}\mathbf{h}_\beta+\sigma_\beta^2} \\ \text{subject to} \quad & g_\beta(\mathbf{p}) = \|\mathbf{p}-\mathbf{p}_0\|_2^2 - r^2 \le 0. \end{aligned} \tag{2.13}$$

Remark 3. *The optimization problem* Π_1 *is a constrained nonlinear programming problem since both the objective function* f_β *and the constraint* g_β *are nonlinear functions of the optimization variable* $\mathbf{p}$*. Furthermore, although the feasible set of* Π_1 *defined by the inequality* $g_\beta(\mathbf{p}) \le 0$ *is convex, the optimization problem* Π_1 *is not a convex problem, because the objective function* f_β *is not convex in* $\mathbf{p}$*.*

2.4 Solution strategy

As mentioned in the previous section, the optimization problem Π_1 is *not convex*, and thus it can have multiple local minima. One approach is to discretize the feasible set and find the point which yields the minimum value for the function f_β. However, such an approach provides no guarantees for finding the global optimum, due to limitations of the grid resolution.

Instead, we seek the global optimal solution as follows: We first analytically determine *all critical/stationary points* (i.e., those points which satisfy the Karush-Kuhn-Tucker (KKT) necessary optimality conditions [64, Ch. 3]) and evaluate their objective values. Then, as optimal solution, we select the critical point whose objective value is the smallest.

Since the measurement matrix $\mathbf{h}_\beta$ [see (2.11)] depends explicitly on both θ and d, we express the optimization variable $\mathbf{p}$ using its polar coordinates, i.e., $\mathbf{p} = d[\cos(\theta)\ \sin(\theta)]^{\mathrm{T}}$

(see Figure 2.3), and rewrite the optimization problem Π_1 in polar coordinates, resulting in the following equivalent optimization problem Π_2:

- Optimization Problem 2 (Π_2)

$$\underset{d,\theta}{\text{minimize}} \quad f_\beta(d,\theta) = -\frac{c_u\, d^{\,2} + 2\,\mathbf{b}_u^{\mathrm{T}}\,\boldsymbol{\alpha}\, d + \boldsymbol{\alpha}^{\mathrm{T}}\mathbf{A}_u\boldsymbol{\alpha}}{c_v\, d^{\,2} + 2\,\mathbf{b}_v^{\mathrm{T}}\,\boldsymbol{\alpha}\, d + \boldsymbol{\alpha}^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}} \tag{2.14}$$

$$\text{subject to} \quad g_\beta(d,\theta) = (d\cos(\theta) - x_0)^2 + (d\sin(\theta) - y_0)^2 - r^2 \leq 0, \tag{2.15}$$

where $\boldsymbol{\alpha} = [-\sin(\theta) \;\; \cos(\theta)]^{\mathrm{T}}$ and $\mathbf{A}_u, \mathbf{b}_u, c_u, \mathbf{A}_v, \mathbf{b}_v$, and c_v are known parameters expressed in terms of $\mathbf{P}$ (for simplicity, the prior covariance matrix $\mathbf{P}$ is partitioned as: $\mathbf{P} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{12}^{\mathrm{T}} & \mathbf{P}_{22} \end{bmatrix}$, where $\mathbf{P}_{11}$ has dimensions 2×2), d_0, and σ_β^2:

$$\begin{aligned}
\mathbf{A}_u &= \mathbf{P}_{11}^2 + d_0^2 \mathbf{P}_{12}\mathbf{P}_{12}^{\mathrm{T}}, \\
\mathbf{b}_u &= -\left(\mathbf{P}_{11} + d_0^2 \mathbf{P}_{22}\mathbf{I}_2\right)\mathbf{P}_{12}, \\
c_u &= \mathbf{P}_{12}^{\mathrm{T}}\mathbf{P}_{12} + d_0^2 \mathbf{P}_{22}^2, \\
\mathbf{A}_v &= \mathbf{P}_{11}, \\
\mathbf{b}_v &= -\mathbf{P}_{12}, \\
c_v &= \mathbf{P}_{22} + \sigma_\beta^2.
\end{aligned}$$

To solve Π_2, we firstly analytically compute *all critical/stationary points* by solving systems of polynomial equations using an elimination procedure. To proceed, we first construct the Lagrange function [64]

$$\mathcal{L}(d,\theta,\lambda) = f_\beta(d,\theta) + \lambda\, g_\beta(d,\theta).$$

Based on the KKT necessary conditions, the critical points d^*, θ^*, and the associated Lagrange multiplier λ^* must satisfy

$$\nabla f_\beta(d^*,\theta^*) + \lambda^* \nabla g_\beta(d^*,\theta^*) = \mathbf{0}_{2\times 1}, \tag{2.16}$$

$$\lambda^* g_\beta(d^*,\theta^*) = 0, \tag{2.17}$$

$$\lambda^* \geq 0, \tag{2.18}$$

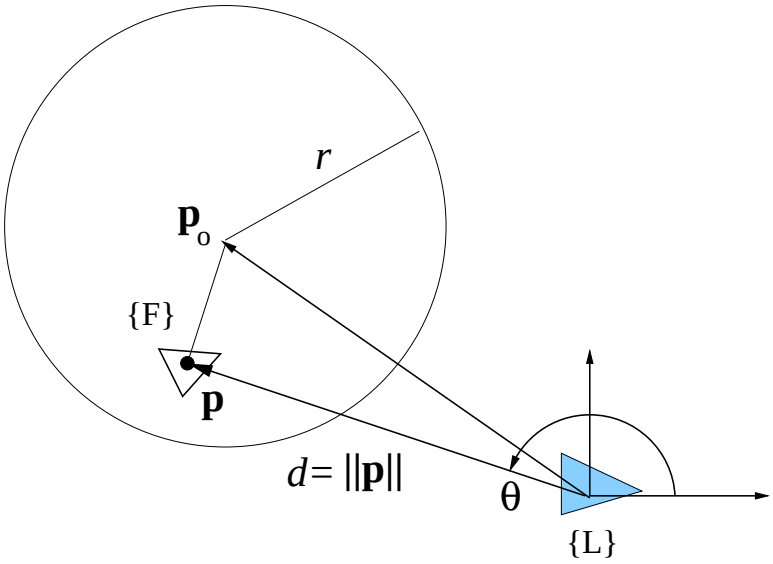


Figure 2.3: The measurement Jacobian $\mathbf{h}_\beta$ is a function of d and θ, which represent the polar coordinates of the predicted relative position $\mathbf{p}$.

where $\nabla f_\beta = [\nabla_d f_\beta \ \nabla_\theta f_\beta]^{\mathrm{T}}$ and $\nabla g_\beta = [\nabla_d g_\beta \ \nabla_\theta g_\beta]^{\mathrm{T}}$.

In what follows, we will examine the properties of the derivatives ∇f_β and ∇g_β. From (2.14)-(2.15), it is straightforward to see that the objective function f_β is a *rational* function with respect to d^*, $\cos(\theta^*)$, and $\sin(\theta^*)$, while the constraint g_β is a *polynomial* in those three variables. Since the derivative of a polynomial (rational) is still a polynomial (rational), we conclude that the derivatives ∇f_β and ∇g_β are *rational* and *polynomial* functions of d^*, $\cos(\theta^*)$, and $\sin(\theta^*)$, respectively.

Thus, (2.16) can be transformed into a *polynomial* equation in d^*, $\cos(\theta^*)$, and $\sin(\theta^*)$ by requiring the numerator to be equal to zero. Therefore, computing all critical points of Π_2 is equivalent to solving the polynomial system defined by (2.16)-(2.17).

In what follows, we examine two cases: (i) the constraint (2.15) is *inactive*, i.e., $g_\beta(d^*, \theta^*) < 0$; (ii) the constraint is *active*, i.e., $g_\beta(d^*, \theta^*) = 0$, and determine the global optimal solution by searching for the critical point that has the smallest cost.

2.4.1 Inactive constraint

We first compute the set of all stationary points, denoted as $\mathbf{\Xi}_u$, by assuming that the constraint is inactive, or equivalently,

$$g_\beta(d^*, \theta^*) < 0, \quad \forall \ (d^*, \theta^*) \in \mathbf{\Xi}_u. \tag{2.19}$$

Combining (2.19) and the complementary slackness condition (2.17), yields $\lambda^* = 0$. Hence, the term $\lambda^* \nabla g_\beta(d^*, \theta^*)$ in (2.16) vanishes, and (2.16) and (2.17) are simplified into

$$\nabla f_\beta(d^*, \theta^*) = \begin{bmatrix} \nabla_d f_\beta(d^*, \theta^*) \\ \nabla_\theta f_\beta(d^*, \theta^*) \end{bmatrix} = \begin{bmatrix} \frac{-2\mathcal{F}_d(d^*, \theta^*)}{(c_v\,(d^*)^{\,2} + 2\,\mathbf{b}_v^{\mathrm{T}}\,\boldsymbol{\alpha}^*\,d^* + (\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}^*)^2} \\ \frac{-2\mathcal{F}_\theta(d^*, \theta^*)}{(c_v\,(d^*)^{\,2} + 2\,\mathbf{b}_v^{\mathrm{T}}\,\boldsymbol{\alpha}^*\,d^* + (\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}^*)^2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{2.20}$$

where $\boldsymbol{\alpha}^* = \begin{bmatrix} -\sin(\theta^*) & \cos(\theta^*) \end{bmatrix}^{\mathrm{T}}$. Furthermore, we denote $\boldsymbol{\alpha}_c^* = -\begin{bmatrix} \cos(\theta^*) & \sin(\theta^*) \end{bmatrix}^{\mathrm{T}}$. The expressions for $\mathcal{F}_d(d^*, \theta^*)$ and $\mathcal{F}_\theta(d^*, \theta^*)$, after algebraic manipulations and simplifications, are provided as follows:

$$\mathcal{F}_d = \zeta_2 (d^*)^2 + \zeta_1 d^* + \zeta_0, \tag{2.21}$$

$$\mathcal{F}_\theta = \gamma_3 (d^*)^3 + \gamma_2 (d^*)^2 + \gamma_1 d^* + \gamma_0, \tag{2.22}$$

where the coefficients ζ_i, $i = 0, 1, 2$ and γ_j, $j = 0, \ldots, 3$ are bivariate polynomials in $\cos(\theta^*)$ and $\sin(\theta^*)$, whose explicit expressions are provided as follows:

$$\begin{aligned}
\zeta_2(\cos(\theta^*), \sin(\theta^*)) &= c_v(\mathbf{b}_u^{\mathrm{T}}\boldsymbol{\alpha}^*) - c_u(\mathbf{b}_v^{\mathrm{T}}\boldsymbol{\alpha}^*), \\
\zeta_1(\cos(\theta^*), \sin(\theta^*)) &= c_u(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}^* - c_v(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_u\boldsymbol{\alpha}^*, \\
\zeta_0(\cos(\theta^*), \sin(\theta^*)) &= (\mathbf{b}_v^{\mathrm{T}}\boldsymbol{\alpha}^*)(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_u\boldsymbol{\alpha}^* - (\mathbf{b}_u^{\mathrm{T}}\boldsymbol{\alpha}^*)(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}^*, \\
\gamma_3(\cos(\theta^*), \sin(\theta^*)) &= c_u(\mathbf{b}_v^{\mathrm{T}}\boldsymbol{\alpha}_c^*) - c_v(\mathbf{b}_u^{\mathrm{T}}\boldsymbol{\alpha}_c^*), \\
\gamma_2(\cos(\theta^*), \sin(\theta^*)) &= 2(\mathbf{b}_u^{\mathrm{T}}\boldsymbol{\alpha}_c^*)(\mathbf{b}_v^{\mathrm{T}}\boldsymbol{\alpha}^*) - 2(\mathbf{b}_v^{\mathrm{T}}\boldsymbol{\alpha}_c^*)(\mathbf{b}_u^{\mathrm{T}}\boldsymbol{\alpha}^*), \\
&\quad + c_v(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_u\boldsymbol{\alpha}_c^* - c_u(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}_c^*, \\
\gamma_1(\cos(\theta^*), \sin(\theta^*)) &= 2(\mathbf{b}_u^{\mathrm{T}}\boldsymbol{\alpha}^*)(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}_c^* - 2(\mathbf{b}_v^{\mathrm{T}}\boldsymbol{\alpha}^*)(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_u\boldsymbol{\alpha}_c^*, \\
&\quad + (\mathbf{b}_v^{\mathrm{T}}\boldsymbol{\alpha}_c^*)(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_u\boldsymbol{\alpha}^* - (\mathbf{b}_u^{\mathrm{T}}\boldsymbol{\alpha}_c^*)(\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}^*, \\
\gamma_0(\cos(\theta^*), \sin(\theta^*)) &= \left((\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_u\boldsymbol{\alpha}_c^*\right)\left((\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}^*\right) - \left((\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_v\boldsymbol{\alpha}_c^*\right)\left((\boldsymbol{\alpha}^*)^{\mathrm{T}}\mathbf{A}_u\boldsymbol{\alpha}^*\right).
\end{aligned}$$

Note that both denominators of $\nabla_d f_\beta(d^*, \theta^*)$ and $\nabla_\theta f_\beta(d^*, \theta^*)$ are positive, hence (2.20) is equivalent to a system of two multivariate polynomial equations $\mathcal{F}_d(d^*, \theta^*) = 0$ and $\mathcal{F}_\theta(d^*, \theta^*) = 0$. The first polynomial $\mathcal{F}_d$ [see (2.21)] has degree 3; whereas the second polynomial $\mathcal{F}_\theta$ [see (2.22)] is of degree 4.

In order to solve $\mathcal{F}_d = \mathcal{F}_\theta = 0$, we first treat $\cos(\theta^*)$ and $\sin(\theta^*)$ as parameters (or equivalently, treat $\zeta_i,\ i = 0, 1, 2$ and $\gamma_j,\ j = 0, \ldots, 3$ as constants) and eliminate the variable d^* from (2.21)-(2.22) using the Sylvester resultant [76, Ch. 3]. The Sylvester matrix of $\mathcal{F}_d$ and $\mathcal{F}_\theta$ with respect to d^*, denoted as $\mathbf{Syl}(\mathcal{F}_d, \mathcal{F}_\theta; d^*)$, is the following 5×5 matrix

$$\mathbf{Syl}(\mathcal{F}_d, \mathcal{F}_\theta; d^*) = \begin{bmatrix} \zeta_2 & & & \gamma_3 & \\ \zeta_1 & \zeta_2 & & \gamma_2 & \gamma_3 \\ \zeta_0 & \zeta_1 & \zeta_2 & \gamma_1 & \gamma_2 \\ & \zeta_0 & \zeta_1 & \gamma_0 & \gamma_1 \\ & & \zeta_0 & & \gamma_0 \end{bmatrix}.$$

The resultant of $\mathcal{F}_d$ and $\mathcal{F}_\theta$ with respect to d^*, denoted as $\mathbf{Res}(\mathcal{F}_d, \mathcal{F}_\theta; d^*)$, is the determinant of the Sylvester matrix $\mathbf{Syl}(\mathcal{F}_d, \mathcal{F}_\theta; d^*)$. Furthermore, since $\zeta_i,\ i = 0, 1, 2$ and $\gamma_j,\ j = 0, \ldots, 3$ are bivariate polynomials of $\cos(\theta^*)$ and $\sin(\theta^*)$, $\mathbf{Res}(\mathcal{F}_d, \mathcal{F}_\theta; d^*)$ is also a bivariate polynomial of $\cos(\theta^*)$ and $\sin(\theta^*)$:

$$\mathcal{R}_u\left(\cos(\theta^*), \sin(\theta^*)\right) = \mathbf{Res}(\mathcal{F}_d, \mathcal{F}_\theta; d^*) = 0\ . \tag{2.23}$$

Together with the trigonometric identity

$$\mathcal{I} = \cos^2(\theta^*) + \sin^2(\theta^*) - 1 = 0\ , \tag{2.24}$$

we employ the Sylvester resultant again to eliminate one of the variables, e.g., $\cos(\theta^*)$, to obtain a univariate polynomial in $\sin(\theta^*)$:

$$\mathcal{S}_u = \mathbf{Res}\left(\mathcal{R}_u, \mathcal{I}; \cos(\theta^*)\right) = \sum_{j=0}^{11} \delta_j \sin^{2j}(\theta^*) = 0, \tag{2.25}$$

where $\delta_j,\ j = 0, \ldots, 11$ are known coefficients expressed in terms of $\mathbf{P}$, d_0, and σ_β^2. Note that (2.25) does not contain odd-degree terms of $\sin(\theta^*)$. Therefore, (2.25) is equivalent

to the following eleventh-order univariate polynomial in the variable $s = \sin^2(\theta^*)$:

$$\mathcal{S}_u(s) = \sum_{j=0}^{11} \delta_j s^j = 0. \tag{2.26}$$

The roots of the univariate polynomial $\mathcal{S}_u(s)$ correspond to the 11 eigenvalues of the associated 11×11 companion matrix $\mathbf{\Delta}$ [77]:

$$\mathbf{\Delta} = \begin{bmatrix} 0 & & & -\delta_0/\delta_{11} \\ 1 & 0 & & -\delta_1/\delta_{11} \\ & \ddots & & \vdots \\ & & 1 & -\delta_{10}/\delta_{11} \end{bmatrix}.$$

Note that since $s = \sin^2(\theta^*)$, we only need to consider the real solutions between 0 and 1 of (2.26). Once s is determined, both $\cos(\theta^*)$ and $\sin(\theta^*)$ can be computed using the trigonometric identity, which can have at most 4 solutions for θ^*. Finally, for each θ^*, we compute the coefficients $\zeta_j,\ j = 0, 1, 2$ [see (2.21)], and solve for d^* from (2.21), which can have at most 2 solutions. Thus, the set $\mathbf{\Xi}_u$ consisting of all critical points (d^*, θ^*), has at most 88 elements. Furthermore, we only need to consider those critical points satisfying the constraint $g(d^*, \theta^*) \leq 0$.

The next step is to evaluate the objective function $f_\beta(d, \theta)$ [see (2.14)] at all the points in $\mathbf{\Xi}_u$ and select the one with the smallest objective value as the global optimal solution of Π_2, for the case when the constraint is inactive. We refer to this optimal solution as (d_u^*, θ_u^*).

2.4.2 Active constraint

When the constraint is active, i.e., $g_\beta(d^*, \theta^*) = 0$, the complementary slackness condition (2.17) is automatically satisfied. Hence, (2.16) and (2.17) are simplified into

$$\nabla f_\beta(d^*, \theta^*) + \lambda^* \nabla g_\beta(d^*, \theta^*) = \mathbf{0}_{2\times 1}, \tag{2.27}$$

$$g_\beta(d^*, \theta^*) = 0. \tag{2.28}$$

We solve this system of equations by an elimination procedure similar to the inactive constraint case. The difference is that an additional variable λ^* needs to be eliminated

first. In order to do so, we multiply both sides of (2.27) with $(\mathbf{J}\nabla g_\beta(d^*, \theta^*))^{\mathrm{T}}$, where $\mathbf{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$, and obtain a polynomial in only d^*, $\cos(\theta^*)$, and $\sin(\theta^*)$:

$$\mathcal{Q}(d^*, \cos(\theta^*), \sin(\theta^*)) = 0. \tag{2.29}$$

Combining (2.29) with (2.28) and the trigonometric identity (2.24), we conclude that the set of all stationary points, denoted as $\mathbf{\Xi}_c$, must satisfy

$$\mathcal{Q}(d^*, \cos(\theta^*), \sin(\theta^*)) = 0, \tag{2.30}$$

$$g_\beta(d^*, \cos(\theta^*), \sin(\theta^*)) = 0, \tag{2.31}$$

$$\mathcal{I}(\cos(\theta^*), \sin(\theta^*)) = 0. \tag{2.32}$$

To solve the above system of polynomial equations, we employ the same strategy outlined in the inactive constraint case. Firstly, we use the Sylvester resultant to eliminate d^* from (2.30) and (2.31) and obtain an eighth-order bivariate polynomial $\mathcal{R}_c(\cos(\theta^*), \sin(\theta^*))$. Secondly, we eliminate $\cos(\theta^*)$ from $\mathcal{R}_c$ and $\mathcal{I}$, and arrive at a univariate polynomial in $\sin(\theta^*)$:

$$\mathcal{S}_c\left(\sin(\theta^*)\right) = \sum_{j=0}^{8} \eta_j \sin^{2j}(\theta^*) = 0, \tag{2.33}$$

where $\eta_j,\ j = 0, \ldots, 8$, are known coefficients expressed in terms of $\mathbf{P}$, $\mathbf{p}_0$, r, d_0, and σ_β^2. Since only even-degree terms appear in the above equation, it is equivalent to the following eighth-order univariate polynomial in $s = \sin^2(\theta^*)$:

$$\mathcal{S}_c(s) = \sum_{j=0}^{8} \eta_j s^j = 0. \tag{2.34}$$

Similar to the inactive constraint case, the roots of (2.34) can be calculated by computing the eigenvalues of its associated 8×8 companion matrix. Once s is determined, both $\cos(\theta^*)$ and $\sin(\theta^*)$ can be computed, which can have at most 4 solutions for θ^*. Finally, for each θ^*, we solve for d^* from (2.28), which can have at most 2 solutions. Thus, the set $\mathbf{\Xi}_c$ consisting of all critical points (d^*, θ^*), has at most 64 elements.

The final step is to evaluate the objective function $f_\beta(d, \theta)$ [see (2.14)] at all the critical points in $\mathbf{\Xi}_c$ and select the one with the smallest objective value as the global optimal solution of Π_2, for the case when the constraint is active. We refer to this optimal solution as (d_c^*, θ_c^*).

2.4.3 Globally optimal solution

Finally, the globally minimal solution $(d_{\text{opt}}, \theta_{\text{opt}})$ for Π_2 can be selected as

$$(d_{\text{opt}}, \theta_{\text{opt}}) = \begin{cases} (d_u^*, \theta_u^*) & : \quad f_\beta(d_u^*, \theta_u^*) \leq f_\beta(d_c^*, \theta_c^*) \\ (d_c^*, \theta_c^*) & : \quad f_\beta(d_u^*, \theta_u^*) > f_\beta(d_c^*, \theta_c^*) \end{cases}$$

and the optimal predicted relative position (or equivalently, the next-best sensing location for the follower) is $\mathbf{p}_{\text{opt}} = \begin{bmatrix} d_{\text{opt}} \cos(\theta_{\text{opt}}) & d_{\text{opt}} \sin(\theta_{\text{opt}}) \end{bmatrix}^{\text{T}}$.

2.5 Simulation results

We have evaluated the performance of our motion strategy for a leader-follower formation in simulation. In particular, we consider the case where the leader moves on a straight line and the follower measuring the relative bearing towards the leader is supposed to maintain its position at $\mathbf{p}_0$ with respect to the leader. We have compared our motion strategy, the relaxed algebraic method (RAM), with four other strategies; namely (i) the "maintaining the formation" (MTF) strategy; (ii) a constrained random motion (CRM); (iii) the active control strategy of Mariottini *et al.* (MME) [61]; and (iv) grid-based search (GBS). The strategy MTF always moves the follower toward the desired position $\mathbf{p}_0$, while CRM moves the follower to a random position inside a circle centered at $\mathbf{p}_0$ with radius r. For GBS, the area inside the circle is discretized into cells, and the follower moves to the cell that has the lowest cost. GBS should be a benchmark for evaluating the performance of all motion strategies, if the discrete cell size is sufficiently small.

We have conducted Monte Carlo simulations with 50 trials with the following settings. The desired follower position is $\mathbf{p}_0 = [-1 \ 2]^{\text{T}}$. The radius of the constrained circle is set to $r = 0.1\|\mathbf{p}_0\|_2$. The initial estimated pose of the follower is $\hat{\mathbf{x}}_{1|1} = [\mathbf{p}_0^{\text{T}} \ 0]^{\text{T}}$, with covariance $\mathbf{P}_{1|1} = \text{diag}(0.1, \ 0.1, \ 4 \times 10^{-4})$. We use the discrete-time kinematic model (2.7) with $\delta t = 0.05$ s to simulate the leader and follower motions. The leader's linear velocity is set to 1 m/s, with rotational velocity 0 rad/s. The follower's control inputs, the linear and rotational velocities v_{F_m} and ω_{F_m}, are generated according to the next time-step position $\mathbf{p}$ determined by the specific motion strategy. In order to drive to $\mathbf{p}$, the follower first rotates with velocity ω_{F_m}, such that its heading points to

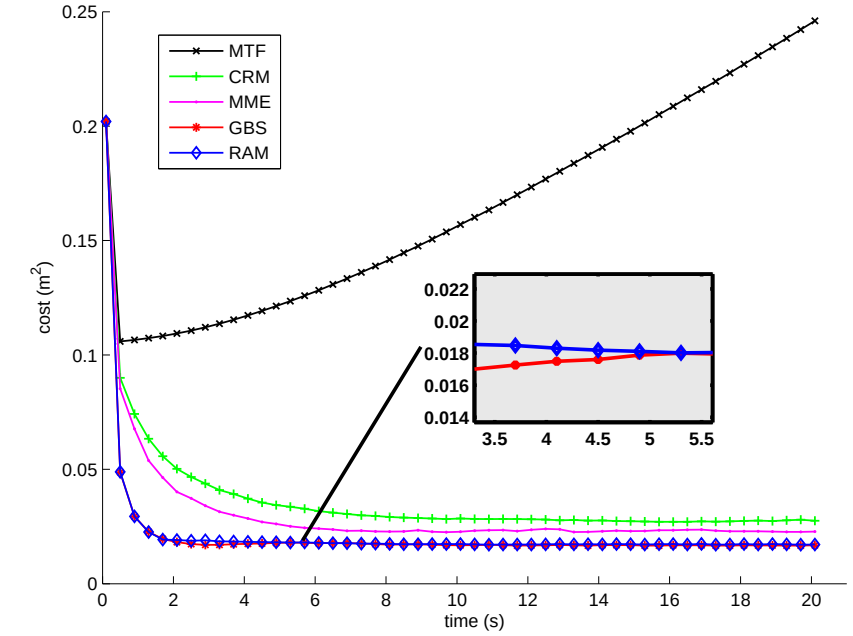


Figure 2.4: [Monte Carlo simulations] Average weighted trace of the follower's posterior covariance matrix in 50 experiments.

p. Then it executes a pure translation with velocity v_{F_m} to finally arrive at **p**. The true follower's linear and rotational velocities are affected by zero-mean white Gaussian noise with standard deviation $\sigma_{v_F} = 0.2$ m/s and $\sigma_{\omega_F} = 0.1$ rad/s, respectively. At **p**, the follower records a bearing measurement to the leader and updates its pose estimate. The standard deviation of the bearing measurement noise is set to $\sigma_\beta = 0.02$ rad.

The time evolution of the weighted trace of the follower's posterior covariance matrix (i.e., the objective function) averaged over 50 experiments is shown in Figure 2.4. As expected, the performance of RAM and GBS is improved compared to the cases of CRM and MME, and is significantly better than that of the unobservable case MTF. Additionally, the uncertainty in the follower's pose estimates (weighted trace of the covariance matrix) achieved by the proposed RAM motion strategy is indistinguishable of that of the GBS.

Figure 2.5 shows the robots' trajectories of RAM, GBS, and MME. The trajectories

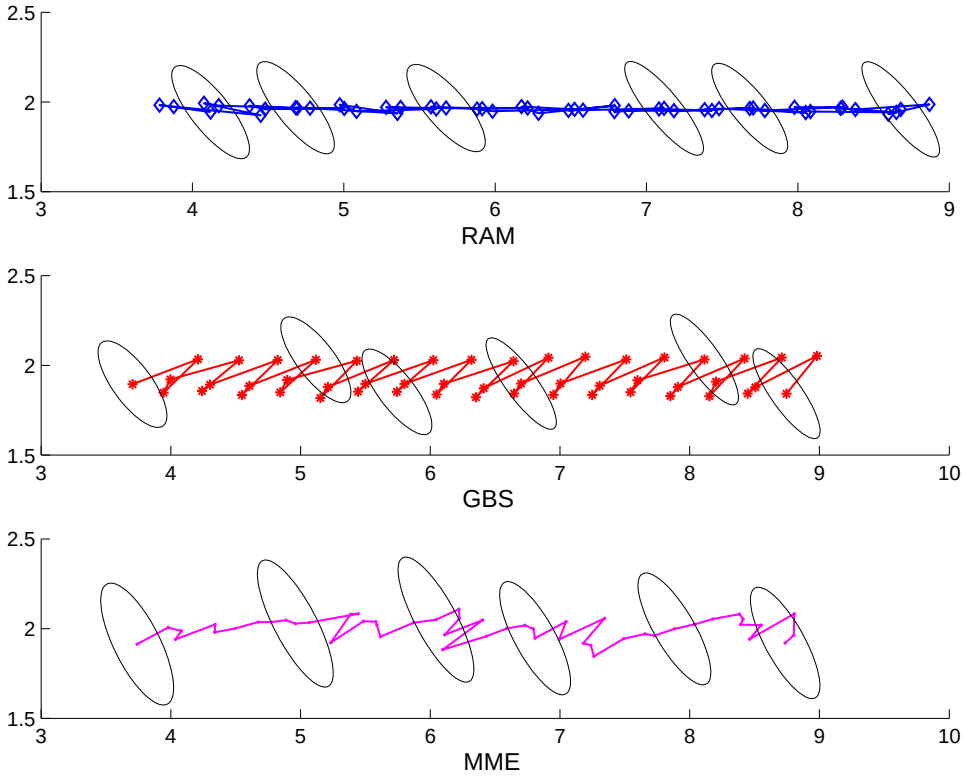


Figure 2.5: The estimated trajectories of the follower and the leader, when employing RAM, GBS, or MME as motion strategy. The ellipses denote the 3σ bounds for the follower's position uncertainty at the corresponding time steps.

of MTF and CRM are not informative and therefore are omitted. RAM generates a zigzagging motion pattern which touches the circular constraint most of the time, while GBS also shows a similar motion pattern. The trajectory of MME is more irregular with larger position uncertainty depicted by the 3σ ellipses.

2.6 Summary

In this chapter, we study the problem of active formation control. It is well known that the estimation accuracy of the relative pose between pairs of robots often degrades when the robots move in a straight line formation. To address this issue, we allow the follower robot to deviate from its desired position but confine it within a circular disk

centered at its desired position. Then our objective becomes to determine the location inside this disk that the robot should move to in order to minimize its relative pose uncertainty. We formulate this as a constrained nonlinear optimization problem, and despite its non-convexity, we are able to analytically compute its global minimum, and significantly improve the follower's localization accuracy.

Chapter 3

Active Target Tracking: Multi-sensor Optimization

3.1 Introduction

Target tracking is one of the fundamental problems studied in robotics [62, 63]. Optimally tracking a moving target under motion and processing constraints is necessary in a number of applications such as environmental monitoring [78], surveillance [79, 80], human-robot interaction [81], as well as defense applications [82].

In most cases in practice, multiple *static* wireless sensors are employed in order to improve the tracking accuracy and increase the size of the surveillance area. In contrast to static sensors, whose density and sensing range are fixed, *mobile* sensors (i.e., robots) offer significant advantages. Firstly, mobile sensors can cover larger areas over time without the need to increase their number [83]. Secondly, by providing mobility to the sensors, their spatial distribution can change dynamically in order to adapt to the target's motion and keep it within sensing range. For example, a team of mobile sensors can actively pursue a target, so as to avert the target's evasion from the sensors' visibility range [84]. Thirdly, the mobility of sensors is a *necessary* condition for performing active target tracking. This is due to the fact that the sensors' relative observations (e.g., distance and/or bearing) are *nonlinear* functions of the relative position between the target and the sensor. Thus the target's estimate uncertainty, in most cases, depends on the sensing locations. Hence, to improve sensing performance, the sensors must have

the ability to adaptively change their sensing locations.

In order to intuitively understand why the locations where relative measurements are collected have a significant effect on estimation accuracy, consider the following simple but illustrative example where a single robot tracks a target using distance-only measurements corrupted by Gaussian noise [see Figures 3.1(a) and 3.1(b)]. In this scenario, the prior uncertainty for the position of the target is depicted by the solid-line 3σ ellipse shown in Figure 3.1(a). If the robot remains still and measures the distance to the target, then based solely on this measurement, the robot believes that the target is within the dotted-line circular ring with probability 99.7%. Combining the prior estimate with this measurement, the posterior uncertainty is only slightly reduced [dashed-line ellipse in Figure 3.1(a)]. As evident, by remaining in the same position, the robot's distance measurement provides limited information about the target's position along the horizontal direction. If instead, the robot moves to a new location [see Figure 3.1(b)], then combining this new distance measurement with the prior estimate will result in significant reduction of the uncertainty in both directions, but primarily along the horizontal direction. The improved confidence in the target-position estimate after this informative distance measurement is processed, is depicted by the small dashed-line ellipse in Figure 3.1(b).

This insightful example clearly demonstrates the impact of the robot's position on the estimate's uncertainty, and thus, motivates designing efficient algorithms to compute the optimal sensing locations. Moreover, the above problem of determining the best sensing location for single-robot target tracking can be extended to the case of multiple robots. Note also that in the above example, the robot is allowed to move to arbitrary locations in 2-D. In reality, however, the robot's motion is restricted by its own physical limitations, as well as by constraints imposed by the target or obstacles in the environment. Our objective in this work is to provide efficient algorithms for determining the optimal trajectories for a team of heterogeneous robots that track a moving target using combinations of relative observations (i.e., distance-only, bearing-only and distance-and-bearing measurements), while explicitly considering the robots' motion constraints.

Since accurately predicting the motion of the target over multiple time steps is very

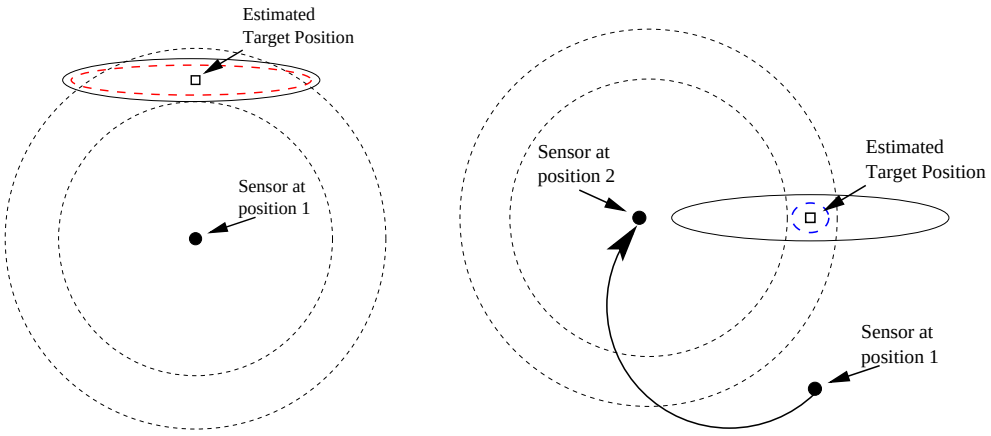


Figure 3.1: (a) Suboptimal target tracking: The robot remains in the same location. (b) Optimal target tracking: The robot moves to the position that minimizes the uncertainty for the target's position along the horizontal direction. In both plots, the prior uncertainty (3σ) is denoted by a solid-line ellipse, the posterior by a dashed-line ellipse, while the measurement uncertainty is depicted as a circular ring (dotted-line) with center the location of the robot.

challenging when the process-noise variance is large, in this chapter we focus our attention to the case where multiple robots must determine their optimal sensing locations for one step ahead at a time.[1] Specifically, we seek to minimize the uncertainty about the position of the target, expressed as the trace of the posterior covariance matrix for the target's position estimates, while considering maximum-speed limitations on the robots' motion. Additionally, in order to avoid collisions, we impose constraints on the minimum distance between any of the robots and the target. This formulation results in a non-convex objective function with non-convex constraints on the optimization variables (i.e., the robots' sensing locations).

The main contributions of this work are the following [85, 86, 87, 88]:

• We provide a complexity analysis of the one-step-ahead multi-robot active target tracking problem [85, 86]. More specifically, we explicitly consider constraints on the speed of the robots and prove that their inclusion makes the resulting constrained optimization problem, corresponding to one-step-ahead multi-robot active target tracking, NP-Hard. Thus, computing the global optimal solution for multi-robot active target tracking is extremely challenging, and it is expected that there exists no polynomial-time algorithm to calculate the global optimal solution, unless P=NP.

• We investigate the case of a single robot and for the first time we prove that the global optimal solution to the single-robot active target tracking problem can be determined analytically for arbitrary target motion models [85, 87]. In particular, we show that depending on the distance between the robot and the target, two distinct cases must be considered, each corresponding to a different pair of polynomial equations in two variables, whose finite and discrete solution set contains the optimal solution.

• We extend the above approach to the case of multiple heterogeneous robots by employing the non-linear Gauss-Seidel-relaxation (GSR) algorithm whose computational complexity is linear in the number of robots [85, 87]. Additionally, we compare the performance of the GSR algorithm to that of a grid-based exhaustive search (GBES), whose cost is exponential in the number of sensors, and show that GSR achieves comparable tracking accuracy at a significantly lower computational cost. Moreover, we demonstrate that the GSR algorithm outperforms gradient-descent-based approaches

[1] In Chapter 4, we address the problem of multi-step-ahead single-sensor active target tracking, when the process-noise variance is small and thus accurate prediction of the target's motion over multiple time steps becomes possible.

and is significantly better compared to the case where the sensors simply follow the target.

Following a brief review of related work in Section 3.2, we present the formulation of the one-step-ahead multi-sensor active target-tracking problem in Section 3.3, as well as provide a detailed complexity analysis of the corresponding constrained non-convex optimization problem in Section 3.4. In Section 3.5, the global optimal solution for a single sensor is determined analytically, while the non-linear GSR algorithm employed for solving the multiple-sensors case is described in Section 3.6. Extensive simulation and real-world experimental results are presented in Sections 3.7 and 3.8, respectively, while the conclusions of this work are summarized in Section 3.9.

3.2 Literature review

Most algorithms proposed for multi-sensor active target tracking focus on using either distance-only or distance-and-bearing measurements. In what follows, we classify and review existing multi-sensor active tracking approaches based on the measurement modality used.

3.2.1 Multi-sensor target tracking: distance-only observations

In [51], Martínez and Bullo addressed the problem of optimal sensor placement and motion coordination strategies for homogeneous sensor networks using distance-only measurements, where the emphasis is on the optimal sensor placement for (non random) static target position estimation. The objective is to minimize the determinant of the covariance matrix. The resulting control law requires that the sensors move on a polygon surrounding the target so as the vectors from the target to the sensors are uniformly (in terms of direction) spaced.

Yang *et al.* [55] presented an active sensing strategy using distance-only measurements, where both the trace and the determinant of the target position estimates' covariance are considered as objective functions. The authors proposed a control law, with constant step size, based on the gradient of the cost function with respect to each sensor's coordinates.

Recently, Stump *et al.* [56] investigated the problem of localizing a stationary target by processing distance-only measurements from mobile sensors. The objective is to select the sensing locations such that the time derivative of the determinant of the target-position estimates' information matrix (i.e., the inverse of the covariance matrix) is maximized. The proposed control law is based on the gradient of the cost function with respect to each sensor's coordinates, and is implemented in a distributed fashion. Additionally, the expected distance measurements in the next time step are approximated by assuming that they will be the same as these recorded at the sensors' current locations.

3.2.2 Multi-sensor target tracking: distance-and-bearing observations

Olfati-Saber [53] addressed the problem of distributed target tracking for mobile sensor networks with a dynamic communication topology. The author tackled the network connectivity issue using a flocking-based mobility model, and presented a modified version of the distributed Kalman filter algorithm for estimating the target's state. In this case, the sensors use both distance and bearing measurements to a target that moves in 2D with constant velocity driven by zero-mean Gaussian noise, and seek to minimize their distances to the target, while avoiding collisions.

Chung *et al.* [54] presented a decentralized motion planning algorithm for solving the multi-sensor target tracking problem using both distance and bearing measurements. The authors employed the determinant of the target's position covariance matrix as the cost function. The decentralized control law in this case is based on the gradient of the cost function with respect to each of the sensor's coordinates with constant step-size of 1.

Stroupe and Balch [59] proposed an approximate tracking behavior, where the mobile sensors attempt to minimize the target's location uncertainty using distance-and-bearing measurements. The objective function is the determinant of the target position estimates' covariance matrix, and the optimization problem is solved by greedy search over the discretized set of candidate headings, separately for each sensor. Additionally, the expected information gain from the teammates' actions is approximated by assuming that their measurements in the next time step will be the same as these recorded at their current locations.

3.2.3 Summary of related work

The main drawback of the previous approaches is that no physical constraints on the motion of the sensors are considered. The only exception is the work presented in [59] for distance-and-bearing observations. However, the proposed grid-based exhaustive search algorithm, when extended to the multi-sensor case, has computational complexity exponential in the number of sensors, which becomes prohibitive when the number of the sensors is large and/or the size of the grid cell is small. In addition, teams of heterogeneous sensors using combinations of distance and distance-and-bearing observations are only considered in [55], whose gradient-based algorithm seeks to find a local minimum, while its convergence rate is not addressed. On the other hand, analytical solutions for multi-sensor active target tracking are provided only for the case of a stationary target [51].

Furthermore, as mentioned in the Section 3.1, active target tracking essentially seeks to minimize a scalar function, related to the estimation uncertainty, by optimizing over the robots' sensing locations. What is missing from the previous work is an in-depth computational-complexity analysis of the underlying optimization problem associated with active target tracking. We are particularly interested in how the complexity increases with respect to the problem size, i.e., the number of sensors. Additionally, our objective is to investigate the impact of the mobility constraints of each robot on the computational complexity. Answering these questions will help us better understand the characteristics of the active target tracking problem, and will offer significant insights for designing efficient algorithms. We should also note that the approximate algorithms employed by previous approaches (gradient descent and exhaustive search) do not take advantage of the (possibly) special structure of the underlying optimization problem. It is our belief that by exploiting the problem's structure can lead to exact algorithms, for global minimization of small-scale problems, and high-accuracy, approximate algorithms for efficiently solving large-scale problems.

In this chapter, we address the most general case of multi-sensor active target tracking when processing combinations of relative measurements (i.e., distance and/or bearing). Specifically, we first establish the NP-Hardness of the one-step-ahead multi-sensor active target tracking problem. We then address the problem of single-sensor target tracking where we explicitly consider constraints on the robot's motion by imposing

bounds on its maximum speed, as well as on the minimum distance between the robot and the target. However, contrary to [51], we require no particular type of target motion. Our main contribution is that we derive the global optimal solutions for distance-only, bearing-only, and distance-and-bearing observations, analytically. Moreover, we generalize these results to the multi-sensor case by employing Gauss-Seidel relaxation that minimizes the trace of the target's position estimate covariance with respect to the motion of all sensors in a coordinate-descent fashion. Our algorithm is applicable to heterogeneous sensor teams that use combinations of observations, has computational complexity linear in the number of sensors, and achieves tracking accuracy indistinguishable of that of an exhaustive search over all possible sensor locations, whose complexity increases exponentially in the number of sensors.

3.3 Problem formulation

Consider a group of mobile sensors moving in a plane and tracking the position of a moving target by processing relative measurements, consisting of distance-only, bearing-only, and distance-and-bearing observations. In this work, we study the case of global tracking, i.e., the position of the target is described with respect to a fixed (global) frame of reference, instead of a relative group-centered one. Hence, we hereafter employ the assumption that the pose of each tracking sensor is known with high accuracy within the global frame of reference (e.g., from precise GPS and compass measurements).

Furthermore, we consider the case where each sensor moves in 2D with speed v_i, which is *upper bounded* by $v_{i\max}$, $i = 1, \ldots, M$, where M is the number of sensors. Therefore, at time-step $k+1$, sensor-i can only move within a circular region centered at its position at time-step k with radius $v_{i\max}\delta t$, where δt is the time step (see Figure 3.2). In order to avoid collisions with the target, we also require that the distance between the target and sensor-i to be greater than a threshold ρ_i, i.e., sensor-i is prohibited to move inside a circular region centered at the target's position estimate at time-step $k+1$ with radius ρ_i (see Figure 3.2).[2] Note also that since the motion of the target can be reliably predicted for the next time step only, our objective is to determine the next

[2] Ideally, the collision-avoidance constraints should be defined using the *true* position of the target. However, since the true target position is *unavailable*, we instead use the *estimated* target position and appropriately increase the safety distance to account for the uncertainty in this estimate.

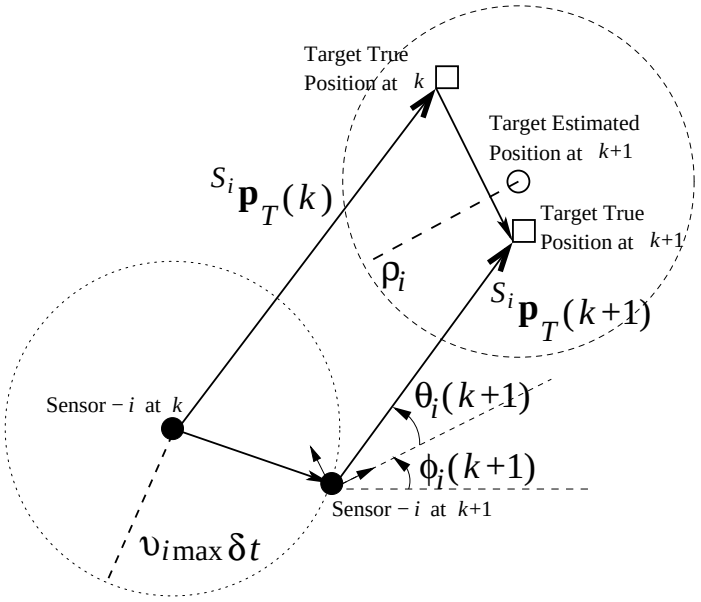


Figure 3.2: Illustration of the i-th sensor's and target's motion: Sensor-i moves in 2D with speed v_i, which is bounded by $v_{i\max}$. From time-step k to $k+1$, the sensor can only move *within* a circular region centered at its position at time-step k with radius $v_{i\max}\delta t$. Furthermore, to avoid collision with the target, sensor-i is prohibited to move inside a circular region centered at the target's position *estimate* at time-step $k+1$ with radius ρ_i. $^{S_i}\mathbf{p}_T$ is the target's position with respect to sensor-i. The distance measurement of sensor-i is the norm of $^{S_i}\mathbf{p}_T(k+1)$ plus noise, and the bearing measurement of sensor-i is $\theta_i(k+1)$ plus noise.

best sensing locations for all sensors at one time step ahead.

In the next two sections, we present the target's state propagation equations and the sensors' measurement models.

3.3.1 State propagation

In this work, we employ the extended Kalman filter (EKF) for recursively estimating the target's state, $\mathbf{x}_T(k)$. This is defined as a vector of dimension $2N$, where $N-1$ is the highest-order time derivative of the target's position described by the motion model,

and can include components such as position, velocity, and acceleration:

$$\begin{aligned}\mathbf{x}_T(k) &= \left[(\mathbf{p}_T(k))^{\mathrm{T}} \;\; (\dot{\mathbf{p}}_T(k))^{\mathrm{T}} \;\; \ldots \;\; \left(\mathbf{p}_T^{(N-1)}(k)\right)^{\mathrm{T}}\right]^{\mathrm{T}} \\ &= \left[x_T(k)\; y_T(k)\; \dot{x}_T(k)\; \dot{y}_T(k)\; \ddot{x}_T(k)\; \ddot{y}_T(k)\; \ldots\; x_T^{(N-1)}(k)\; y_T^{(N-1)}(k)\right]^{\mathrm{T}}. \quad (3.1)\end{aligned}$$

We consider the case where the target moves randomly and assume that we know the stochastic model describing the motion of the target (e.g., constant-acceleration or constant-velocity, etc.). However, as it will become evident later on, our sensing strategy does not depend on the particular selection of the target's motion model.

The discrete-time state propagation equation is:

$$\mathbf{x}_T(k+1) = \mathbf{\Phi}_k \mathbf{x}_T(k) + \mathbf{w}_d(k), \quad (3.2)$$

where $\mathbf{w}_d$ is a zero-mean white Gaussian noise with covariance $\mathbf{Q}_k = \mathbb{E}[\mathbf{w}_d(k)\mathbf{w}_d^{\mathrm{T}}(k)]$. The state transition matrix, $\mathbf{\Phi}_k$ that appears in (3.2) depends on the motion model used [62]. In our work, this can be an *arbitrary*, but *known*, matrix, since no assumptions on its properties are imposed.

The estimate of the target's state is propagated by:[3]

$$\hat{\mathbf{x}}_T(k+1|k) = \mathbf{\Phi}_k \hat{\mathbf{x}}_T(k|k), \quad (3.3)$$

where $\hat{\mathbf{x}}_T(\ell|j)$ is the state estimate at time-step ℓ, after measurements up to time-step j have been processed.

The error-state covariance matrix is propagated as:

$$\mathbf{P}_{k+1|k} = \mathbf{\Phi}_k \mathbf{P}_{k|k} \mathbf{\Phi}_k^{\mathrm{T}} + \mathbf{Q}_k, \quad (3.4)$$

where $\mathbf{P}_{\ell|j}$ is the covariance of the error, $\tilde{\mathbf{x}}_T(\ell|j) = \mathbf{x}_T(\ell) - \hat{\mathbf{x}}_T(\ell|j)$, in the state estimate.

3.3.2 Measurement model

Let us denote the complete set of the sensor team as $\mathcal{M} = \{1, \ldots, M\}$, where M is the number of the sensors. At time-step $k+1$, based on the type of the measurement that

[3] In the remainder of this chapter, the "hat" symbol, $\hat{}$, denotes the estimated value of a quantity, while the "tilde" symbol, $\tilde{}$, represents the error between the actual value of a quantity and its estimate. The relationship between a variable, $\mathbf{x}$, and its estimate, $\hat{\mathbf{x}}$, is $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$. Additionally, "$\succ$" and "$\succeq$" denote the matrix inequality in the positive definite and positive semidefinite sense, respectively. $\mathbf{0}_{m\times n}$ and $\mathbf{I}_n$ represent the $m \times n$ zero matrix and $n \times n$ identity matrix, respectively.

each sensor collects, $\mathcal{M}$ can be partitioned into $\mathcal{M}_1 \cup \mathcal{M}_2 \cup \mathcal{M}_3$, where $\mathcal{M}_1$ denotes the set of sensors that have access to both distance and bearing observations; $\mathcal{M}_2$ comprises sensors that measure only bearing; and $\mathcal{M}_3$ consists of sensors that record distance-only measurements. In what follows, $\mathbf{p}_T(k+1) = [x_T(k+1)\ y_T(k+1)]^{\mathrm{T}}$ and $\mathbf{p}_{S_i}(k+1) = [x_{S_i}(k+1)\ y_{S_i}(k+1)]^{\mathrm{T}}$ denote the positions of the target and the i-th sensor, respectively, expressed in the global frame of reference. Furthermore, to simplify the notation, we introduce the following quantities ($i = 1, \ldots, M$):

$$\begin{aligned}
\Delta x_{T_i}(k+1) &= x_T(k+1) - x_{S_i}(k+1),\\
\Delta y_{T_i}(k+1) &= y_T(k+1) - y_{S_i}(k+1),\\
\widehat{\Delta x}_{T_i}(k+1|k) &= \hat{x}_T(k+1|k) - x_{S_i}(k+1),\\
\widehat{\Delta y}_{T_i}(k+1|k) &= \hat{y}_T(k+1|k) - y_{S_i}(k+1),\\
\mathbf{p}_i = \mathbf{p}_i(k+1) &= \mathbf{p}_{S_i}(k+1) - \hat{\mathbf{p}}_T(k+1|k).
\end{aligned} \tag{3.5}$$

Distance-and-bearing observation model

At time-step $k+1$, sensor-j ($j \in \mathcal{M}_1$) records its distance-and-bearing observations [$d_j(k+1)$ and $\theta_j(k+1)$] to the target, as shown in Figure 3.2. The measurement equation is

$$\mathbf{z}_j(k+1) = \begin{bmatrix} d_j(k+1) \\ \theta_j(k+1) \end{bmatrix} + \begin{bmatrix} n_{d_j}(k+1) \\ n_{\theta_j}(k+1) \end{bmatrix}, \tag{3.6}$$

with

$$d_j(k+1) = \sqrt{\Delta x_{T_j}^2(k+1) + \Delta y_{T_j}^2(k+1)}\,, \tag{3.7}$$

$$\theta_j(k+1) = \arctan\left(\frac{\Delta y_{T_j}(k+1)}{\Delta x_{T_j}(k+1)}\right) - \phi_j(k+1), \tag{3.8}$$

where $\phi_j(k+1)$ is the orientation of sensor-j, and $\mathbf{n}_j(k+1) = \left[n_{d_j}(k+1)\ n_{\theta_j}(k+1)\right]^{\mathrm{T}}$ is the noise in the j-th sensor's measurements, which is a zero-mean white Gaussian process with covariance $\mathbf{R}_j = \mathbb{E}[\mathbf{n}_j(k+1)\mathbf{n}_j^{\mathrm{T}}(k+1)] = \mathrm{diag}(\sigma_{d_j}^2, \sigma_{\theta_j}^2)$, and independent of the noise in the other sensors, i.e., $\mathbb{E}[\mathbf{n}_j(k+1)\mathbf{n}_i^{\mathrm{T}}(k+1)] = \mathbf{0}$ for $i \neq j$.

The measurement of sensor-j is a nonlinear function of the state variable $\mathbf{x}_T$ [see (3.6)]. The measurement-error equation for sensor-j, obtained by linearizing (3.6) is

$$\tilde{\mathbf{z}}_j(k+1|k) = \mathbf{z}_j(k+1) - \hat{\mathbf{z}}_j(k+1|k) \simeq \mathbf{H}_{k+1}^{(j)} \tilde{\mathbf{x}}_T(k+1|k) + \mathbf{n}_j(k+1), \tag{3.9}$$

where

$$\begin{aligned}
\hat{\mathbf{z}}_j(k+1|k) &= [\hat{d}_j(k+1|k) \quad \hat{\theta}_j(k+1|k)]^{\mathrm{T}}, \\
\hat{d}_j(k+1|k) &= \sqrt{\widehat{\Delta x}_{T_j}^2(k+1|k) + \widehat{\Delta y}_{T_j}^2(k+1|k)}\,, \\
\hat{\theta}_j(k+1|k) &= \arctan\left(\frac{\widehat{\Delta y}_{T_j}(k+1|k)}{\widehat{\Delta x}_{T_j}(k+1|k)}\right) - \phi_j(k+1).
\end{aligned}$$

Note that the measurement matrix in (3.9) has a block column structure, which is given by the following expression:

$$\mathbf{H}_{k+1}^{(j)} = \begin{bmatrix} \mathbf{h}_j^{\mathrm{T}}(k+1) & \mathbf{0}_{2\times(2N-2)} \end{bmatrix}, \tag{3.10}$$

where $2N$ is the dimension of the state vector and

$$\mathbf{h}_j(k+1) = \begin{bmatrix} \mathbf{h}_{d_j}(k+1) & \mathbf{h}_{\theta_j}(k+1) \end{bmatrix} = \begin{bmatrix} \frac{-1}{\sqrt{\mathbf{p}_j^{\mathrm{T}}\mathbf{p}_j}}\mathbf{p}_j & \frac{1}{\mathbf{p}_j^{\mathrm{T}}\mathbf{p}_j}\mathbf{J}\mathbf{p}_j \end{bmatrix}, \tag{3.11}$$

where $\mathbf{J} = \mathbf{C}\left(-\frac{\pi}{2}\right)$ and $\mathbf{C}(\cdot)$ is the 2×2 rotational matrix.

Bearing-only observation model

At time-step $k+1$, sensor-ℓ ($\ell \in \mathcal{M}_2$) only has access to its bearing measurement $\theta_\ell(k+1)$ towards the target [see (3.8)], and the measurement and measurement-error equations are

$$\begin{aligned}
\mathbf{z}_\ell(k+1) &= \theta_\ell(k+1) + n_{\theta_\ell}(k+1), \\
\tilde{\mathbf{z}}_\ell(k+1|k) &\simeq \mathbf{H}_{k+1}^{(\ell)} \tilde{\mathbf{x}}_T(k+1|k) + \mathbf{n}_\ell(k+1),
\end{aligned} \tag{3.12}$$

where $\mathbf{n}_\ell(k+1) = n_{\theta_\ell}(k+1)$ is the zero-mean white Gaussian measurement noise with variance $\mathbf{R}_\ell = \mathbb{E}[\mathbf{n}_\ell(k+1)\mathbf{n}_\ell^{\mathrm{T}}(k+1)] = \sigma_{\theta_\ell}^2$, which is independent of the noise in the other sensors. As before, the measurement matrix $\mathbf{H}_{k+1}^{(\ell)}$ has the following structure:

$$\mathbf{H}_{k+1}^{(\ell)} = \begin{bmatrix} \mathbf{h}_\ell^{\mathrm{T}}(k+1) & \mathbf{0}_{1\times(2N-2)} \end{bmatrix}, \tag{3.13}$$

$$\mathbf{h}_\ell(k+1) = \mathbf{h}_{\theta_\ell}(k+1) = \frac{1}{\mathbf{p}_\ell^{\mathrm{T}}\mathbf{p}_\ell}\mathbf{J}\mathbf{p}_\ell. \tag{3.14}$$

Distance-only observation model

At time-step $k+1$, sensor-ι ($\iota \in \mathcal{M}_3$) only measures its distance $d_\iota(k+1)$ to the target [see (3.7)], therefore the measurement equation is

$$\mathbf{z}_\iota(k+1) = d_\iota(k+1) + n_{d_\iota}(k+1),$$

and the corresponding measurement-error equation is

$$\tilde{\mathbf{z}}_\iota(k+1|k) \simeq \mathbf{H}_{k+1}^{(\iota)} \tilde{\mathbf{x}}_T(k+1|k) + \mathbf{n}_\iota(k+1), \tag{3.15}$$

where $\mathbf{n}_\iota(k+1) = n_{d_\iota}(k+1)$ is the noise in the ι-th sensor's distance measurement, which is a zero-mean white Gaussian process with variance $\mathbf{R}_\iota = \mathbb{E}[\mathbf{n}_\iota(k+1)\mathbf{n}_\iota^{\mathrm{T}}(k+1)] = \sigma_{d_\iota}^2$, and independent of the noise in the other sensors. Additionally, the measurement matrix $\mathbf{H}_{k+1}^{(\iota)}$ in (3.15) is given by the following expression:

$$\mathbf{H}_{k+1}^{(\iota)} = \begin{bmatrix} \mathbf{h}_\iota^{\mathrm{T}}(k+1) & \mathbf{0}_{1\times(2N-2)} \end{bmatrix}, \tag{3.16}$$

$$\mathbf{h}_\iota(k+1) = \mathbf{h}_{d_\iota}(k+1) = \frac{-1}{\sqrt{\mathbf{p}_\iota^{\mathrm{T}}\mathbf{p}_\iota}}\mathbf{p}_\iota. \tag{3.17}$$

Linearized measurement-error equation

The overall measurement-error equation at time-step $k+1$, obtained by stacking all measurement-error equations corresponding to each sensor [see (3.9), (3.12), and (3.15)], is

$$\tilde{\mathbf{z}}(k+1|k) = \begin{bmatrix} \tilde{\mathbf{z}}_1^{\mathrm{T}}(k+1|k) & \dots & \tilde{\mathbf{z}}_M^{\mathrm{T}}(k+1|k) \end{bmatrix}^{\mathrm{T}} \simeq \mathbf{H}_{k+1}\tilde{\mathbf{x}}_T(k+1|k) + \mathbf{n}(k+1),$$

with $\mathbf{n}(k+1) = \begin{bmatrix} \mathbf{n}_1^{\mathrm{T}}(k+1) & \dots & \mathbf{n}_M^{\mathrm{T}}(k+1) \end{bmatrix}^{\mathrm{T}}$ and [see (3.10), (3.13), and (3.16)]

$$\mathbf{H}_{k+1} = \begin{bmatrix} \left(\mathbf{H}_{k+1}^{(1)}\right)^{\mathrm{T}} & \dots & \left(\mathbf{H}_{k+1}^{(M)}\right)^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \mathbf{H}_{e,k+1} & \mathbf{0} \end{bmatrix},$$

where $\mathbf{H}_{e,k+1}$ is the block element of the measurement matrix corresponding to the target's position, i.e.,

$$\mathbf{H}_{e,k+1}^{\mathrm{T}} = \begin{bmatrix} \mathbf{h}_1(k+1) & \dots & \mathbf{h}_M(k+1) \end{bmatrix}, \tag{3.18}$$

where $\mathbf{h}_i(k+1)$, $i = 1, \dots, M$, are defined based on the type of the observations considered [see (3.11), (3.14), and (3.17)]. Note also that $\mathbf{R} = \mathbb{E}[\mathbf{n}(k+1)\mathbf{n}^{\mathrm{T}}(k+1)] = \mathrm{diag}(\mathbf{R}_i)$, $i = 1, \dots, M$, due to the independence of the noise in each sensor.

3.3.3 State and covariance update

Once the measurements, $\mathbf{z}_i(k+1),\ i=1,\ldots,M$, from all the sensors are available, they are transmitted and processed at a fusion center (e.g., one of the robots in the team), and the target's state estimate and its covariance are updated as:

$$\begin{aligned}\hat{\mathbf{x}}_T(k+1|k+1) &= \hat{\mathbf{x}}_T(k+1|k) + \mathbf{K}_{k+1}\tilde{\mathbf{z}}(k+1|k),\\ \mathbf{P}_{k+1|k+1} &= \mathbf{P}_{k+1|k} - \mathbf{K}_{k+1}\mathbf{S}_{k+1}\mathbf{K}_{k+1}^{\mathrm{T}}.\end{aligned} \tag{3.19}$$

where $\mathbf{K}_{k+1} = \mathbf{P}_{k+1|k}\mathbf{H}_{k+1}^{\mathrm{T}}\mathbf{S}_{k+1}^{-1}$ is the Kalman gain, and $\mathbf{S}_{k+1} = \mathbf{H}_{k+1}\mathbf{P}_{k+1|k}\mathbf{H}_{k+1}^{\mathrm{T}} + \mathbf{R}$ is the measurement residual covariance.

Our objective in this work is to determine the active-sensing strategy that minimizes the uncertainty for the *position* estimate of the target. In order to account for the impact of the prior state estimates on the motion of the sensors, we first present the following lemma.

Lemma 4. *The posterior (updated) covariance for the target's position estimate depends on (i) the measurement sub-matrix corresponding to the target's* position*, and (ii) the prior (propagated) covariance sub-matrix of the target's* position*, i.e.,*

$$\mathbf{P}_{k+1|k+1,11} = \Big(\big(\mathbf{P}_{k+1|k,11}\big)^{-1} + \mathbf{H}_{e,k+1}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}_{e,k+1}\Big)^{-1}, \tag{3.20}$$

where $\mathbf{H}_{e,k+1}$ is defined in (3.18) *and $\mathbf{P}_{\ell|j,11}$ denotes the 2×2 upper diagonal sub-matrix of $\mathbf{P}_{\ell|j}$ corresponding to the covariance in the position estimates.*

Proof. The covariance matrices appearing in (3.20) are defined based on the following partition:

$$\mathbf{P}_{\ell|j} = \begin{bmatrix}\mathbf{P}_{\ell|j,11} & \mathbf{P}_{\ell|j,12}\\ \mathbf{P}_{\ell|j,12}^{\mathrm{T}} & \mathbf{P}_{\ell|j,22}\end{bmatrix}, \tag{3.21}$$

where the 2×2 matrix $\mathbf{P}_{\ell|j,11}$ denotes the covariance for the target's *position* estimate, $\hat{\mathbf{p}}_T = [\hat{x}_T\ \hat{y}_T]^{\mathrm{T}}$, at time-step ℓ given measurements up to time-step j.

Employing the matrix inversion lemma [89], the covariance update equation (3.19) can be written as:

$$\mathbf{P}_{k+1|k+1}^{-1} = \mathbf{P}_{k+1|k}^{-1} + \mathbf{H}_{k+1}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}_{k+1}. \tag{3.22}$$

Note that if the state vector only contains the position of the target, then (3.20) is identical to (3.22).

In the general case, when the state vector also contains higher-order derivatives of the position (e.g., velocity, acceleration, etc.), substituting

$$\mathbf{P}_{k+1|k}^{-1} = \boldsymbol{\Upsilon} = \begin{bmatrix} \boldsymbol{\Upsilon}_{11} & \boldsymbol{\Upsilon}_{12} \\ \boldsymbol{\Upsilon}_{12}^{\mathrm{T}} & \boldsymbol{\Upsilon}_{22} \end{bmatrix} \tag{3.23}$$

and

$$\mathbf{H}_{k+1}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}_{k+1} = \begin{bmatrix} \mathbf{H}_{e,k+1}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}_{e,k+1} & \mathbf{0}_{2\times(2N-2)} \\ \mathbf{0}_{(2N-2)\times 2} & \mathbf{0}_{(2N-2)\times(2N-2)} \end{bmatrix}$$

on the right hand-side of (3.22) yields:

$$\mathbf{P}_{k+1|k+1} = \begin{bmatrix} \boldsymbol{\Upsilon}_{11} + \mathbf{H}_{e,k+1}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}_{e,k+1} & \boldsymbol{\Upsilon}_{12} \\ \boldsymbol{\Upsilon}_{12}^{\mathrm{T}} & \boldsymbol{\Upsilon}_{22} \end{bmatrix}^{-1}. \tag{3.24}$$

Employing the property of the Schur complement [89] for the inversion of a partitioned matrix in (3.23)-(3.24), we obtain

$$\begin{aligned} \mathbf{P}_{k+1|k+1,11} &= \left(\boldsymbol{\Upsilon}_{11} + \mathbf{H}_{e,k+1}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}_{e,k+1} - \boldsymbol{\Upsilon}_{12}\boldsymbol{\Upsilon}_{22}^{-1}\boldsymbol{\Upsilon}_{12}^{\mathrm{T}}\right)^{-1} \\ &= \left(\left(\mathbf{P}_{k+1|k,11}\right)^{-1} + \mathbf{H}_{e,k+1}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}_{e,k+1}\right)^{-1}. \end{aligned}$$

□

The importance of this lemma is that the optimization algorithms presented in Sections 3.5–3.6 can be derived based on (3.20) for the position covariance update – instead of (3.19) for the entire state covariance update – regardless of the stochastic process model employed for describing the target's motion.

Exploiting the fact that $\mathbf{R}$ is diagonal, and substituting (3.18) into (3.20), we obtain the following expression for $\mathbf{P}_{k+1|k+1,11}$:

$$\begin{aligned} \mathbf{P}_{k+1|k+1,11} = \Bigg[&\left(\mathbf{P}_{k+1|k,11}\right)^{-1} + \sum_{j\in\mathcal{M}_1}\left(\frac{1}{\sigma_{d_j}^2}\frac{\mathbf{p}_j\mathbf{p}_j^{\mathrm{T}}}{\mathbf{p}_j^{\mathrm{T}}\mathbf{p}_j} + \frac{1}{\sigma_{\theta_j}^2}\frac{\mathbf{J}\mathbf{p}_j\mathbf{p}_j^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{(\mathbf{p}_j^{\mathrm{T}}\mathbf{p}_j)^2}\right) \\ &+ \sum_{\ell\in\mathcal{M}_2}\frac{1}{\sigma_{\theta_\ell}^2}\frac{\mathbf{J}\mathbf{p}_\ell\mathbf{p}_\ell^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{(\mathbf{p}_\ell^{\mathrm{T}}\mathbf{p}_\ell)^2} + \sum_{\iota\in\mathcal{M}_3}\frac{1}{\sigma_{d_\iota}^2}\frac{\mathbf{p}_\iota\mathbf{p}_\iota^{\mathrm{T}}}{\mathbf{p}_\iota^{\mathrm{T}}\mathbf{p}_\iota}\Bigg]^{-1}. \end{aligned} \tag{3.25}$$

In order to encapsulate all three measurement models (see Section 3.3.2) into a unified framework, we introduce two binary variables $\kappa_{d_i} \in \{0,1\}$ and $\kappa_{\theta_i} \in \{0,1\}$ for sensor-i, $i = 1,\ldots,M$. $\kappa_{d_i} = 1$ if sensor-i can measure relative distance at time-step $k+1$, otherwise $\kappa_{d_i} = 0$; similarly, $\kappa_{\theta_i} = 1$ if sensor-i is capable of taking a bearing observation at time-step $k+1$, otherwise $\kappa_{\theta_i} = 0$. Following this convention, we have $\kappa_{d_i} = \kappa_{\theta_i} = 1,\ \forall i \in \mathcal{M}_1$; $\kappa_{d_i} = 0,\ \kappa_{\theta_i} = 1,\ \forall i \in \mathcal{M}_2$; $\kappa_{d_i} = 1,\ \kappa_{\theta_i} = 0,\ \forall i \in \mathcal{M}_3$. Using this convention, (3.25) can be written as:

$$\mathbf{P}_{k+1|k+1,11} = \left(\left(\mathbf{P}_{k+1|k,11}\right)^{-1} + \sum_{i=1}^{M} \frac{\kappa_{d_i}}{\sigma_{d_i}^2} \frac{\mathbf{p}_i \mathbf{p}_i^{\mathrm{T}}}{\mathbf{p}_i^{\mathrm{T}} \mathbf{p}_i} + \sum_{i=1}^{M} \frac{\kappa_{\theta_i}}{\sigma_{\theta_i}^2} \frac{\mathbf{J}\mathbf{p}_i \mathbf{p}_i^{\mathrm{T}} \mathbf{J}^{\mathrm{T}}}{(\mathbf{p}_i^{\mathrm{T}} \mathbf{p}_i)^2} \right)^{-1}. \tag{3.26}$$

Remark 5. *Note that $\forall i \in \mathcal{M}_2$, the term $\sigma_{d_i}^2$ is irrelevant, i.e., $\sigma_{d_i}^2$ can be set to any positive real number, since $\kappa_{d_i}\sigma_{d_i}^{-2} = 0$ regardless of the specific value of $\sigma_{d_i}^2$. Similarly, $\sigma_{\theta_i}^2$ is irrelevant $\forall i \in \mathcal{M}_3$.*

Remark 6. *When sensor-i is unable to detect the target and hence records neither distance nor bearing observations at time-step $k+1$, the corresponding κ_{d_i} and κ_{θ_i} in* (3.26) *are set to zero. In this case, the target's position posterior covariance is independent of the variable $\mathbf{p}_i$. However, we still require that sensor-i minimizes its distance ($\|\mathbf{p}_i\|$) to the estimated target location, while adhering to its motion and collision-avoidance constraints, so as to increase its probability of re-detecting the target in the following time steps. The updated estimate of the target's state $\hat{\mathbf{x}}_T(k+1|k+1)$ is communicated to sensor-i by those sensors that are able to detect and take relative measurements at time-step $k+1$. In case none of the robots can detect the target, i.e., $\kappa_{d_i} = \kappa_{\theta_i} = 0,\ \forall\, i \in \mathcal{M}$, then all robots propagate the previous state estimate [see* (3.3)*], and plan their motions so as to minimize their distances from the predicted target's location.*

In the next section, we formulate the sensors' one-step-ahead *optimal motion strategy* as a constrained optimization problem, and discuss its properties.

3.3.4 Problem statement and reformulation

As is evident from (3.5) and (3.26), after each update step the target's position covariance matrix will depend, through $\mathbf{p}_i$, on all the next sensors' positions $\mathbf{p}_{S_i}(k+1) = [x_{S_i}(k+1)\ y_{S_i}(k+1)]^{\mathrm{T}},\ i = 1,\ldots,M$. Assume that at time-step k, sensor-i is located at

$\mathbf{p}_{S_i}(k) = [x_{S_i}(k)\ y_{S_i}(k)]^{\mathrm{T}}$. At time-step $k+1$ its position $\mathbf{p}_{S_i}(k+1)$ is confined within a circular region centered at $\mathbf{p}_{S_i}(k)$, due to the maximum-speed constraint, but outside a circular region centered at $\hat{\mathbf{p}}_T(k+1|k)$ so as to avoid collisions (see Figure 3.2), i.e.,

$$\|\mathbf{p}_{S_i}(k+1) - \mathbf{p}_{S_i}(k)\| \leq r_i, \tag{3.27}$$

$$\|\mathbf{p}_{S_i}(k+1) - \hat{\mathbf{p}}_T(k+1|k)\| \geq \rho_i, \tag{3.28}$$

where $r_i = \min\left(v_{i\max}\delta t, \|\mathbf{p}_{S_i}(k) - \hat{\mathbf{p}}_T(k+1|k)\|\right) \leq v_{i\max}\delta t,\ i = 1, \ldots, M$.

Substituting $\mathbf{p}_i$ [see (3.5)] in the above two inequalities, yields:

$$\left\|\mathbf{p}_i - [\mathbf{p}_{S_i}(k) - \hat{\mathbf{p}}_T(k+1|k)]\right\| \leq r_i, \tag{3.29}$$

$$\|\mathbf{p}_i\| \geq \rho_i, \tag{3.30}$$

thus, the feasible region of $\mathbf{p}_i$ is inside a circle of radius r_i centered at $\mathbf{p}_{S_i}(k) - \hat{\mathbf{p}}_T(k+1|k)$, and outside a circle of radius ρ_i centered at the origin $[0\ 0]^{\mathrm{T}}$. Note that the estimate $\hat{\mathbf{p}}_T(k+1|k)$ [see (3.3)] is shared among all sensors, and can be treated as a constant at time-step $k+1$. Hence, once $\mathbf{p}_i,\ i = 1, \ldots, M$, is determined, the location of sensor-i at time-step $k+1$, $\mathbf{p}_{S_i}(k+1),\ i = 1, \ldots, M$, can be obtained through (3.5).

The problem we address in this work is that of determining the sensors' *optimal motion strategy*, i.e., the set $\{\mathbf{p}_i,\ i = 1, \ldots, M\}$, that minimizes the *trace* of the target's position estimate covariance matrix [see (3.26)], under the constraints specified in (3.29)–(3.30):

- Optimization Problem 3 (Π_3)

$$\begin{aligned} \underset{\mathbf{p}_1,\ldots,\mathbf{p}_M}{\text{minimize}}\ & \operatorname{tr}\left(\left(\mathbf{P}_{k+1|k,11}\right)^{-1} + \sum_{i=1}^{M} \frac{\kappa_{d_i}}{\sigma_{d_i}^2} \frac{\mathbf{p}_i\mathbf{p}_i^{\mathrm{T}}}{\mathbf{p}_i^{\mathrm{T}}\mathbf{p}_i} + \sum_{i=1}^{M} \frac{\kappa_{\theta_i}}{\sigma_{\theta_i}^2} \frac{\mathbf{J}\mathbf{p}_i\mathbf{p}_i^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{(\mathbf{p}_i^{\mathrm{T}}\mathbf{p}_i)^2}\right)^{-1} \\ \text{subject to}\ & \left\|\mathbf{p}_i - [\mathbf{p}_{S_i}(k) - \hat{\mathbf{p}}_T(k+1|k)]\right\| \leq r_i, \\ & \|\mathbf{p}_i\| \geq \rho_i, \qquad\qquad i = 1, \ldots, M. \end{aligned} \tag{3.31}$$

In what follows, we apply a coordinate transformation (see Lemma 7), to convert the objective function of Π_3 into (3.32), in which $\mathbf{\Lambda}$ is a diagonal matrix.

Lemma 7. *Assume the symmetric positive-definite matrix* $\mathbf{P}_{k+1|k,11}$ *is non-diagonal, and consider the eigen-decomposition* $\mathbf{P}_{k+1|k,11}^{-1} = \mathbf{C}(\varphi_0)\mathbf{\Lambda}\mathbf{C}(-\varphi_0)$*, where* $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1^{-1}, \lambda_2^{-1})$ *and* $\lambda_1 \geq \lambda_2 > 0$*. Then*

$$\mathrm{tr}(\mathbf{P}_{k+1|k+1,11}) = \mathrm{tr}\left(\mathbf{\Lambda} + \sum_{i=1}^{M} \frac{\kappa_{d_i}}{\sigma_{d_i}^2} \frac{\mathbf{s}_i \mathbf{s}_i^{\mathrm{T}}}{\mathbf{s}_i^{\mathrm{T}} \mathbf{s}_i} + \sum_{i=1}^{M} \frac{\kappa_{\theta_i}}{\sigma_{\theta_i}^2} \frac{\mathbf{J}\mathbf{s}_i \mathbf{s}_i^{\mathrm{T}} \mathbf{J}^{\mathrm{T}}}{(\mathbf{s}_i^{\mathrm{T}} \mathbf{s}_i)^2}\right)^{-1}, \tag{3.32}$$

where $\mathbf{s}_i = \mathbf{C}(-\varphi_0)\mathbf{p}_i,\ i = 1, \ldots, M$*.*

Proof. Substituting $\mathbf{P}_{k+1|k,11}^{-1} = \mathbf{C}(\varphi_0)\mathbf{\Lambda}\mathbf{C}(-\varphi_0)$ and $\mathbf{p}_i = \mathbf{C}(\varphi_0)\mathbf{s}_i$ in (3.31), employing the equality $\mathbf{C}(-\varphi_0)\mathbf{J} = \mathbf{J}\mathbf{C}(-\varphi_0)$ which holds since both are 2×2 rotational matrices, and noting that the trace operation is invariant to similarity transformations results in (3.32). □

Note also that the similarity transformation does not change the norm of a vector; thus, constraint (3.29) is equivalent to $\|\mathbf{s}_i - \mathbf{c}_i\| \leq r_i$, with $\mathbf{c}_i = \mathbf{C}\left(-\varphi_0\right)\left[\mathbf{p}_{S_i}(k) - \hat{\mathbf{p}}_T(k+1|k)\right]$, and constraint (3.30) is equivalent to $\|\mathbf{s}_i\| \geq \rho_i$. Therefore, Π_3 is equivalent to the following optimization problem:

- Optimization Problem 4 (Π_4)

$$\underset{\mathbf{s}_1, \ldots, \mathbf{s}_M}{\text{minimize}} \quad \mathrm{tr}\left(\mathbf{\Lambda} + \sum_{i=1}^{M} \frac{\kappa_{d_i}}{\sigma_{d_i}^2} \frac{\mathbf{s}_i \mathbf{s}_i^{\mathrm{T}}}{\mathbf{s}_i^{\mathrm{T}} \mathbf{s}_i} + \sum_{i=1}^{M} \frac{\kappa_{\theta_i}}{\sigma_{\theta_i}^2} \frac{\mathbf{J}\mathbf{s}_i \mathbf{s}_i^{\mathrm{T}} \mathbf{J}^{\mathrm{T}}}{(\mathbf{s}_i^{\mathrm{T}} \mathbf{s}_i)^2}\right)^{-1} \tag{3.33}$$

$$\text{subject to} \quad \|\mathbf{s}_i - \mathbf{c}_i\|^2 \leq r_i^2, \tag{3.34}$$

$$\|\mathbf{s}_i\|^2 \geq \rho_i^2, \qquad i = 1, \ldots, M. \tag{3.35}$$

Once the optimal solution $\{\mathbf{s}_i,\ i = 1, \ldots, M\}$ is obtained, the best sensing location for sensor-i at time-step $k+1$, $\mathbf{p}_{S_i}(k+1)$, can be calculated through $\mathbf{p}_i = \mathbf{C}(\varphi_0)\mathbf{s}_i$ and (3.5).

Remark 8. *The optimization problem* Π_4 *is a nonlinear programming problem since both the objective function [see* (3.33)*] and constraints [see* (3.34)$-$(3.35)*] are nonlinear functions with respect to the optimization variable* $\mathbf{s} = \left[\mathbf{s}_1^{\mathrm{T}} \ \ldots \ \mathbf{s}_M^{\mathrm{T}}\right]^{\mathrm{T}}$*. Moreover,* Π_4 *(and equivalently,* Π_3*) is not a convex program since the objective function* (3.33) *is non-convex with respect to* $\mathbf{s}$*, and the feasible set defined by constraint* (3.35) *is not convex.*

Next (see Section 3.4), we analyze the complexity of determining the optimal solution for one-step-ahead, multi-sensor active target tracking, described by the nonlinear constrained optimization problem Π_4 (or equivalently, Π_3). Our key contribution is to show that given distance-only observations, the corresponding optimization problem, when considering maximum-speed constraints, is *NP-Hard*. Therefore, we conclude that the more general problem Π_4 (or equivalently, Π_3) is also *NP-Hard* in general.

3.4 Complexity analysis

In this section, our objective is to determine the computational complexity of seeking the next best sensing locations for multiple robots tracking a moving target under maximum speed constraints [see (3.34)] when using distance-only measurements. Mathematically, this is described by the following optimization problem Π_5 (formulated by setting $\kappa_{d_i} = 1, \kappa_{\theta_i} = 0, i = 1, \ldots, M$ and removing the target avoidance constraints (3.35) in Π_4):

- Optimization Problem 5 (Π_5)

$$\underset{\mathbf{s}_1,\ldots,\mathbf{s}_M}{\text{minimize}} \quad \text{tr}\left(\mathbf{\Lambda} + \sum_{i=1}^{M} \frac{1}{\sigma_{d_i}^2} \frac{\mathbf{s}_i \mathbf{s}_i^{\mathrm{T}}}{\mathbf{s}_i^{\mathrm{T}} \mathbf{s}_i}\right)^{-1} \tag{3.36}$$

$$\text{subject to} \quad \|\mathbf{s}_i - \mathbf{c}_i\|^2 \leq r_i^2, \qquad i = 1, \ldots, M. \tag{3.37}$$

Our objective is to prove that the aforementioned optimization problem Π_5 is *NP-Hard* in general.

To proceed, we firstly notice that the term $\frac{\mathbf{s}_i \mathbf{s}_i^{\mathrm{T}}}{\mathbf{s}_i^{\mathrm{T}} \mathbf{s}_i}$ is independent of the relative distance $\|\mathbf{s}_i\|$ between the target and sensor-i ($i = 1, \ldots, M$). Thus, the cost function (3.36) can be expressed as a function of $\bar{\theta}_i, i = 1, \ldots, M$, where $\bar{\theta}_i$ is the angular coordinate (also called polar angle, or azimuth) of $\mathbf{s}_i$ (see Figure 3.3). On the other hand, from the geometry of Figure 3.3, the constraint on $\mathbf{s}_i$ [see (3.37)] can be transformed into an interval constraint on $\bar{\theta}_i$, i.e., $\bar{\theta}_i \in [\bar{\theta}_{i\,\min}, \bar{\theta}_{i\,\max}]$ with

$$\bar{\theta}_{i\,\min} = \bar{\theta}_{\mathbf{c}_i} - \arcsin\left(\frac{r_i}{\|\mathbf{c}_i\|}\right), \quad \bar{\theta}_{i\,\max} = \bar{\theta}_{\mathbf{c}_i} + \arcsin\left(\frac{r_i}{\|\mathbf{c}_i\|}\right), \quad i = 1, \ldots, M, \tag{3.38}$$

where $\bar{\theta}_{\mathbf{c}_i}$ is the angular coordinate of the known vector $\mathbf{c}_i$, $i = 1, \ldots, M$.

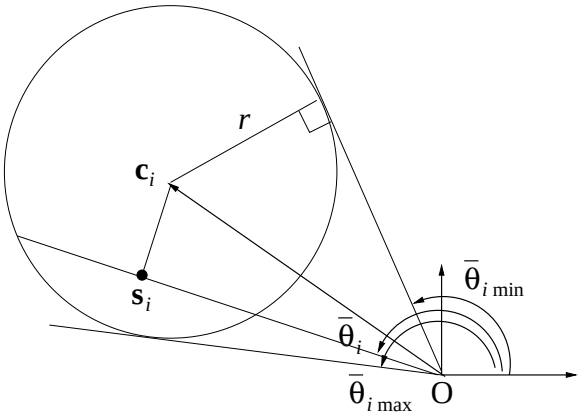


Figure 3.3: Geometric interpretation of the polar angle constraints: The circular constraint on $\mathbf{s}_i$ is transformed into an interval constraint on the polar angle $\bar{\theta}_i$, i.e., $\bar{\theta}_i \in [\bar{\theta}_{i\,\min}, \bar{\theta}_{i\,\max}], i = 1 \ldots, M$. Also note that the objective function for the distance-only target tracking problem depends only on the polar angles $\bar{\theta}_i, i = 1, \ldots, M$. In other words, the cost function remains the same for arbitrary point $\mathbf{s}_i$ located along the ray with angle $\bar{\theta}_i$, $i = 1, \ldots, M$.

In what follows, we seek the explicit form of the objective function (3.36) with respect to the polar angles $\bar{\theta}_i, i = 1, \ldots, M$. To do so, we substitute $\mathbf{s}_i = \|\mathbf{s}_i\| \begin{bmatrix} \cos(\bar{\theta}_i) & \sin(\bar{\theta}_i) \end{bmatrix}^{\mathrm{T}}$, $i = 1, \ldots, M$ into (3.36), and perform algebraic manipulations where we invoke the trigonometric identities $\cos^2 \bar{\theta}_i = (1 + \cos 2\bar{\theta}_i)/2$, $\sin^2 \bar{\theta}_i = (1 - \cos 2\bar{\theta}_i)/2$, $\cos \bar{\theta}_i \sin \bar{\theta}_i = (\sin 2\bar{\theta}_i)/2$, to get

$$\mathrm{tr} \left(\mathbf{\Lambda} + \sum_{i=1}^{M} \frac{1}{\sigma_{d_i}^2} \frac{\mathbf{s}_i \mathbf{s}_i^{\mathrm{T}}}{\mathbf{s}_i^{\mathrm{T}} \mathbf{s}_i} \right)^{-1} = \frac{d_{c_1}}{d_{c_2} - \frac{1}{4} d_{\bar{\theta}}}, \tag{3.39}$$

where

$$d_{c_1} = \lambda_1^{-1} + \lambda_2^{-1} + \sum_{i=1}^{M} \sigma_{d_i}^{-2},$$

$$d_{c_2} = (\lambda_1^{-1} + \frac{1}{2} \sum_{i=1}^{M} \sigma_{d_i}^{-2})(\lambda_2^{-1} + \frac{1}{2} \sum_{i=1}^{M} \sigma_{d_i}^{-2}) + \frac{1}{4}(\lambda_1^{-1} - \lambda_2^{-1})^2$$

are constant, and

$$d_{\bar{\theta}} = \left(\left(\lambda_1^{-1} - \lambda_2^{-1}\right) + \sum_{i=1}^{M} \sigma_{d_i}^{-2} \cos(2\bar{\theta}_i)\right)^2 + \left(\sum_{i=1}^{M} \sigma_{d_i}^{-2} \sin(2\bar{\theta}_i)\right)^2$$
$$= \left|\left|\bar{\lambda}_0 \exp\left(\jmath\,\pi\right) + \sum_{i=1}^{M} \bar{\lambda}_i \exp\left(\jmath\, 2\bar{\theta}_i\right)\right|\right|_2^2,$$

where $\jmath = \sqrt{-1}$, $\bar{\lambda}_0 = \lambda_2^{-1} - \lambda_1^{-1} \geq 0$, and $\bar{\lambda}_i = \sigma_{d_i}^{-2} > 0$.

Based on the expression of (3.39), we immediately obtain the following key lemma.

Lemma 9. *The optimization problem* Π_5 *is equivalent to the following constrained 2-norm minimization problem* Π_6 *optimized over the polar angles* $\bar{\theta}_i, i = 1, \ldots, M$*:*

- Optimization Problem 6 (Π_6)

$$\underset{\bar{\theta}_1,\ldots,\bar{\theta}_M}{\text{minimize}} \quad \left|\left|\bar{\lambda}_0 \exp\left(\jmath\,\pi\right) + \sum_{i=1}^{M} \bar{\lambda}_i \exp\left(\jmath\, 2\bar{\theta}_i\right)\right|\right|_2^2 \tag{3.40}$$
$$\text{subject to} \quad \bar{\theta}_{i\,\min} \leq \bar{\theta}_i \leq \bar{\theta}_{i\,\max}, \qquad i = 1, \ldots, M, \tag{3.41}$$

with $\bar{\lambda}_0 = \lambda_2^{-1} - \lambda_1^{-1} \geq 0$, $\bar{\lambda}_i = \sigma_{d_i}^{-2} > 0, i = 1 \ldots, M$, *and* $\bar{\theta}_{i\,\min}$ *and* $\bar{\theta}_{i\,\max}$ *provided in* (3.38)*; or equivalently:*

$$\underset{\bar{\theta}_1,\ldots,\bar{\theta}_M}{\text{minimize}} \quad \left|\left|\sum_{i=0}^{M} \mathbf{v}_i\right|\right|_2$$
$$\text{subject to} \quad \bar{\theta}_{i\,\min} \leq \bar{\theta}_i \leq \bar{\theta}_{i\,\max}, \qquad i = 1, \ldots, M,$$

with (*for* $i = 1, \ldots, M$)

$$\mathbf{v}_0 = [-\bar{\lambda}_0 \quad 0]^{\mathrm{T}},\ \mathbf{v}_i = [\bar{\lambda}_i \cos(2\bar{\theta}_i) \quad \bar{\lambda}_i \sin(2\bar{\theta}_i)]^{\mathrm{T}}.$$

We thus see that the original problem of minimizing the trace of the posterior covariance matrix of the target's position estimate is *exactly reformulated* to that of *minimizing the norm of the sum of* $M + 1$ *vectors in* 2-D (see Lemma 9). Note that although the vector $\mathbf{v}_0 = [-\lambda_0 \quad 0]^{\mathrm{T}}$ remains constant (affixed to the negative x semi-axis), each of the vectors $\mathbf{v}_i,\ i = 1, \ldots, M$, has fixed length $\bar{\lambda}_i$ but its direction can vary under the constraints described by (3.41). This geometric interpretation is depicted in Figure 3.4.

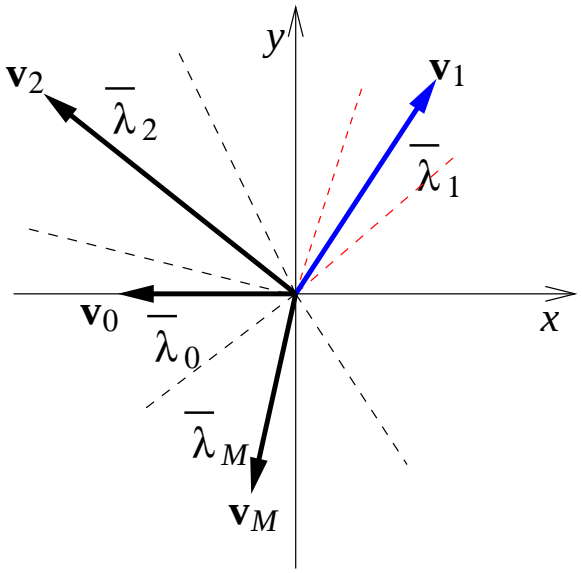


Figure 3.4: Geometric interpretation of the optimal motion strategy problem: The $M+1$ vectors shown have fixed lengths $\bar{\lambda}_i, i = 0, \ldots, M$. The vector $\mathbf{v}_0$ is affixed to the negative x axis, while the direction of each of the vectors $\mathbf{v}_i,\ i = 1, \ldots, M$, can change, within the interval denoted by the enclosing dashed lines, based on the motion of the corresponding sensor. The objective is to find the directions of the vectors $\mathbf{v}_i,\ i = 1, \ldots, M$ – directly related to the optimal polar angles of $\mathbf{s}_i,\ i = 1, \ldots, M$ – that minimize the Euclidean norm of $\sum_{i=0}^{M} \mathbf{v}_i$.

It is clear that the objective function in (3.40) is non-convex in the optimization variables $\bar{\theta}_i, i = 1, \ldots, M$. More importantly, in this section we show that the optimization problem Π_6 is *NP-Hard* in general.

We proceed by first considering the following well-known *NP-Complete* problem [90, Ch. 3]:

- Partition Problem
  Given M positive integers $\bar{\lambda}_1, \ldots, \bar{\lambda}_M$, determine whether there exist binary variables $\zeta_i \in \{-1, +1\}$, $i = 1, \ldots, M$, such that $\sum_{i=1}^{M} \bar{\lambda}_i \zeta_i = 0$.

and

- Optimization Problem 7 (Π_7)

$$\underset{\bar{\theta}_1,\ldots,\bar{\theta}_M}{\text{minimize}} \quad \left(\left(\sum_{i=1}^{M} \bar{\lambda}_i \cos(2\bar{\theta}_i)\right)^2 + \left(\sum_{i=1}^{M} \bar{\lambda}_i \sin(2\bar{\theta}_i)\right)^2\right)^{\frac{1}{2}} \tag{3.42}$$

$$\text{subject to} \quad 0 \leq \bar{\theta}_i \leq \pi/2, \qquad i = 1,\ldots,M, \tag{3.43}$$

$$\bar{\lambda}_i \in \mathbb{Z}^+, \qquad i = 1,\ldots,M,$$

which is an instance of optimization problem Π_6 described by (3.40)–(3.41), for $\bar{\lambda}_0 = 0$, $\bar{\theta}_{i\,\min} = 0$, $\bar{\theta}_{i\,\max} = \pi/2$ and $\bar{\lambda}_i \in \mathbb{Z}^+$.

Proving by restriction [90, Ch. 3] that Π_6 is *NP-Hard*, in general, requires to show that solving Π_7, which is a special case of Π_6, is *equivalent* to solving the partition problem. Since the partition problem is *NP-Complete*, it will follow that the general problem Π_6 is *at least* as hard as that, i.e., Π_6 is *NP-Hard*. We first prove that the answer to the partition problem is positive ("yes"), if and only if Π_7 achieves optimal value of zero.

Lemma 10. *For positive integers $\bar{\lambda}_1,\ldots,\bar{\lambda}_M$, there exist $\zeta_i \in \{-1,+1\}$, $i = 1,\ldots,M$, such that $\sum_{i=1}^{M} \bar{\lambda}_i\zeta_i = 0$,* if *and* only if*, the optimal value of Π_7 is 0.*

Proof. (*Necessary*): Assume $\exists\ \zeta_i \in \{-1,+1\}, i = 1,\ldots,M$, such that

$$\sum_{i=1}^{M} \bar{\lambda}_i\zeta_i = 0. \tag{3.44}$$

Based on these, consider the following choice of $\bar{\theta}_i$ for Π_7

$$\bar{\theta}_i^* = \begin{cases} 0 & \text{if} \quad \zeta_i = 1 \\ \pi/2 & \text{if} \quad \zeta_i = -1 \end{cases} \tag{3.45}$$

Note that $\bar{\theta}_i^*$, $i = 1,\ldots,M$, satisfies the constraints of Π_7 [see (3.43)]. Additionally, it is easy to verify that $\cos(2\bar{\theta}_i^*) = \zeta_i$ and $\sin(2\bar{\theta}_i^*) = 0$, $i = 1,\ldots,M$. Substituting in the objective function (squared) of Π_7 [see (3.42)] yields

$$\left(\sum_{i=1}^{M} \bar{\lambda}_i \cos(2\bar{\theta}_i^*)\right)^2 + \left(\sum_{i=1}^{M} \bar{\lambda}_i \sin(2\bar{\theta}_i^*)\right)^2 = \left(\sum_{i=1}^{M} \bar{\lambda}_i\zeta_i\right)^2 = 0,$$

where the last equality follows from (3.44).

Since the objective function of Π_7 is always nonnegative and the choice of $\bar{\theta}_i^*$ [see (3.45)] based on ζ_i achieves zero, the set $\{\ \bar{\theta}_i^*,\ i = 1, \ldots, M\ \}$ is the optimal solution of Π_7.

(*Sufficient*): Suppose $\exists\ \bar{\theta}_i^*$, with $0 \leq \bar{\theta}_i^* \leq \pi/2$, $i = 1, \ldots, M$, and

$$\left(\left(\sum_{i=1}^{M} \bar{\lambda}_i \cos(2\bar{\theta}_i^*) \right)^2 + \left(\sum_{i=1}^{M} \bar{\lambda}_i \sin(2\bar{\theta}_i^*) \right)^2 \right)^{\frac{1}{2}} = 0. \tag{3.46}$$

This last equality for the objective function of Π_7 requires

$$\sum_{i=1}^{M} \bar{\lambda}_i \sin(2\bar{\theta}_i^*) = 0, \tag{3.47}$$

$$\sum_{i=1}^{M} \bar{\lambda}_i \sin(2\bar{\theta}_i^*) = 0. \tag{3.48}$$

Note that the constraints on $\bar{\theta}_i$ [see (3.43)] imply that $\sin(2\bar{\theta}_i^*) \geq 0$, $i = 1, \ldots, M$. Additionally, since $\bar{\lambda}_i > 0$, it follows from (3.47) that $\sin(2\bar{\theta}_i^*) = 0 \Rightarrow \cos(2\bar{\theta}_i^*) = \pm 1$, $i = 1, \ldots, M$. Thus, there exist $\zeta_i = \cos(2\bar{\theta}_i^*) \in \{-1, +1\}$, such that $\sum_{i=1}^{M} \bar{\lambda}_i \zeta_i = \sum_{i=1}^{M} \bar{\lambda}_i \cos(2\bar{\theta}_i^*) = 0$ [see (3.48)]. □

Lemma 10, establishes a one-to-one correspondence between every instance of Π_7 and that of the partition problem.[4] In particular, if we are able to solve the optimization problem Π_7, then by examining its optimal value, we can answer the partition problem, i.e., zero (vs. positive) optimal value for the objective function of Π_7 corresponds to positive (vs. negative) answer to the partition problem. Based on the result of Lemma 10, we hereafter state and prove the main result of this section.

Theorem 11. *The optimization problem* Π_6 (*or equivalently* Π_5) *of generating the one-step-ahead optimal motion strategy for a team of mobile sensors tracking a moving target using distance-only measurements under mobility constraints is* NP-Hard *in general.*

Proof. Assume that the general problem Π_6 (or equivalently Π_5) is *not NP-Hard*. Then there exists a polynomial-time algorithm that can solve all instances of Π_6, and hence Π_7. From Lemma 10, however, the answer to the partition problem can be determined

[4] Note that the parameters for both the partition problem and the optimization problem Π_7 are $\bar{\lambda}_1, \ldots, \bar{\lambda}_M$. An instance of these two problems is obtained by specifying particular values for $\bar{\lambda}_1, \ldots, \bar{\lambda}_M$.

based on the optimal value of Π_7. This implies that the partition problem can be solved in polynomial time, which is a contradiction. □

Note that since Π_5 (or Π_6) is a special case of Π_4 (or equivalently of Π_3), from Theorem 11, we immediately obtain the following theorem regarding the worst-case computational complexity of Π_3 (or equivalently of Π_4):

Theorem 12. *The optimization problem* Π_3 *of generating the one-step-ahead optimal motion strategy for a team of mobile sensors tracking a moving target using combinations of relative* (*distance and/or bearing*) *observations under mobility constraints is* NP-Hard *in general.*

Theorem 12 establishes the fact that the problem of *optimal trajectory generation* for multiple sensors with mobility constraints that track a moving target using combinations of relative observations (i.e., distance and/or bearing), is *NP-Hard* in general. Hence, finding the *global* optimal solution of Π_3 or Π_4 is extremely challenging. Ideally, the optimal solution can be determined if one discretizes the feasible set of all sensors [see (3.34)-(3.35)] and performs an exhaustive search. This approach, however, has computational complexity *exponential* in the number of sensors, which is of limited practical use given realistic processing constraints.

In order to design algorithms that can operate in real time, appropriate relaxations of Π_4 become necessary. In what follows, we first derive analytically the optimal solution for the single-sensor case (see Section 3.5) and based on that we propose a *Gauss-Seidel relaxation* (GSR) to solve the general problem of multiple sensors (see Section 3.6), which has computational complexity *linear* in the number of sensors.

3.5 Single-sensor active target tracking

For $M = 1$, the optimization problem Π_4 described by (3.33)–(3.35) is simplified to:[5]

- Optimization Problem 8 (Π_8)

[5] To simplify notation, we drop the indices of $\mathbf{s}_1$, σ_{d_1}, σ_{θ_1}, κ_{d_1}, κ_{θ_1}, $\mathbf{c}_1$, r_1, and ρ_1.

$$\underset{\mathbf{s}}{\text{minimize}} \quad f_0(\mathbf{s}) = \text{tr}\left(\mathbf{\Lambda} + \frac{\kappa_d}{\sigma_d^2}\frac{\mathbf{s}\mathbf{s}^{\text{T}}}{\mathbf{s}^{\text{T}}\mathbf{s}} + \frac{\kappa_\theta}{\sigma_\theta^2}\frac{\mathbf{J}\mathbf{s}\mathbf{s}^{\text{T}}\mathbf{J}^{\text{T}}}{(\mathbf{s}^{\text{T}}\mathbf{s})^2}\right)^{-1} \tag{3.49}$$

$$\text{subject to} \quad \|\mathbf{s}-\mathbf{c}\|^2 \le r^2, \tag{3.50}$$

$$\|\mathbf{s}\|^2 \ge \rho^2. \tag{3.51}$$

In order to solve Π_8, we proceed as follows: We first determine *all critical/stationary points* (i.e., those points which satisfy the Karush-Kuhn-Tucker (KKT) necessary optimality conditions [64, Ch. 3]) analytically and evaluate their objective values. Then, as optimal solution for Π_8 we select the critical point whose objective value is the smallest.

To proceed, we first construct the Lagrange function [64]:

$$L(\mathbf{s},\mu,\nu) = f_0(\mathbf{s}) + \frac{\mu}{2}\left(\|\mathbf{s}-\mathbf{c}\|^2 - r^2\right) + \frac{\nu}{2}\left(\rho^2 - \|\mathbf{s}\|^2\right).$$

Based on the KKT necessary conditions, the critical points $\mathbf{s}^*$, and the associated Lagrange multipliers μ^* and ν^*, must satisfy:

$$\nabla f_0(\mathbf{s}^*) + \mu^*(\mathbf{s}^*-\mathbf{c}) - \nu^*\mathbf{s}^* = \mathbf{0}_{2\times 1}, \tag{3.52}$$

$$\mu^* \ge 0, \quad \mu^*\left(\|\mathbf{s}^*-\mathbf{c}\|^2 - r^2\right) = 0, \tag{3.53}$$

$$\nu^* \ge 0, \quad \nu^*\left(\rho^2 - \|\mathbf{s}^*\|^2\right) = 0. \tag{3.54}$$

Clearly (3.53)–(3.54) are degree-3 multivariate *polynomial* equations in the unknowns $\mathbf{s}^*$, μ^* and ν^*. Furthermore, as shown in Appendix A.1, both f_0 and its derivative ∇f_0 are *rational* functions with respect to $\mathbf{s}^*$, and thus (3.52) can be transformed into a *polynomial* equality in $\mathbf{s}^*$, μ^*, and ν^*. Therefore, computing all critical points of Π_5 is equivalent to solving the polynomial system defined by (3.52)–(3.54). Moreover, it is worth mentioning that unlike linear systems, in general there exist *multiple solutions* for the above polynomial system. In order to efficiently solve (3.52)–(3.54), we first prove the following lemma:

Lemma 13. *Assume $\bar{\Omega} = \Omega \cup \partial\Omega$ is a compact and connected set*[6] *in 2D, and the origin $O = [0\ 0]^{\text{T}} \notin \bar{\Omega}$. For any $\mathbf{s} \in \Omega$, the line segment connecting $\mathbf{s}$ and the origin will*

[6] Ω stands for the open set consisting of all interior points of $\bar{\Omega}$, while $\partial\Omega$ and $\bar{\Omega}$ represent its boundary and closure, respectively.

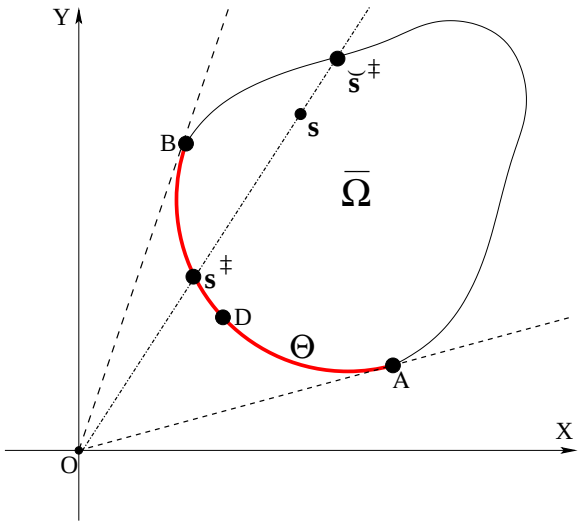


Figure 3.5: Geometric illustration of Lemma 13. The global optimal solution resides only in Θ, i.e., the portion of the boundary of the feasible set $\bar{\Omega}$ (depicted by the red-colored curve ADB), defined by the two tangent lines OA and OB, which is closest to O.

intersect $\partial\Omega$ *at one or multiple points. Let* $\mathbf{s}^{\ddagger} \in \partial\Omega$ *denotes the closest intersection to the origin* (*see Figure 3.5*)*, then* $f_0(\mathbf{s}^{\ddagger}) \leq f_0(\mathbf{s})$.

Proof. Based on the construction of $\mathbf{s}^{\ddagger}$, we have $\mathbf{s}^{\ddagger} = \varkappa\mathbf{s}$, with $\varkappa \in (0, 1)$, and thus:

$$\frac{(\mathbf{s}^{\ddagger})(\mathbf{s}^{\ddagger})^{\mathrm{T}}}{(\mathbf{s}^{\ddagger})^{\mathrm{T}}(\mathbf{s}^{\ddagger})} = \frac{\mathbf{s}\mathbf{s}^{\mathrm{T}}}{\mathbf{s}^{\mathrm{T}}\mathbf{s}}\ , \ \frac{\mathbf{J}(\mathbf{s}^{\ddagger})(\mathbf{s}^{\ddagger})^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{((\mathbf{s}^{\ddagger})^{\mathrm{T}}(\mathbf{s}^{\ddagger}))^2} = \frac{1}{\varkappa^2}\frac{\mathbf{J}\mathbf{s}\mathbf{s}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{(\mathbf{s}^{\mathrm{T}}\mathbf{s})^2} \succeq \frac{\mathbf{J}\mathbf{s}\mathbf{s}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{(\mathbf{s}^{\mathrm{T}}\mathbf{s})^2}$$

$$\Rightarrow \quad \left(\mathbf{\Lambda} + \frac{\kappa_d}{\sigma_d^2}\frac{(\mathbf{s}^{\ddagger})(\mathbf{s}^{\ddagger})^{\mathrm{T}}}{(\mathbf{s}^{\ddagger})^{\mathrm{T}}(\mathbf{s}^{\ddagger})} + \frac{\kappa_\theta}{\sigma_\theta^2}\frac{\mathbf{J}(\mathbf{s}^{\ddagger})(\mathbf{s}^{\ddagger})^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{((\mathbf{s}^{\ddagger})^{\mathrm{T}}(\mathbf{s}^{\ddagger}))^2}\right)^{-1} \preceq \left(\mathbf{\Lambda} + \frac{\kappa_d}{\sigma_d^2}\frac{\mathbf{s}\mathbf{s}^{\mathrm{T}}}{\mathbf{s}^{\mathrm{T}}\mathbf{s}} + \frac{\kappa_\theta}{\sigma_\theta^2}\frac{\mathbf{J}\mathbf{s}\mathbf{s}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{(\mathbf{s}^{\mathrm{T}}\mathbf{s})^2}\right)^{-1}$$

$$\Rightarrow \quad f_0(\mathbf{s}^{\ddagger}) \leq f_0(\mathbf{s}).$$

□

Remark 14. *Lemma 13 establishes the fact that the global optimal solution for* Π_8*, when optimizing over the feasible set* $\bar{\Omega}$ (*see Figure 3.5*)*, is always on its* boundary $\partial\Omega$*, defined by* (3.50)–(3.51)*, i.e.,* $\mathbf{s}^*$ *satisfies either* $\|\mathbf{s}^* - \mathbf{c}\| = r$ *or* $\|\mathbf{s}^*\| = \rho$*. Moreover,*

by applying the same argument as before (see Figure 3.5), it can be easily shown that $f_0(\mathbf{s}^\ddagger) \leq f_0(\breve{\mathbf{s}}^\ddagger)$*, where* $\breve{\mathbf{s}}^\ddagger$ *is any other intersection point in the direction of* $\mathbf{s}^\ddagger$*. Therefore, the global optimal solution* $\mathbf{s}^*$ *resides only in the portion of* $\partial\Omega$ *facing the origin, denoted as* Θ *(see Figure 3.5).*[7]

As shown in Figures 3.6(a)–3.6(d), depending on the values of the parameters $\mathbf{c}$, r, and ρ, there exist four cases that we need to consider for the feasible set $\bar{\Omega}$ of Π_5. In what follows, we *analytically* solve the KKT conditions (3.52)–(3.54) for each of the first three cases [see Figures 3.6(a)–3.6(c)], while for the fourth case [see Figure 3.6(d)], we propose a strategy for handling the empty (or infeasible) set $\bar{\Omega}$. In the ensuing derivations, we use the definitions $\mathbf{s}^* := \begin{bmatrix} x & y \end{bmatrix}^{\mathrm{T}}$ and $\mathbf{c} := \begin{bmatrix} c_1 & c_2 \end{bmatrix}^{\mathrm{T}}$.

3.5.1 Case I: $0 < \rho \leq \|\mathbf{c}\| - r$

As shown in Figure 3.6(a), the only *active* constraint for Case I is the maximum-speed constraint [see (3.50)]. Based on Lemma 13 and setting $v = v_{\max}$, the optimal solution $\mathbf{s}^*$ must reside in the arc ADB, where A and B are two tangent points, whose Cartesian coordinates are computed later on [see (3.70)]. Since the collision-avoidance constraint (3.51) is *inactive*, its corresponding Lagrange multiplier $\nu^* = 0$, and the system of (3.52)–(3.54) is simplified to:

$$\nabla f_0(\mathbf{s}^*) + \mu^* (\mathbf{s}^* - \mathbf{c}) = \mathbf{0}_{2\times 1}, \tag{3.55}$$

$$\|\mathbf{s}^* - \mathbf{c}\|^2 - r^2 = 0. \tag{3.56}$$

Clearly, (3.56) is a 2nd-order polynomial equation in the variables x and y, i.e.,

$$0 = f_2(x, y) = (x - c_1)^2 + (y - c_2)^2 - r^2. \tag{3.57}$$

Since we aim at transforming (3.55) into a polynomial equation only containing x and y, we eliminate μ^* by multiplying both sides of (3.55) with $(\mathbf{s}^* - \mathbf{c})^{\mathrm{T}} \mathbf{C}\left(\frac{\pi}{2}\right)$, which yields:

$$(\mathbf{s}^* - \mathbf{c})^{\mathrm{T}} \mathbf{C}\left(\frac{\pi}{2}\right) \nabla f_0(\mathbf{s}^*) = 0. \tag{3.58}$$

[7] It is straightforward to extend and generalize Lemma 13 to the multi-sensor case and conclude that the global optimal solution $\{\mathbf{s}_i^*, i = 1, \ldots, M\}$ for Π_4 is also always on the *boundaries* of the corresponding feasible sets defined by (3.34)–(3.35), i.e., $\mathbf{s}_i^*$ satisfies either $\|\mathbf{s}_i^* - \mathbf{c}_i\| = r_i$ or $\|\mathbf{s}_i^*\| = \rho_i, \forall i = 1, \ldots, M$.

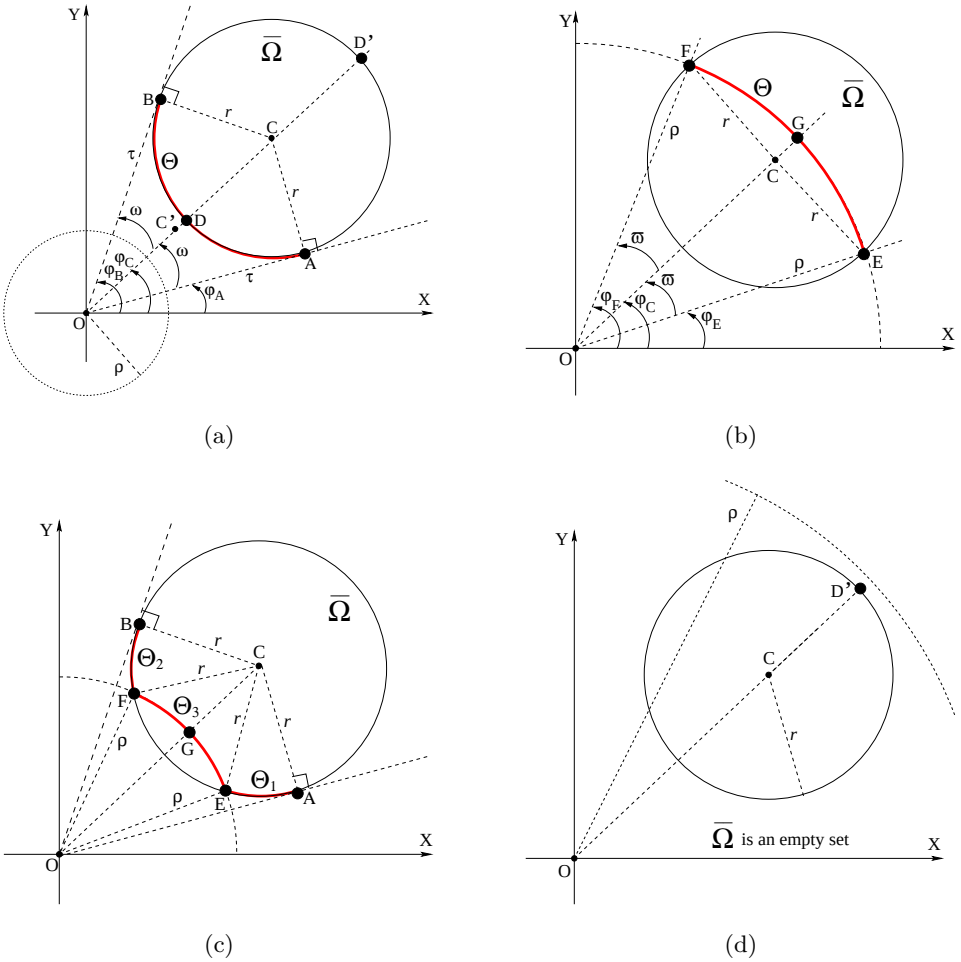


Figure 3.6: Four cases of the feasible set $\bar{\Omega}$. (a) Case I: $0 < \rho \leq \|\mathbf{c}\| - r$. (b) Case II: $\sqrt{\|\mathbf{c}\|^2 - r^2} \leq \rho < \|\mathbf{c}\| + r$. (c) Case III: $\|\mathbf{c}\| - r < \rho < \sqrt{\|\mathbf{c}\|^2 - r^2}$. (d) Case IV: $\|\mathbf{c}\| + r \leq \rho$, which corresponds to the feasible set $\bar{\Omega}$ being empty. In the first three cases (a)-(c), the global optimal solution resides in a subset Θ of the boundary of $\bar{\Omega}$, which is depicted by the red-colored curve ADB in Case I, EGF in Case II, $AEGFB$ in Case III, respectively. In the above plots, O is the origin; C is the center of the circle $\|\mathbf{s} - \mathbf{c}\| = r$; A and B are the two tangent points residing in the circle $\|\mathbf{s} - \mathbf{c}\| = r$; E and F are the intersection points of the two circles $\|\mathbf{s} - \mathbf{c}\| = r$ and $\|\mathbf{s}\| = \rho$; the ray starting from O and passing through C intersects the circle $\|\mathbf{s}\| = \rho$ at G, and the circle $\|\mathbf{s} - \mathbf{c}\| = r$ at D and D'. Finally C' is the midpoint between O and C.

Note that (3.58) is equivalent to the following bivariate polynomial equation (see Appendix A.2):

$$0 = f_1(x,y) = \beta_3 xy\Delta^3 + (\alpha_8 x + \alpha_7 y + \beta_2)xy\Delta^2 + (\alpha_6 x^3 + \alpha_5 x^2 y + \alpha_4 xy^2 + \alpha_3 y^3 + \beta_1 xy)\Delta + (\alpha_2 x + \alpha_1 y)xy, \tag{3.59}$$

where $\Delta = x^2 + y^2$, and the parameters $\beta_i,\ i = 1,2,3$, and $\alpha_j,\ j = 1,\ldots,8$, are known coefficients expressed in terms of $\lambda_1, \lambda_2, c_1, c_2, \kappa_d\sigma_d^{-2}$, and $\kappa_\theta\sigma_\theta^{-2}$.

In order to obtain all the critical points of Π_8, we need to solve the system of polynomial equations $f_1(x,y) = 0$ *and* $f_2(x,y) = 0$ analytically [see (3.57) and (3.59)]. Although $f_2(x,y)$ is independent of the measurement type, $f_1(x,y)$ is a function of κ_d and κ_θ. Additionally, as it will become evident, the total degree of $f_1(x,y)$ depends on $\lambda_1 - \lambda_2$. (Note that in Lemma 7 it is assumed that $\lambda_1 \geq \lambda_2$). In what follows, we first present the solution of the system of bivariate polynomial equations (3.57) and (3.59) under the assumption $\lambda_1 > \lambda_2$ for each different type of measurement, and then address the case of $\lambda_1 = \lambda_2$.

Distance-and-bearing observations

When the sensor measures both distance and bearing to the target, or equivalently, $\kappa_d = \kappa_\theta = 1$, it can be shown (see Appendix A.3) that $\beta_i \neq 0, i = 1,2,3$, and $\alpha_j \neq 0, j = 1,\ldots,8$. Therefore, f_1 [see (3.59)] is an 8th-order polynomial in the variables x and y.

To solve $f_1 = f_2 = 0$ analytically, we first treat x as a parameter and rewrite (3.59) as a sum of y-monomials in decreasing order:

$$f_1 = \chi_7 y^7 + \chi_6 y^6 + \chi_5 y^5 + \chi_4 y^4 + \chi_3 y^3 + \chi_2 y^2 + \chi_1 y + \chi_0, \tag{3.60}$$

where χ_i, $i = 0,\ldots,7$, are coefficients expressed in terms of $\lambda_1, \lambda_2, c_1, c_2, \sigma_d^{-2}, \sigma_\theta^{-2}$, and x (see Appendix A.3 for the specific expressions of χ_i, $i = 0,\ldots,7$).

Similarly, (3.57) can be rewritten as:

$$f_2 = \eta_2 y^2 + \eta_1 y + \eta_0, \tag{3.61}$$

where

$$\eta_2 = 1, \quad \eta_1 = -2c_2, \quad \eta_0 = x^2 - 2c_1 x + c_1^2 + c_2^2 - r^2. \tag{3.62}$$

Thus, the Sylvester matrix of f_1 and f_2 with respect to y, denoted as $\mathbf{Syl}(f_1, f_2; y)$, is the following 9×9 matrix [76, Ch. 3]:

$$\mathbf{Syl}(f_1, f_2; y) = \begin{bmatrix} \chi_7 & & \eta_2 & & & & & & \\ \chi_6 & \chi_7 & \eta_1 & \eta_2 & & & & & \\ \chi_5 & \chi_6 & \eta_0 & \eta_1 & \eta_2 & & & & \\ \chi_4 & \chi_5 & & \eta_0 & \eta_1 & \eta_2 & & & \\ \chi_3 & \chi_4 & & & \eta_0 & \eta_1 & \eta_2 & & \\ \chi_2 & \chi_3 & & & & \eta_0 & \eta_1 & \eta_2 & \\ \chi_1 & \chi_2 & & & & & \eta_0 & \eta_1 & \eta_2 \\ \chi_0 & \chi_1 & & & & & & \eta_0 & \eta_1 \\ & \chi_0 & & & & & & & \eta_0 \end{bmatrix}.$$

The resultant of f_1 and f_2 with respect to y, denoted as $\mathbf{Res}(f_1, f_2; y)$, is the determinant of the Sylvester matrix $\mathbf{Syl}(f_1, f_2; y)$. Furthermore, note that since χ_i, $i = 0, \ldots, 7$, and η_0 are polynomials of x, $\mathbf{Res}(f_1, f_2; y)$ is also a polynomial of x only. Hence, by employing the Sylvester resultant [76, Ch. 3], we are able to eliminate variable y from (3.60) and (3.61), and obtain the following 10th-order univariate polynomial in variable x:

$$0 = f_3(x) = \mathbf{Res}(f_1, f_2; y) = \det\big(\mathbf{Syl}(f_1, f_2; y)\big) = \sum_{j=0}^{10} \gamma_j x^j, \tag{3.63}$$

where $\gamma_j,\ j = 0, \ldots, 10$, are known coefficients expressed in terms of $\lambda_1, \lambda_2, c_1, c_2, \sigma_d^{-2}$, σ_θ^{-2}, and r.

The roots of the univariate polynomial f_3 correspond to the 10 eigenvalues of the associated 10×10 companion matrix $\mathbf{\Gamma}$ [77]:

$$\mathbf{\Gamma} = \begin{bmatrix} 0 & & & -\gamma_0/\gamma_{10} \\ 1 & 0 & & -\gamma_1/\gamma_{10} \\ & \ddots & & \vdots \\ & & 1 & -\gamma_9/\gamma_{10} \end{bmatrix}.$$

Note also that we only need to consider the real solutions of (3.63). Once x is determined, y is computed from (3.57), which can have at most 2 real solutions for every real solution x. In addition, from Lemma 13, we only need to consider those

critical points *belonging to* the arc ADB. Thus the set $\bar{\Xi}_1$ consisting of all critical points $\mathbf{s}^* = [x\ y]^{\mathrm{T}}$, has at most 20 elements.

The final step is to evaluate the objective function $f_0(\mathbf{s})$ [see (3.49)] at all the critical points in $\bar{\Xi}_1$ and select the one with the smallest objective value as the global optimal solution of Π_8, for the case $\kappa_d = \kappa_\theta = 1, \lambda_1 > \lambda_2$, and $\rho \leq \|\mathbf{c}\| - r$.

Bearing-only observation

When only a bearing measurement is available, i.e., $\kappa_d = 0, \kappa_\theta = 1$, it can be shown (see Appendix A.4) that $\beta_3 = \alpha_8 = \alpha_7 = 0$, and $\beta_2 > 0$. Thus, $f_1(x, y)$ [see (3.59)] can be simplified into the following 6th-order bivariate polynomial:

$$\begin{aligned} 0 = f_1(x,y) = {} & \beta_2 xy\Delta^2 + (\alpha_6 x^3 + \alpha_5 x^2 y + \alpha_4 xy^2 + \alpha_3 y^3 + \beta_1 xy)\Delta \\ & + (\alpha_2 x + \alpha_1 y)xy. \end{aligned} \tag{3.64}$$

Similarly to the case of distance-and-bearing observations, we rewrite f_1 as:

$$f_1 = \zeta_5 y^5 + \zeta_4 y^4 + \zeta_3 y^3 + \zeta_2 y^2 + \zeta_1 y + \zeta_0, \tag{3.65}$$

where $\zeta_i,\ i = 0, \ldots, 5$, are coefficients expressed in terms of $\lambda_1, \lambda_2, c_1, c_2, \sigma_\theta^{-2}$, and x (see Appendix A.4).

The Sylvester matrix of f_1 and f_2 [see (3.57) and (3.65)] with respect to y is the following 7×7 matrix, where η_0, η_1, η_2 are defined in (3.62):

$$\mathbf{Syl}(f_1, f_2; y) = \begin{bmatrix} \zeta_5 & & \eta_2 & & & & \\ \zeta_4 & \zeta_5 & \eta_1 & \eta_2 & & & \\ \zeta_3 & \zeta_4 & \eta_0 & \eta_1 & \eta_2 & & \\ \zeta_2 & \zeta_3 & & \eta_0 & \eta_1 & \eta_2 & \\ \zeta_1 & \zeta_2 & & & \eta_0 & \eta_1 & \eta_2 \\ \zeta_0 & \zeta_1 & & & & \eta_0 & \eta_1 \\ & \zeta_0 & & & & & \eta_0 \end{bmatrix}.$$

The resultant of f_1 and f_2 with respect to y is a 6th-order univariate polynomial:

$$0 = f_3(x) = \mathbf{Res}(f_1, f_2; y) := \det\big(\mathbf{Syl}(f_1, f_2; y)\big) = \sum_{j=0}^{6} \psi_j x^j, \tag{3.66}$$

where $\psi_j,\ j = 0, \ldots, 6$, are known coefficients expressed in terms of $\lambda_1, \lambda_2, c_1, c_2, \sigma_\theta^{-2}$, and r. The real roots of f_3 are the real eigenvalues of the 6×6 companion matrix $\mathbf{\Psi}$:

$$\mathbf{\Psi} = \begin{bmatrix} 0 & & & -\psi_0/\psi_6 \\ 1 & 0 & & -\psi_1/\psi_6 \\ & \ddots & & \vdots \\ & & 1 & -\psi_5/\psi_6 \end{bmatrix}.$$

Once x is determined, y can be computed from (3.57), and those pairs of $[x\ y]^{\mathrm{T}}$ falling on the arc ADB are included in the set $\bar{\Xi}_1$, which has at most 12 elements.

Finally we evaluate the objective function $f_0(\mathbf{s})$ [see (3.49)] at all the critical points in $\bar{\Xi}_1$ and select the one with the smallest objective value as the global optimal solution of Π_8, for the case $\kappa_d = 0, \kappa_\theta = 1, \lambda_1 > \lambda_2$, and $\rho \leq \|\mathbf{c}\| - r$.

Distance-only observation

When the sensor can only measure its distance to the target, i.e., $\kappa_d = 1, \kappa_\theta = 0$, it can be shown (see Appendix A.5) that the coefficients appearing in $f_1(x, y)$ [see (3.59)] are:

$$\beta_3 < 0, \quad \alpha_8 = -c_1\beta_3, \quad \alpha_7 = -c_2\beta_3,$$
$$\beta_2 = \beta_1 = \alpha_6 = \alpha_5 = \alpha_4 = \alpha_3 = \alpha_2 = \alpha_1 = 0.$$

Therefore, (3.59) can be simplified into the following 8th-order bivariate polynomial:

$$0 = f_1(x, y) = \beta_3\Delta^2 xy(x^2 + y^2 - c_1x - c_2y). \tag{3.67}$$

Since $\Delta = x^2 + y^2 > 0$ and $\beta_3 < 0$, the roots of f_1 must satisfy *either* one of the following two polynomial equations:

$$0 = \xi_1(x, y) = x^2 + y^2 - c_1x - c_2y, \tag{3.68}$$
$$0 = \xi_2(x, y) = xy. \tag{3.69}$$

Thus, the set of all the critical points given a distance-only measurement is $\bar{\Xi}_{1l} \cup \bar{\Xi}_{1r}$, where $\bar{\Xi}_{1l} = \{(x, y) | \xi_1(x, y) = f_2(x, y) = 0\}$ and $\bar{\Xi}_{1r} = \{(x, y) | \xi_2(x, y) = f_2(x, y) = 0\}$. Note though that the set of possible global minima, $\bar{\Xi}_1$, contains only the critical points that belong to the arc ADB (see Lemma 13), and thus $\bar{\Xi}_1$ is a subset of $\bar{\Xi}_{1l} \cup \bar{\Xi}_{1r}$.

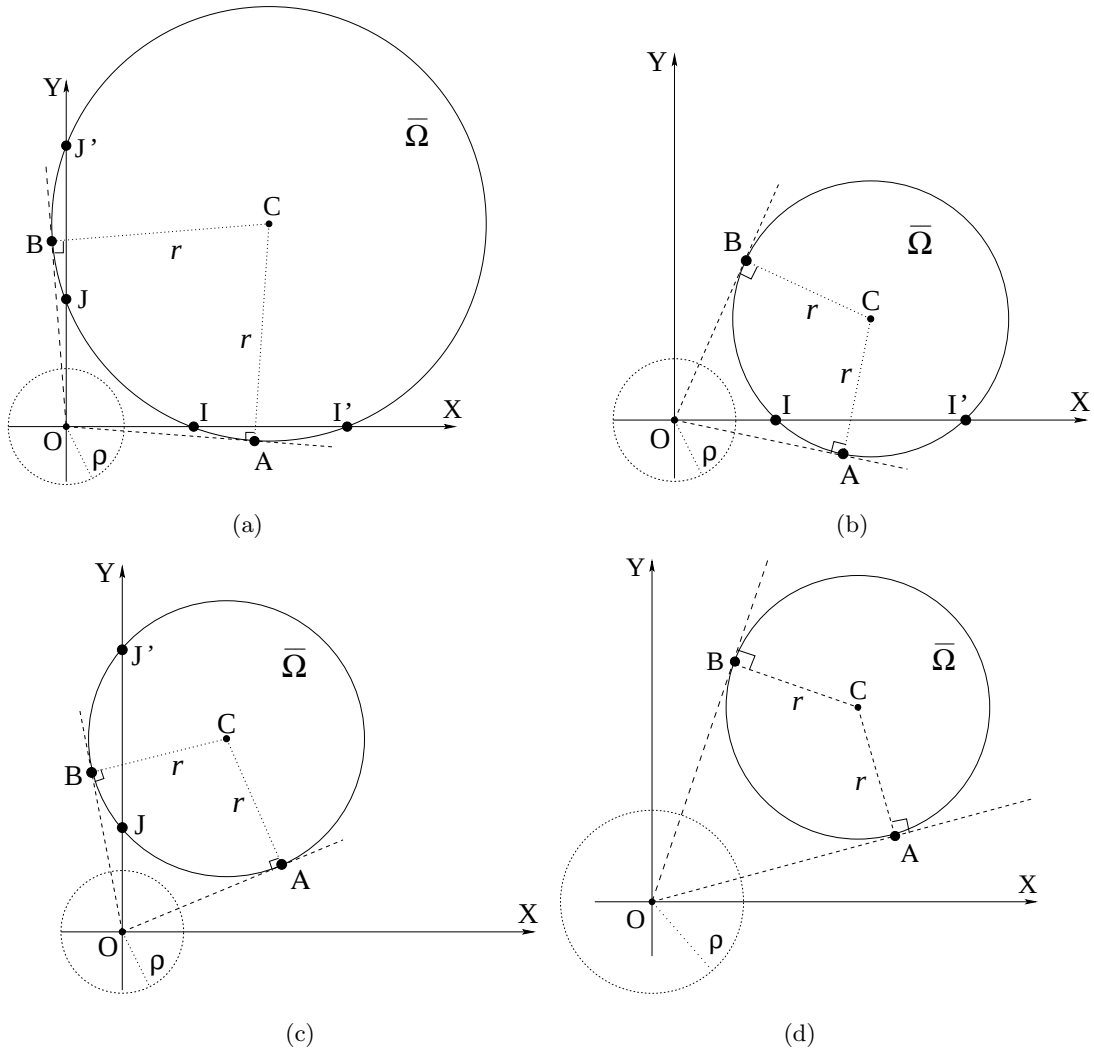


Figure 3.7: Critical points for single-sensor target tracking with distance-only observations. (a) $\max(|c_1|, |c_2|) \leq r$: There exist six critical points, A, B, I, I', J, J'. (b) $|c_2| \leq r \leq |c_1|$: The four critical points are A, B, I, I'. (c) $|c_1| \leq r \leq |c_2|$: The four critical points are A, B, J, J'. (d) $\min(|c_1|, |c_2|) \geq r$: Only A and B are real critical points, and there exists no real solution satisfying $\xi_2(x, y) = 0$ and $f_2(x, y) = 0$ simultaneously.

In order to determine the elements of $\bar{\Xi}_{1l}$, we note that (geometrically) ξ_1 [see (3.68)] and f_2 [see (3.57)] describe two circles in the plane whose intersection points belong to $\bar{\Xi}_{1l}$. In Appendix A.6, it is shown that $\bar{\Xi}_{1l}$ contains exactly two real elements, which correspond to the two tangent points A and B, shown in Figure 3.6(a). The Cartesian coordinates of A and B are (see Appendix A.6):

$$\begin{bmatrix} x_A \\ y_A \end{bmatrix} = \tau \begin{bmatrix} \cos(\varphi_C - \omega) \\ \sin(\varphi_C - \omega) \end{bmatrix}, \quad \begin{bmatrix} x_B \\ y_B \end{bmatrix} = \tau \begin{bmatrix} \cos(\varphi_C + \omega) \\ \sin(\varphi_C + \omega) \end{bmatrix}, \tag{3.70}$$

where [see Figure 3.6(a)]

$$\tau = \sqrt{\|\mathbf{c}\|^2 - r^2},\ \varphi_C = \arctan\left(\frac{c_2}{c_1}\right),\ \omega = \arcsin\left(\frac{r}{\|\mathbf{c}\|}\right).$$

Next we focus on $\bar{\Xi}_{1r}$. It is straightforward to conclude from ξ_2 [see (3.69)] that *either* $x = 0$ *or* $y = 0$. Substituting $x = 0$ or $y = 0$ into $f_2 = 0$ [see (3.57)], we obtain the following four critical points [see Figure 3.7(a)]:

$$[x_I\ y_I]^{\mathrm{T}} = \left[\mathrm{sign}(c_1)\left(|c_1| - \sqrt{r^2 - c_2^2}\right) \quad 0\right]^{\mathrm{T}}, \text{if } |c_2| \le r,$$
$$[x_{I'}\ y_{I'}]^{\mathrm{T}} = \left[\mathrm{sign}(c_1)\left(|c_1| + \sqrt{r^2 - c_2^2}\right) \quad 0\right]^{\mathrm{T}}, \text{if } |c_2| \le r,$$
$$[x_J\ y_J]^{\mathrm{T}} = \left[0 \quad \mathrm{sign}(c_2)\left(|c_2| - \sqrt{r^2 - c_1^2}\right)\right]^{\mathrm{T}}, \text{if } |c_1| \le r,$$
$$[x_{J'}\ y_{J'}]^{\mathrm{T}} = \left[0 \quad \mathrm{sign}(c_2)\left(|c_2| + \sqrt{r^2 - c_1^2}\right)\right]^{\mathrm{T}}, \text{if } |c_1| \le r,$$

where $\mathrm{sign}(x)$ is the sign function of a real variable x.

Note that the number of the real solutions satisfying $\xi_2 = f_2 = 0$ depends on $|c_1|, |c_2|$, and r. Specifically, if $\max(|c_1|, |c_2|) \le r$ [see Figure 3.7(a)], there are four real solutions (I, I', J, J') in $\bar{\Xi}_{1r}$. If $|c_2| \le r \le |c_1|$ [see Figure 3.7(b)], $\bar{\Xi}_{1r}$ only consists of I and I'. Similarly, if $|c_1| \le r \le |c_2|$ [see Figure 3.7(c)], only J and J' are valid solutions in $\bar{\Xi}_{1r}$. Finally, when $\min(|c_1|, |c_2|) \ge r$ [see Figure 3.7(d)], $\bar{\Xi}_{1r}$ becomes an empty set, i.e., there exists no real solution that can fulfill $\xi_2 = 0$ and $f_2 = 0$ simultaneously.

In summary, $\bar{\Xi}_1$, containing all the critical points in the arc ADB, is a subset of $\bar{\Xi}_{1l} \cup \bar{\Xi}_{1r}$, which has at most six elements (A,B,I,I',J,J'). The final step is to evaluate the objective function $f_0(\mathbf{s})$ [see (3.49)] at all the critical points in $\bar{\Xi}_1$, and select the

one with the smallest objective value as the global optimal solution of Π_8, for the case $\kappa_d = 1, \kappa_\theta = 0, \lambda_1 > \lambda_2$, and $\rho \leq \|\mathbf{c}\| - r$.

$\mathbf{\Lambda}$ has two same eigenvalues: $\lambda_1 = \lambda_2 = \lambda$

In the previous sections, we have analyzed and presented the solutions for the three observation models under the assumption $\lambda_1 > \lambda_2$. We hereafter consider the special case $\lambda_1 = \lambda_2 = \lambda$, i.e., $\mathbf{\Lambda} = \lambda^{-1}\mathbf{I}_2$.

In Appendix A.7, we show that for single-sensor target tracking with bearing-only or distance-and-bearing observations, $f_1(x, y)$ [see (3.59)] can be transformed into a linear equation, $c_2 x - c_1 y = 0$, which depicts a line passing through the origin O and the center C [see Figure 3.6(a)]. Furthermore, the coordinates $\mathbf{s}_D$ and $\mathbf{s}_{D'}$ of the two critical points D and D' (obtained by the intersection of the circle described by $f_2(x, y) = 0$ [see (3.57)] with the line $f_1(x, y) = c_2 x - c_1 y = 0$), satisfy the relation $f_0(\mathbf{s}_D) \leq f_0(\mathbf{s}_{D'})$ (see Lemma 13). Therefore, for the bearing-only and distance-and-bearing observation models, the global optimal solution of Π_8 is $\mathbf{s}^* = \mathbf{s}_D = \frac{\mathbf{c}}{\|\mathbf{c}\|}\big(\|\mathbf{c}\| - r\big)$ [see Figure 3.6(a)], when $\lambda_1 = \lambda_2$.

On the other hand, as shown in Appendix A.7, the objective function $f_0(\mathbf{s})$ in (3.49) remains a constant and is independent of $\mathbf{s}$ for single-sensor target tracking with distance-only measurements. In other words, $\nabla f_0(\mathbf{s}) = \mathbf{0}_{2\times 1}$ when $\kappa_d = 1, \kappa_\theta = 0, \lambda_1 = \lambda_2$. Thus, the sensor can move anywhere within $\bar{\Omega}$. However, in order to increase the probability of target re-detection at the following time steps, we require the sensor to move to D, which is the closest to the target point of $\bar{\Omega}$.

In summary, if $\lambda_1 = \lambda_2$, the best sensing location, regardless of the employed observation model, is D [see Figure 3.6(a)], i.e., $\mathbf{s}^* = \mathbf{s}_D = \frac{\mathbf{c}}{\|\mathbf{c}\|}\big(\|\mathbf{c}\| - r\big)$.

3.5.2 Case II: $\sqrt{\|\mathbf{c}\|^2 - r^2} \leq \rho < \|\mathbf{c}\| + r$

As shown in Figure 3.6(b), and based on Lemma 13, the only *active* constraint for Case II is the collision-avoidance constraint (3.51), while the maximum-speed constraint (3.50)

is *inactive* and hence its corresponding Lagrange multiplier is $\mu^* = 0$. Thus, (3.52)–(3.54) are simplified into:

$$\nabla f_0(\mathbf{s}^*) - \nu^* \mathbf{s}^* = \mathbf{0}_{2\times 1}, \tag{3.71}$$

$$\|\mathbf{s}^*\|^2 - \rho^2 = 0. \tag{3.72}$$

Clearly, (3.72) is a 2nd-order polynomial equation in the variables x and y, i.e.,

$$0 = f_5(x, y) = x^2 + y^2 - \rho^2. \tag{3.73}$$

Applying the same technique as in Case I to eliminate ν^* from (3.71), yields:

$$(\mathbf{s}^*)^{\mathrm{T}} \mathbf{C} \left(\frac{\pi}{2}\right) \nabla f_0(\mathbf{s}^*) = 0. \tag{3.74}$$

Further analysis shows that, if (i) $\lambda_1 > \lambda_2$; *and* (ii) $\kappa_d \sigma_d^{-2} \rho^2 \neq \kappa_\theta \sigma_\theta^{-2}$ (which is automatically satisfied for the distance-only and bearing-only measurement models, and also holds true if $\rho \neq \frac{\sigma_d}{\sigma_\theta}$ for the distance-and-bearing observation model), then (3.74) is equivalent to the following 2nd-order bivariate polynomial f_4 (see Appendix A.8):

$$0 = f_4(x, y) = \ xy. \tag{3.75}$$

It is easy to verify that the four real solutions satisfying f_4 [see (3.75)] and f_5 [see (3.73)] are $\left\{[\pm\rho \ \ 0]^{\mathrm{T}},\ [0 \ \ \pm\rho]^{\mathrm{T}}\right\}$. However, *not all* these critical points belong to the feasible region $\bar{\Omega}$. In particular, $[-\mathrm{sign}(c_1)\rho \ \ \ 0]^{\mathrm{T}}$ and $[0 \ \ \ -\mathrm{sign}(c_2)\rho]^{\mathrm{T}}$ violate the maximum-speed constraint (3.50) (see Appendix A.9). The remaining two points $[\mathrm{sign}(c_1)\rho \ \ \ 0]^{\mathrm{T}}$ and $[0 \ \ \ \mathrm{sign}(c_2)\rho]^{\mathrm{T}}$ belong to $\bar{\Omega}$ [see Figure 3.6(b)], if the following conditions are satisfied (see Appendix A.9):

$$[\mathrm{sign}(c_1)\rho \ \ 0]^{\mathrm{T}} \in \bar{\Omega} \iff \left(\rho - |c_1|\right)^2 \leq r^2 - c_2^2, \tag{3.76}$$

$$[0 \ \ \mathrm{sign}(c_2)\rho]^{\mathrm{T}} \in \bar{\Omega} \iff \left(\rho - |c_2|\right)^2 \leq r^2 - c_1^2. \tag{3.77}$$

Hence, the set Ξ_2 containing all the *feasible* critical points has at most two elements. Specifically, if both (3.76) and (3.77) are satisfied, $\Xi_2 = \left\{[\mathrm{sign}(c_1)\rho \ \ 0]^{\mathrm{T}}, [0 \ \ \mathrm{sign}(c_2)\rho]^{\mathrm{T}}\right\}$; if only (3.76) is satisfied, $\Xi_2 = \left\{[\mathrm{sign}(c_1)\rho \ \ 0]^{\mathrm{T}}\right\}$; if only (3.77) is satisfied, $\Xi_2 = \left\{[0 \ \ \mathrm{sign}(c_2)\rho]^{\mathrm{T}}\right\}$; when neither (3.76) nor (3.77) is satisfied, $\Xi_2 = \varnothing$, which corresponds to the case shown in Figure 3.6(b).

Since the curve EGF is an arc of the circle defined by (3.72), it is also necessary to consider the objective value attained at the two boundary points E and F, or equivalently, the intersection points of the two circles: $\|\mathbf{s}-\mathbf{c}\| = r$ and $\|\mathbf{s}\| = \rho$ [see Figure 3.6(b)], whose Cartesian coordinates are (see Appendix A.10):

$$\begin{bmatrix} x_E \\ y_E \end{bmatrix} = \rho \begin{bmatrix} \cos(\varphi_C - \varpi) \\ \sin(\varphi_C - \varpi) \end{bmatrix}, \quad \begin{bmatrix} x_F \\ y_F \end{bmatrix} = \rho \begin{bmatrix} \cos(\varphi_C + \varpi) \\ \sin(\varphi_C + \varpi) \end{bmatrix}, \tag{3.78}$$

where [see Figure 3.6(b)]

$$\varphi_C = \arctan\left(\frac{c_2}{c_1}\right), \quad \varpi = \arccos\left(\frac{\rho^2 + \|\mathbf{c}\|^2 - r^2}{2\rho\|\mathbf{c}\|}\right).$$

Therefore, the set Ξ_2 is augmented into $\bar{\Xi}_2 = \Xi_2 \cup \{E, F\}$, which can have two, three, or at most four elements. The global optimal solution of Π_5 in Case II is selected as the $\mathbf{s}^* \in \bar{\Xi}_2$ with the smallest objective value $f_0(\mathbf{s}^*)$. Note that the sensor is not necessarily required to move at its maximum speed $v_{\max}$ in Case II.

Remark 15. *The preceding derivations follow the assumption that* (i) $\lambda_1 > \lambda_2$*; and* (ii) $\kappa_d \sigma_d^{-2} \rho^2 \neq \kappa_\theta \sigma_\theta^{-2}$*. In Appendix A.11, we also address the special cases where* (i) $\lambda_1 = \lambda_2$*; or* (ii) $\kappa_d = \kappa_\theta = 1$ *and* $\rho = \frac{\sigma_d}{\sigma_\theta}$*, and show that* $f_0(\mathbf{s})$ *remains constant along the curve* EGF *[see Figure* 3.6(b)*] if either one of these two conditions is satisfied. This means that any point belonging to the curve* EGF *is a global optimal solution. In such cases, we require the sensor to move to the location* G *[see Figure* 3.6(b)*], which is the closest point of the arc* EGF *to* C*, i.e.,* $\mathbf{s}^* = \mathbf{s}_G = \frac{\mathbf{c}}{\|\mathbf{c}\|}\rho$*.*

3.5.3 Case III: $\|\mathbf{c}\| - r < \rho < \sqrt{\|\mathbf{c}\|^2 - r^2}$

As shown in Figure 3.6(c), and based on Lemma 13, the optimal solution $\mathbf{s}^* \in \bar{\Omega}$ must reside on the curve $AEGFB$, which is composed of three segments, i.e., $\Theta = \Theta_1 \cup \Theta_2 \cup \Theta_3$. Θ_1 and Θ_2 are due to the maximum-speed constraint (3.50), and Θ_3 is due to the collision-avoidance constraint (3.51).

To obtain the critical points for Case III, we proceed as follows: We first ignore the collision-avoidance constraint (3.51), and calculate all critical points of Π_8 under the maximum-speed constraint (3.50) following the same process as for Case I (see Section 3.5.1). Note, however, that we only need to consider those critical points that

reside in Θ_1 and Θ_2, which is a *subset* Ξ_3 of $\bar{\Xi}_1$. Then, we ignore the maximum-speed constraint (3.50) and apply the same method as for Case II (see Section 3.5.2) to compute the optimal solution $\mathbf{s}^\dagger$ of Π_8 over the set Θ_3. Following the above strategy, the set $\bar{\Xi}_3$ of all the critical points for Case III is $\bar{\Xi}_3 = \Xi_3 \cup \{\mathbf{s}^\dagger\}$.

The final step is to evaluate the objective function $f_0(\mathbf{s})$ at all the critical points in $\bar{\Xi}_3$, and select the one with the smallest objective value as the global optimal solution of Π_8.

3.5.4 Case IV: $\|\mathbf{c}\| + r \leq \rho$

From the geometry of Figure 3.6(d), we immediately conclude that there exists no real solution that satisfies both (3.50) and (3.51) simultaneously, i.e., the feasible set $\bar{\Omega}$ for Π_8 is empty. In this case, regardless of the measurement model, we require the sensor to move to D', as shown in Figure 3.6(d), which ensures that (i) the sensor maintains the largest possible distance from the target so as to avoid collision, and (ii) it satisfies the maximum-speed constraint (3.50). Thus, the solution of Π_8 in Case IV is [see Figure 3.6(d)]:

$$\mathbf{s}^* = \mathbf{s}_{D'} = \frac{\mathbf{c}}{\|\mathbf{c}\|}\big(r + \|\mathbf{c}\|\big).$$

3.5.5 Summary: single-sensor solution algorithm

For completeness, we outline the algorithmic flow chart for computing the global optimum of the single-sensor optimization problem Π_8 in Algorithm 1.

3.5.6 Extension to obstacle avoidance

Our approach to determine the global optimal solution for single-sensor target tracking, as described above, can be readily extended to include more complicated motion constraints, such as limitations on the sensor's kinematics and constraints imposed by obstacles. To proceed, we can employ one or multiple polynomials to describe (exactly or approximately) the obstacles' boundaries,[8] or simply seek the minimal circle that encloses the obstacles. From Lemma 13, the global optimal solution must be on the

[8] Note that kinematic constraints can also be described as obstacles in the sensor's vicinity limiting its motion range.

Algorithm 1 Single-sensor optimization

Require: $\lambda_1, \lambda_2, \mathbf{c}, r, \rho, \sigma_d, \sigma_\theta, \kappa_d, \kappa_\theta$

Ensure: $\mathbf{s}$ {Minimize (3.49), while satisfying (3.50)-(3.51)}

if $\rho \leq \|\mathbf{c}\| - r$ **then**
  if $\lambda_1 \neq \lambda_2$ **then**
    if $\kappa_d = \kappa_\theta = 1$ **then**
      Compute $\mathbf{s}$ from (3.63) and (3.57) {See Section 3.5.1}
    else if $\kappa_d = 0,\ \kappa_\theta = 1$ **then**
      Compute $\mathbf{s}$ from (3.66) and (3.57) {See Section 3.5.1}
    else
      Compute $\mathbf{s}$ from (3.68)-(3.69), and (3.57) {See Section 3.5.1}
    end if
  else
    Compute $\mathbf{s} = \frac{\mathbf{c}}{\|\mathbf{c}\|}(\|\mathbf{c}\| - r)$ {See Section 3.5.1}
  end if
else if $\sqrt{\|\mathbf{c}\|^2 - r^2} \leq \rho < \|\mathbf{c}\| + r$ **then**
  Compute $\mathbf{s}$ from (3.76)-(3.78) {See Section 3.5.2}
else if $\|\mathbf{c}\| - r < \rho < \sqrt{\|\mathbf{c}\|^2 - r^2}$ **then**
  Determine $\mathbf{s}$ following the strategy outlined in Section 3.5.3
else
  Compute $\mathbf{s} = \frac{\mathbf{c}}{\|\mathbf{c}\|}(r + \|\mathbf{c}\|)$ {See Section 3.5.4}
end if

boundary of the feasible set. In other words, if the obstacle-avoidance constraint is *active* and its associated Lagrange multiplier is *nonzero*, the global optimal solution must satisfy the polynomial equation describing the boundary of the obstacles, denoted as $c(\mathbf{s}^*) = 0$.[9] Thus, the corresponding KKT necessary condition, similar to (3.55), has the form:[10]

$$\nabla f_0(\mathbf{s}^*) + \upsilon^* \nabla c(\mathbf{s}^*) = \mathbf{0}_{2\times 1}, \tag{3.79}$$

where υ^* is the Lagrange multiplier. Moreover, since $c(\mathbf{s}^*)$ is a polynomial, $\nabla c(\mathbf{s}^*)$ is a 2×1 vector whose components are also polynomials in $\mathbf{s}^*$. To eliminate υ^*, we multiply both sides of (3.79) by $\left(\nabla c(\mathbf{s}^*)\right)^{\mathrm{T}} \mathbf{C}(\frac{\pi}{2})$, which yields:

$$\left(\nabla c(\mathbf{s}^*)\right)^{\mathrm{T}} \mathbf{C}\left(\frac{\pi}{2}\right) \nabla f_0(\mathbf{s}^*) = 0. \tag{3.80}$$

Note that the only difference between (3.80) and (3.58) is that it contains the term $\nabla c(\mathbf{s}^*)$ instead of $\mathbf{s}^* - \mathbf{c}$. Therefore, we can apply the same process described in Section 3.5.1 to transform (3.80) into a polynomial equation $f(x, y) = 0$, and solve the corresponding polynomial system $f(x, y) = c(x, y) = 0$ by employing the Sylvester resultant and the companion matrix. In fact, our approach can be generalized to solve any optimization problem with two optimization variables (i.e., 2-D sensor motion), while only requiring that the objective function and all constraints are expressed as rational functions with respect to the two variables.

3.6 Multiple-sensor active target tracking

Motivated by the simplicity of the analytic-form solution for the single-sensor optimal target tracking (see Section 3.5), a straightforward approach to solve the optimization problem Π_4 is to iteratively minimize its objective function [see (3.33)] for each optimization variable separately. Specifically, the solution of Π_4 is acquired by employing

[9] Since there exists a linear relation between $\mathbf{s}$ and $\mathbf{p}$ (see Lemma 7), any polynomial $h(\mathbf{p})$, expressed in $\mathbf{p}$, preserves its polynomial property under linear transformation, i.e., $h(\mathbf{p}) = h\left(\mathbf{C}(\varphi_0)\,\mathbf{s}\right) = c(\mathbf{s})$, and $c(\mathbf{s})$ is a polynomial with respect to $\mathbf{s}$.

[10] Note that in (3.79) we only consider one constraint $c(\mathbf{s}^*) = c(x, y) = 0$ as being active. In case of two (or more) *active* constraints $c_i(x, y)$ and $c_j(x, y)$, the solutions that simultaneously satisfy $c_i(x, y) = c_j(x, y) = 0$ are generally discrete and finite. Thus, the optimal solution can be easily obtained by evaluating $f_0(\mathbf{s})$ at each solution and selecting the one with the smallest objective value.

the cyclic coordinate descent method, also referred to as nonlinear Gauss-Seidel algorithm [91, Ch. 3], which requires to solve the following optimization problem at each step:

- Optimization Problem 9 (Π_9)

$$\underset{\mathbf{s}_i^{(\ell+1)}}{\text{minimize}}\ \text{tr}\left(\left(\mathbf{P}_i^{(\ell+1)}\right)^{-1}+\frac{\kappa_{d_i}}{\sigma_{d_i}^2}\frac{\left(\mathbf{s}_i^{(\ell+1)}\right)\left(\mathbf{s}_i^{(\ell+1)}\right)^{\mathrm{T}}}{\left(\mathbf{s}_i^{(\ell+1)}\right)^{\mathrm{T}}\left(\mathbf{s}_i^{(\ell+1)}\right)}+\frac{\kappa_{\theta_i}}{\sigma_{\theta_i}^2}\frac{\mathbf{J}\left(\mathbf{s}_i^{(\ell+1)}\right)\left(\mathbf{s}_i^{(\ell+1)}\right)^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{\left(\left(\mathbf{s}_i^{(\ell+1)}\right)^{\mathrm{T}}\left(\mathbf{s}_i^{(\ell+1)}\right)\right)^2}\right)^{-1} \tag{3.81}$$

$$\text{subject to}\ \left\|\mathbf{s}_i^{(\ell+1)}-\mathbf{c}_i\right\|\leq r_i\quad\text{and}\quad\left\|\mathbf{s}_i^{(\ell+1)}\right\|\geq\rho_i$$

where $\mathbf{s}_i^{(\ell+1)}$ is the sought new optimal value of $\mathbf{s}_i$ at iteration $\ell+1$, $\mathbf{P}_i^{(\ell+1)}$ is defined in (3.82), and $\mathbf{s}_j^{(\ell+1)}, j=1,\ldots,i-1$, and $\mathbf{s}_j^{(\ell)}, j=i+1,\ldots,M$, are the remaining optimization variables, considered fixed during this step, computed sequentially during the previous iterations. Note that the matrix $\mathbf{P}_i^{(\ell+1)}$ is positive definite, and in general, non-diagonal. However, based on Lemma 7, through a similarity transformation, the optimization algorithm employed for a single sensor can be readily applied to solve Π_9.

$$\begin{aligned}\left(\mathbf{P}_i^{(\ell+1)}\right)^{-1}=&\boldsymbol{\Lambda}+\sum_{j=i+1}^{M}\left(\frac{\kappa_{d_j}}{\sigma_{d_j}^2}\frac{\left(\mathbf{s}_j^{(\ell)}\right)\left(\mathbf{s}_j^{(\ell)}\right)^{\mathrm{T}}}{\left(\mathbf{s}_j^{(\ell)}\right)^{\mathrm{T}}\left(\mathbf{s}_j^{(\ell)}\right)}+\frac{\kappa_{\theta_j}}{\sigma_{\theta_j}^2}\frac{\mathbf{J}\left(\mathbf{s}_j^{(\ell)}\right)\left(\mathbf{s}_j^{(\ell)}\right)^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{\left(\left(\mathbf{s}_j^{(\ell)}\right)^{\mathrm{T}}\left(\mathbf{s}_j^{(\ell)}\right)\right)^2}\right)\\&+\sum_{j=1}^{i-1}\left(\frac{\kappa_{d_j}}{\sigma_{d_j}^2}\frac{\left(\mathbf{s}_j^{(\ell+1)}\right)\left(\mathbf{s}_j^{(\ell+1)}\right)^{\mathrm{T}}}{\left(\mathbf{s}_j^{(\ell+1)}\right)^{\mathrm{T}}\left(\mathbf{s}_j^{(\ell+1)}\right)}+\frac{\kappa_{\theta_j}}{\sigma_{\theta_j}^2}\frac{\mathbf{J}\left(\mathbf{s}_j^{(\ell+1)}\right)\left(\mathbf{s}_j^{(\ell+1)}\right)^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{\left(\left(\mathbf{s}_j^{(\ell+1)}\right)^{\mathrm{T}}\left(\mathbf{s}_j^{(\ell+1)}\right)\right)^2}\right)\end{aligned} \tag{3.82}$$

The optimization process in the above Gauss-Seidel relaxation (GSR) algorithm (*sequentially* optimizing over $\mathbf{s}_1,\mathbf{s}_2,\ldots,\mathbf{s}_M$) is repeated until the maximum allowed number of iterations is reached (here set to 4), or the change in the objective function [see (3.33)] is less than 1%, whichever occurs first. Note that since the optimization process in the GSR algorithm is carried out sequentially for each variable $\mathbf{s}_i$, its computational complexity is only *linear* in the number of sensors, i.e., $\mathcal{O}(M)$. Furthermore, it is easily implemented, has low memory requirements and, as demonstrated in Section 3.7, it achieves the same level of tracking accuracy as the exhaustive search approach.

Finally, the algorithmic flow chart for the multi-sensor one-step-ahead single-target active tracking using GSR is outlined in Algorithm 2.

Algorithm 2 Multi-sensor one-step-ahead single-target active tracking

Require: $\mathbf{s}_i^{(0)} = \mathbf{c}_i,\ i = 1, \ldots, M$

Ensure: $\mathbf{s}_i = \mathbf{s}_i^{(\ell+1)},\ i = 1, \ldots, M$ {Minimize (3.33)}

1: **repeat**
2: **for** $i = 1$ to M **do**
3: Calculate $\left(\mathbf{P}_i^{(\ell+1)}\right)^{-1}$ from (3.82)
4: Determine $\mathbf{s}_i^{(\ell+1)}$ from (3.81) {See Algorithm 1}
5: $\mathbf{s}_i^{(\ell)} \Leftarrow \mathbf{s}_i^{(\ell+1)}$ {Update $\mathbf{s}_i$}
6: **end for**
7: **until** max. number of iterations is reached or change in the objective function is less than 1%

3.7 Simulation results

In order to evaluate the presented *constrained* optimal motion strategy, Gauss-Seidel Relaxation (GSR), we have conducted extensive simulation experiments and compared the performance of GSR to the following methods:

- *Grid-Based Exhaustive Search* (GBES). In this case, we discretize the feasible set of all sensors and perform an exhaustive search over all possible combinations of these to find the one that minimizes the trace of the posterior covariance matrix for the target's position estimates [see (3.31)]. Ideally, the GBES should return the global optimal solution and it could be used as a benchmark for evaluating the GSR, if the grid size is sufficiently small. However, this is difficult to guarantee in practice since its computational complexity is exponential in the number of sensors. Hence implementing the GBES becomes prohibitive when the number of sensors, M, increases and/or when the size of the grid cells decreases. Throughout the simulations, we discretize the curve Θ [see Figures 3.6(a)–3.6(c)] for each sensor-i ($i = 1, \ldots, M$) into 24 cells (arcs) of equal length.

- *Gradient Descent with Constant Step Size* (GDC). In order to compare GSR

with the methods proposed in [55] and [54], we implemented the steepest-descent algorithm [64, Ch. 1] with the same step size $\alpha = 50$ as in [55]. However, both [55] and [54] do not address the sensors' motion constraints. Therefore, to account for mobility constraints, we project each solution $\mathbf{s}_i^*$ generated by GDC back into the sensor-i's feasible region $\bar{\Omega}_i$, if $\mathbf{s}_i^* \notin \bar{\Omega}_i$ $(i = 1, \ldots, M)$.

• *Random Motion* (RM). This is a modification of an intuitive strategy that would require the sensors to move towards the target. In this case, however, and in order to ensure that the sensors do not converge to the same point, we require that at every time step sensor-i $(i = 1, \ldots, M)$ selects its heading direction with uniform probability towards points within the curve Θ [see Figures 3.6(a)–3.6(c)].

3.7.1 Simulation setup

For the purposes of this simulation, we adopt a zero-acceleration target motion model:

$$\dot{\mathbf{x}}_T(t) = \mathbf{F}\ \mathbf{x}_T(t) + \mathbf{G}\ \mathbf{w}(t), \tag{3.83}$$

where

$$\mathbf{F} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{G} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{x}_T(t) = \begin{bmatrix} x_T(t) \\ y_T(t) \\ \dot{x}_T(t) \\ \dot{y}_T(t) \end{bmatrix},$$

and $\mathbf{w}(t) = [w_x(t)\ w_y(t)]^{\mathrm{T}}$ is a zero-mean white Gaussian noise vector with covariance $\mathbb{E}\left[\mathbf{w}(t)\mathbf{w}^{\mathrm{T}}(t')\right] = q\mathbf{I}_2\delta(t - t')$, $q = 1$, and $\delta(t - t')$ is the Dirac delta. In our implementation, we discretize the continuous-time system model [see (3.83)] with time step $\delta t = 0.1$ sec.

The initial true state of the target is $\mathbf{x}_T(0) = [0\ 0\ -8\ 6]^{\mathrm{T}}$. The initial estimate for the target's state is $\hat{\mathbf{x}}_T(0|0) = [2\ -2\ 0\ 0]^{\mathrm{T}}$. This can be obtained by processing the first measurements from the sensors at time-step 0. At the beginning of the experiment, the sensors are randomly distributed within a circle of radius 5 m, which is at a distance of about 20 m from the target's initial position. The maximum speed for each sensor is set to 12 m/sec, i.e., the largest distance that a sensor can travel during any time step is 1.2 m. The minimum distance between the target and sensors is set to $\rho = 2$ m. The

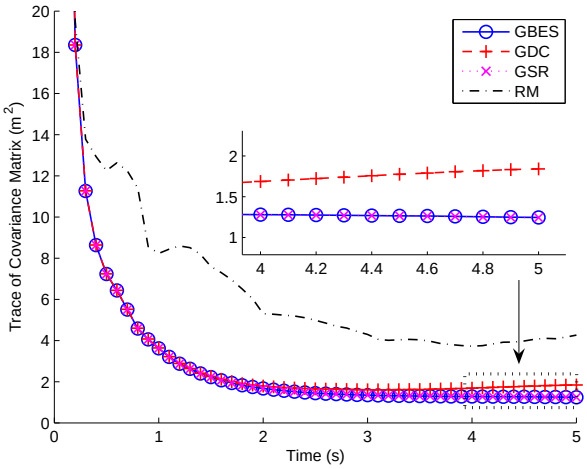


Figure 3.8: [Two-sensor case] Trace of the target's position posterior covariance matrix. Comparison between GBES, GDC, GSR, and RM.

duration of the simulations is 5 sec (i.e., 50 time steps). At every time step, we employ the methods described (i.e., GBES, GDC, GSR, and RM) to calculate the next sensing location of each sensor.

3.7.2 Target tracking with 2 sensors (homogeneous team)

We first investigate the scenario where 2 identical sensors track a moving target with distance-and-bearing observations (i.e., $\kappa_{d_i} = \kappa_{\theta_i} = 1,\ i = 1, 2$). The noise variances of the measurements are $\mathbf{R}_i = \text{diag}(\sigma_{d_i}^2, \sigma_{\theta_i}^2)$ with $\sigma_{d_i}^2 = 4$ m^2, and $\sigma_{\theta_i}^2 = 0.5$ rad^2, $i = 1, 2$.

The time evolution of the trace of the target's position covariance in a typical simulation is shown in Figure 3.8. As expected, the performance of GSR and GBES is improved compared to the case of GDC, and is significantly better than that of the non-optimized case RM. Additionally, the uncertainty in the target's position estimates (trace of the covariance matrix) achieved by the proposed GSR motion strategy is indistinguishable of that of the GBES, at a cost linear, instead of exponential, in the number of sensors. These results are typical for all experiments conducted and are summarized, for 50 trials, in Figure 3.9.

Figures 3.10(a)–3.10(d) depict the actual and estimated trajectories of the target,

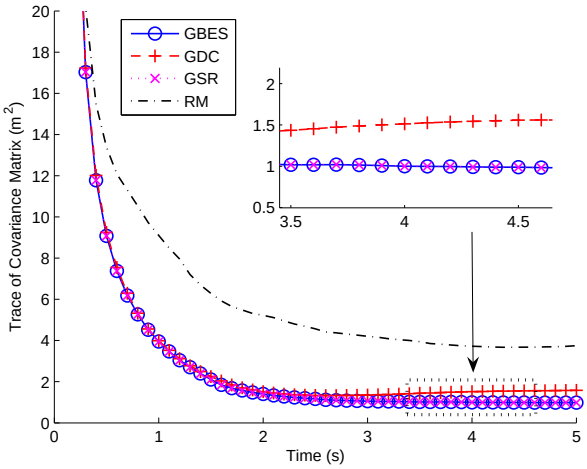


Figure 3.9: [Two-sensor case, Monte Carlo simulations] Average trace of the target's position posterior covariance matrix in 50 experiments. Comparison between GBES, GDC, GSR, and RM.

along with the trajectories of the two sensors, when employing as motion strategy GBES, GDC, GSR, and RM, respectively. As evident, the accuracy of the target's position estimates for GSR is better than the case of GDC or RM, and almost identical to that of GBES. Additionally, the EKF produces consistent estimates for GSR, in other words, the real target's position is within the 3σ ellipse centered at the target's estimated position.

Finally, we plot the 2-norm of the estimation error between the true target position and its posterior estimate in Fig. 3.11, when employing as motion strategy GBES, GDC, GSR, and RM, respectively. As evident, the estimates produced by RM have the largest error. Note that the other three methods generate comparable estimation performance through most time steps, while GSR slightly outperforms GDC between the time interval 2 to 3 sec (i.e., time steps 20 to 30).

3.7.3 Target tracking with 3 sensors (heterogeneous team)

We hereafter examine the performance of the GSR motion strategy for a heterogeneous team of 3 sensors tracking a moving target with a mixture of relative observations. In

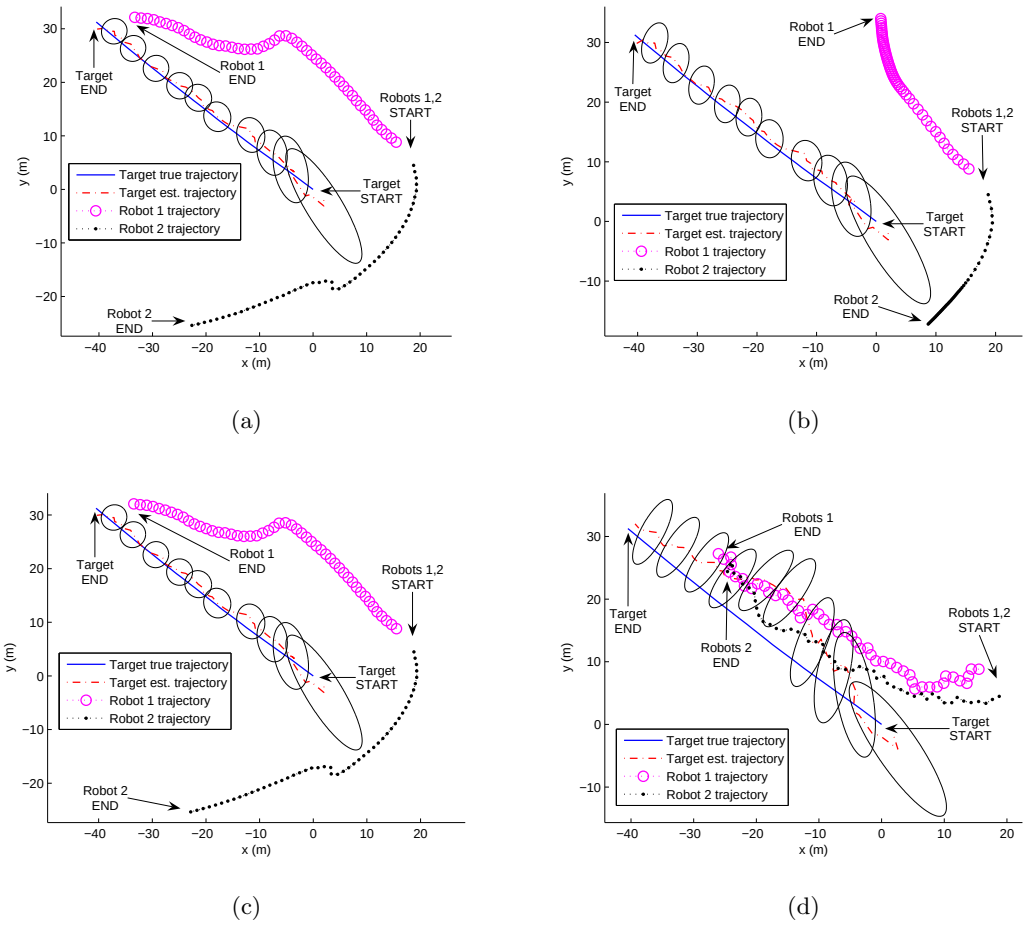


Figure 3.10: [Two-sensor case] Trajectories of the two sensors, and the actual and estimated trajectories of the target, when employing as motion strategy (a) GBES, (b) GDC, (c) GSR, and (d) RM. The ellipses denote the 3σ bounds for the target's position uncertainty at the corresponding time steps.

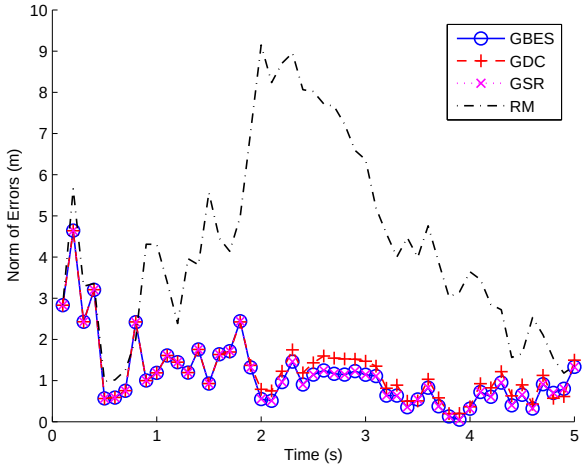


Figure 3.11: [Two-sensor case] 2-norm of the actual error between the target's position estimate and its true value. Comparison between GBES, GDC, GSR, and RM.

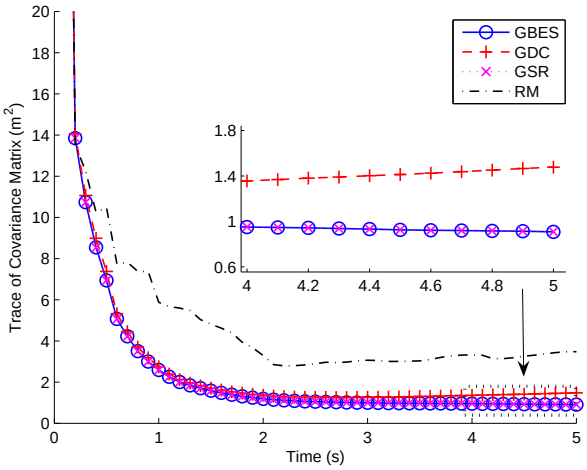


Figure 3.12: [Three-sensor case] Trace of the target's position posterior covariance matrix. Comparison between GBES, GDC, GSR, and RM.

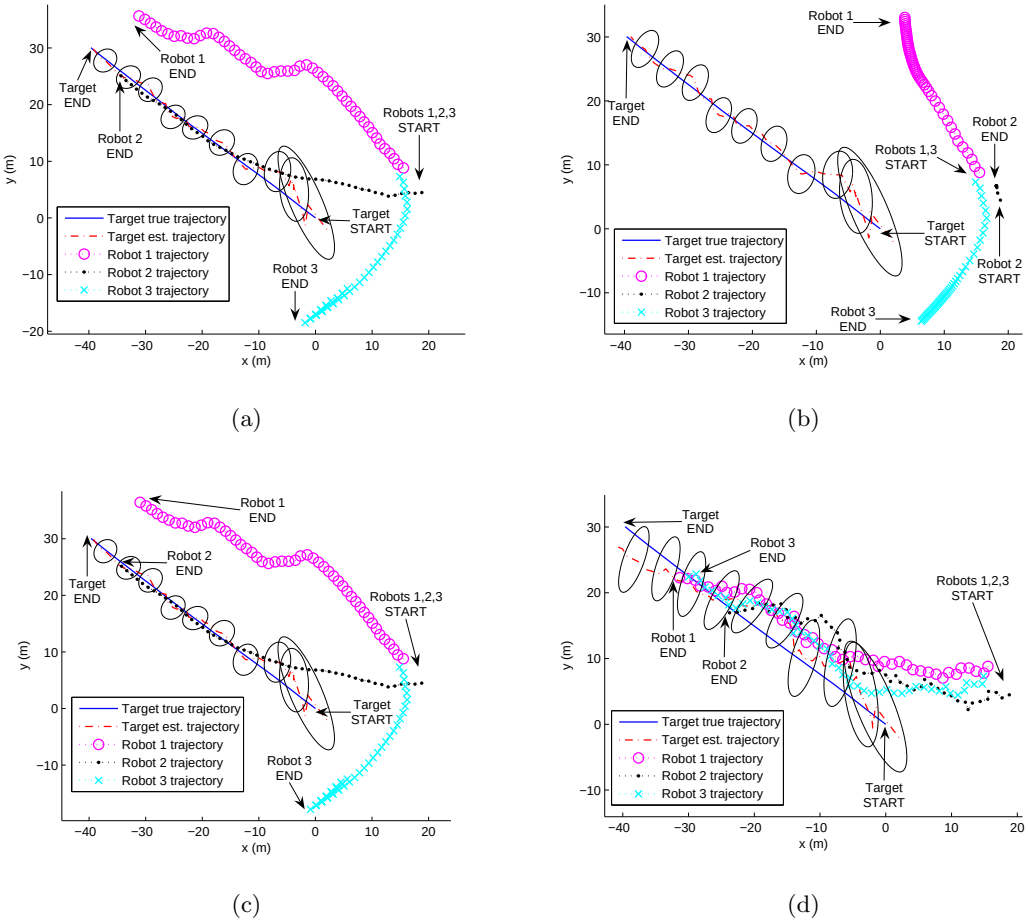


Figure 3.13: [Three-sensor case] Trajectories of the three sensors, and the actual and estimated trajectories of the target, when employing as motion strategy (a) GBES, (b) GDC, (c) GSR, and (d) RM. The ellipses denote the 3σ bounds for the target's position uncertainty at the corresponding time steps.

this case, sensor-1 can measure both distance and bearing to the target ($\kappa_{d_1} = \kappa_{\theta_1} = 1$), and its measurement noise covariance is set to $\mathbf{R}_1 = \text{diag}(\sigma_{d_1}^2, \sigma_{\theta_1}^2)$ with $\sigma_{d_1}^2 = 4$ m^2 and $\sigma_{\theta_1}^2 = 0.5$ rad^2. On the other hand, sensor-2 can only record bearing observations ($\kappa_{d_2} = 0, \kappa_{\theta_2} = 1$) with measurement noise variance $\sigma_{\theta_2}^2 = \sigma_{\theta_1}^2/2 = 0.25$ rad^2, while sensor-3 only has access to relative distance measurements ($\kappa_{d_3} = 1, \kappa_{\theta_3} = 0$) with noise variance $\sigma_{d_3}^2 = \sigma_{d_1}^2/2 = 2$ m^2.

Figures 3.13(a)–3.13(d) depict the actual and estimated trajectories of the target, along with the trajectories of the three sensors, when employing as motion strategy GBES, GDC, GSR, and RM, respectively. As evident, the accuracy of the target's position estimates for GSR is better than that of GDC or RM, and almost identical to that of GBES. Furthermore, the EKF estimates for the sensors that employ the GSR motion strategy are consistent.

Interestingly, in this case for both the GBES and GSR motion strategies, sensor-2, which only measures relative bearing, immediately starts following the target, and attempts to minimize its distance to it. The reason for this is the following: As shown in Lemma 13, although the information contributed by a distance measurement (i.e., the term $\frac{1}{\sigma_d^2}\frac{\mathbf{s}\mathbf{s}^\mathrm{T}}{\mathbf{s}^\mathrm{T}\mathbf{s}}$ in the proof of Lemma 13) is independent of the relative distance $\|\mathbf{s}\|$ between the target and the sensor, the information from a bearing measurement (i.e., the term $\frac{1}{\sigma_\theta^2}\frac{\mathbf{J}\mathbf{s}\mathbf{s}^\mathrm{T}\mathbf{J}^\mathrm{T}}{(\mathbf{s}^\mathrm{T}\mathbf{s})^2}$ in the proof of Lemma 13) increases as the relative distance, $\|\mathbf{s}\|$, decreases. Therefore this prompts sensor-2 to approach the target as close as possible.

Finally, we note that the time evolution of the trace of the target's position covariance matrix is similar to that of the two-sensor case, and is illustrated in Figure 3.12. Furthermore, the 2-norm of the estimation error is depicted in Figure 3.14.

3.7.4 Scalability and run-time

Contrary to the GBES method, which has computational and memory requirements exponential in the number of sensors, the complexity of the GSR algorithms is only linear. In order to corroborate our theoretical analysis, we have evaluated the computation time required by the four algorithms (GBES, GDC, GSR, and RM) for the case of a homogeneous sensor team ($\sigma_{d_i}^2 = 4$ m^2, and $\sigma_{\theta_i}^2 = 0.5$ rad^2, $i = 1, \ldots, M$) tracking a moving target. Specifically, we have examined the scalability of our algorithms by varying M from 2 to 100. These results are summarized in Table 3.1. In contrast, due

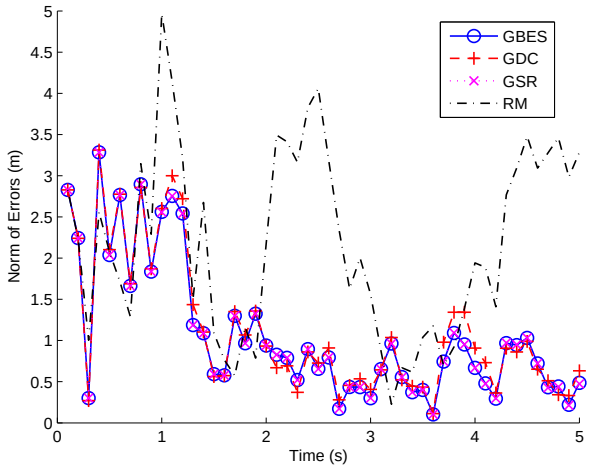


Figure 3.14: [Three-sensor case] 2-norm of the actual error between the target's position estimate and its true value. Comparison between GBES, GDC, GSR, and RM.

to its exponential computational complexity, we are only able to apply GBES to teams of up to 3 sensors.

Additionally, we plot the computational time with respect to the number of sensors in Fig. 3.15, when employing GSR as motion strategy. The plot clearly validates the claim that the GSR algorithm has linear, in the number of sensors, computational complexity. Finally, we should note that the main reason for the slower performance of the GSR algorithm (when compared to the GDC) is that we directly employ the MATLAB built-in function to compute the eigenvalues associated with the companion matrices, which improves the numerical accuracy at the expense of additional preprocessing steps. One of our future research directions is to improve the GSR performance (in terms of CPU running time) to be comparable to that of the GDC.

3.8 Experimental results

We hereafter describe one of the experiments performed to validate the performance of our proposed GSR algorithm. Our experimental setup is shown in Figure 3.16, where a team of three Pioneer II robots are deployed in a rectangular region of size approximately 4 m $\times$ 3 m. In Figure 3.16, the target is shown at the bottom right, while the other

Table 3.1: Computational time (sec)

M	GBES	GDC	GSR	RM
2	0.1539	0.0002489	0.0011	0.00007053
3	8.9945	0.0002947	0.0014	0.00007106
10	N/A	0.0008	0.0047	0.0001263
20	N/A	0.0018	0.0109	0.0004553
30	N/A	0.0024	0.0153	0.0004853
40	N/A	0.0027	0.0185	0.0003653
50	N/A	0.0033	0.0227	0.0004121
60	N/A	0.0040	0.0274	0.0005056
70	N/A	0.0045	0.0315	0.0005477
80	N/A	0.0052	0.0362	0.0006736
90	N/A	0.0059	0.0406	0.0007331
100	N/A	0.0066	0.0450	0.0008845

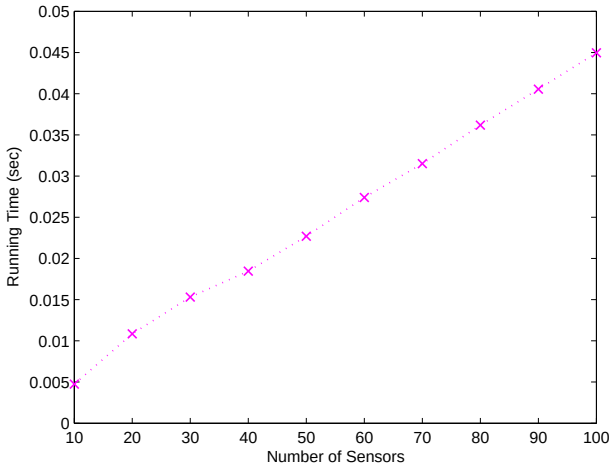


Figure 3.15: CPU time vs. number of sensors, when employing GSR as motion strategy.

two Pioneers are acting as tracking sensors. An overhead camera is employed to provide ground truth for evaluating the estimator's performance. To do so, rectangular boards with specific patterns (see Figure 3.16) are mounted on top of the Pioneers, and the pose (position and orientation) of each Pioneer, with respect to a global frame of reference, is computed from the captured images.

In the experiment, we adopt a zero-acceleration target motion model, where the target moves with constant speed of approximately 0.1 m/sec. The process noise $\mathbf{w}(t)$ [see (3.83)] is assumed to be a zero-mean white Gaussian noise vector with covariance $\mathbb{E}\left[\mathbf{w}(t)\mathbf{w}^{\mathrm{T}}(t')\right] = 10^{-6}\mathbf{I}_2\delta(t-t')$. In our implementation, the sampling time is set to $\delta t = 0.5$ sec. The initial true state of the target, computed from the overhead camera, is $\mathbf{x}_T(0) = [0.23\ 0.16\ 0.05\ 0.01]^{\mathrm{T}}$, while the initial estimate for the target's state is set to $\hat{\mathbf{x}}_T(0|0) = [0\ 0\ 0\ 0]^{\mathrm{T}}$. At the beginning, the two sensors are deployed at $\mathbf{p}_{S_1}(0) = [0.20\ 1.69]^{\mathrm{T}}$ and $\mathbf{p}_{S_2}(0) = [2.34\ 0.17]^{\mathrm{T}}$, respectively. The maximum speed for each sensor is set to 0.12 m/sec, and the minimum distance between the target and sensors is $\rho = 1$ m. We consider the scenario where each sensor measures both relative distance and bearing to the target (i.e., $\kappa_{d_i} = \kappa_{\theta_i} = 1,\ i = 1, 2$). These relative measurements are generated synthetically by adding noise to the relative distance and bearing calculated from the Pioneers' pose estimates using the overhead camera. In this experiment, the standard deviations of the distance and bearing measurement noise are set to $\sigma_{d_i} = 0.05$ m and $\sigma_{\theta_i} = 0.05$ rad, $i = 1, 2$, respectively.

The duration of the experiment is 30 sec (i.e., 60 time steps). At every time step, we employ the GSR method to calculate the next best sensing location of each sensor.

Figure 3.17 depicts the time evolution of the trace of the target's position covariance, which shows that at steady state, the standard deviation of the estimation error along each direction is around 0.02 m. The real estimation error, computed as the 2-norm between the target's estimated and true position (obtained from the overhead camera), is shown in Figure 3.18. As evident, the estimation error, when employing the GSR-based motion strategy, is immediately reduced from 0.28 m to 0.04 m, and is less than 0.05 m for most of the remaining time steps.

Figure 3.19 depicts the actual and estimated trajectories of the target, along with the real trajectories of the two sensors, when employing the GSR-based motion strategy. Again, as was the case in the simulations, the EKF produces consistent estimates for

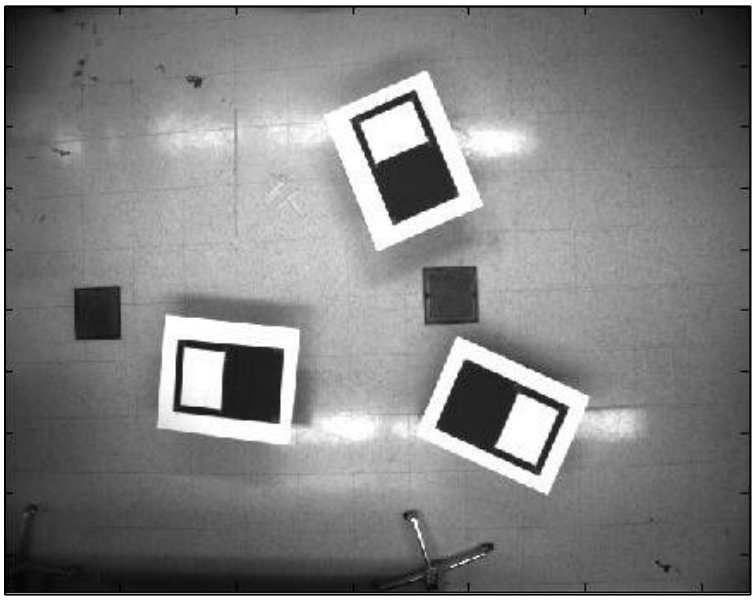


Figure 3.16: [Two-sensor case, experimental setup] Three Pioneer robots, each with a pattern board attached on its top. The target is located at the bottom right of the image, while the other two robots act as tracking sensors.

GSR, i.e., the real target's position is within the 3σ ellipse centered at the target's estimated position. This validates that our proposed GSR algorithm is robust and applicable to real systems.

3.9 Summary

In this chapter, we have provided analytical expressions for determining the next best sensing location (i.e., *one-step-ahead* optimization) of a *single* sensor, under mobility constraints, tracking a *single* moving target using distance-only, bearing-only, or distance and bearing observations.

We have also addressed the problem of generating optimal *one-step-ahead* motion strategies for *multiple heterogeneous* sensors, under mobility constraints, tracking a

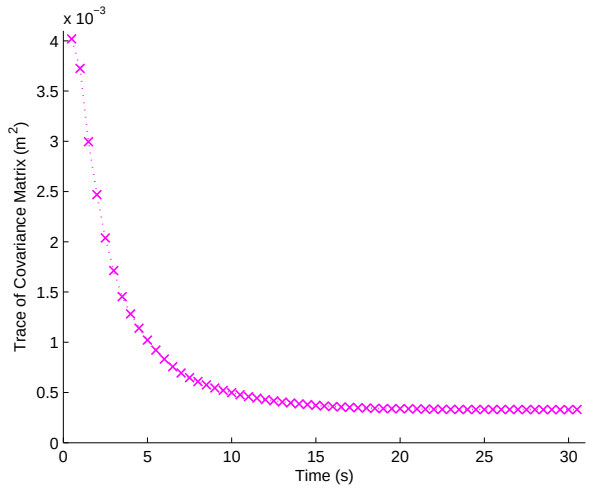


Figure 3.17: [Two-sensor case, experimental result] Trace of the target's position posterior covariance matrix, when employing GSR as motion strategy.

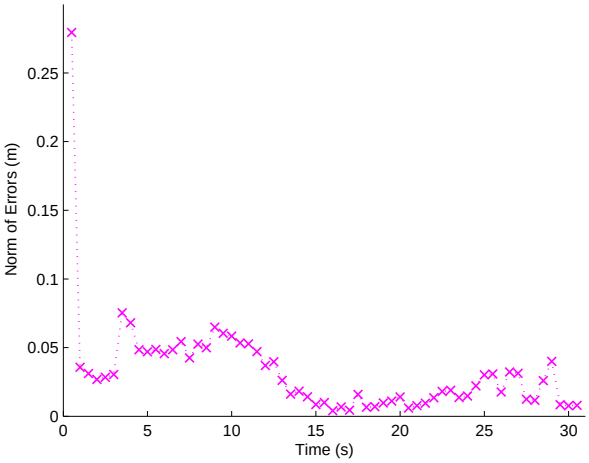


Figure 3.18: [Two-sensor case, experimental result] 2-norm of the error of the target's position posterior estimates, when employing GSR as motion strategy.

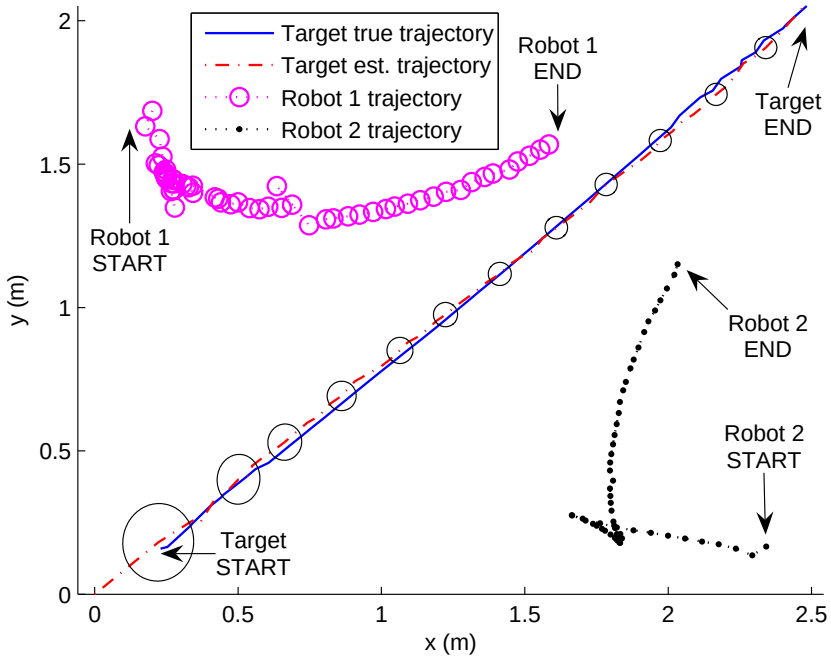


Figure 3.19: [Two-sensor case, experimental result] Real trajectories of the two sensors, and the actual and estimated trajectories of the target, when employing GSR as motion strategy. The ellipses denote the 3σ bounds for the target's position uncertainty at the corresponding time steps.

single moving target using *combinations* of relative observations (i.e., distance-only, bearing-only, or distance and bearing). In particular, we have shown that by imposing maximum-speed constraints on the sensors, the resulting non-convex constrained optimization problem of determining the next best sensing locations is, in general, NP-Hard. In order to provide an efficient, real-time solution for the multi-sensor case, we have relaxed the original NP-Hard problem and proposed an iterative algorithm, termed nonlinear Gauss-Seidel relaxation (GSR), that leverages the single-sensor analytical solution, and finds an approximate solution to the original problem with computational complexity only linear in the number of sensors. We have tested the GSR algorithm in Monte-Carlo simulations for target tracking using (i) homogeneous and (ii) heterogeneous teams of sensors, and shown that it achieves performance indistinguishable from that of the grid-based exhaustive search.

Chapter 4

Active Target Tracking: Multi-step-ahead Optimization

4.1 Introduction

In the previous chapter, we studied the one-step-ahead multi-sensor active target tracking problem, i.e., our objective was to determine the optimal sensing locations for tracking sensors at the next time step (also called myopic or short-term active target tracking) [92]. One-step-ahead motion strategies are preferred over multi-step-ahead strategies when the process-noise variance is large, since we are unable to reliably predict the target's motion for more than one time step [see Figure 4.1(a)]. On the other hand, if the process-noise variance is small, and thus we are able to predict the target's position over multiple time steps with relatively high accuracy [see Figure 4.1(b)], it is desirable to generate the optimal trajectories over multiple time steps (or long term) [93]. Long-term scheduling becomes particularly important when the sensor's coverage (e.g., sensing range, field of view) is limited [94]. For example, consider a robot tracking a target using a camera with limited field of view. In this scenario, it is necessary to make a long-term plan for the robot so as to ensure that the target is always visible.

At this point, it is important to note that applying the one-step-ahead optimal motion strategy over multiple time steps does not necessarily guarantee finding the global optimal solution at the end of a finite time horizon. Or equivalently, allowing sensors to follow the one-step-ahead motion strategy, the target's position uncertainty at the end

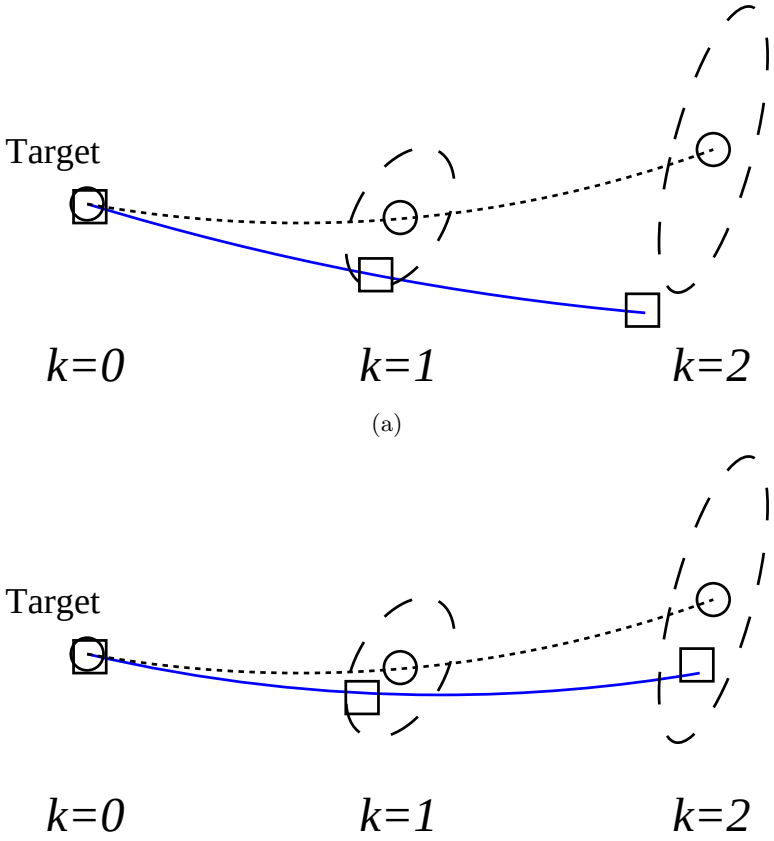


Figure 4.1: (a) The predicted target's positions over multiple time steps can significantly differ from its true positions if the target motion model is inaccurate. (b) Under small process noise variance, the predicted target's positions over a finite time horizon can be accurate enough to carry out the optimization over multiple time steps. In both plots, the target's true trajectory is shown by the blue solid line, while its predicted trajectory is denoted by the dotted black line. The target's true positions are shown as square boxes, while its predicted positions are denoted by circles, whose associated 3σ uncertainties are shown by the dashed line ellipses.

of the time interval being considered may not attain its global minimum. This is due to the inherent "near-sightedness" (hence, myopia) of the one-step-ahead motion strategy. More precisely, one-step-ahead optimization solves the multi-step-ahead problem in a greedy fashion, with respect to time, and in general greedy algorithms only guarantee convergence to a local, not global, optimum [64]. This motivates us to investigate alternative algorithms to generate the optimal trajectories for the multi-step-ahead active tracking problem.

In this chapter, we study the single-sensor multi-step-ahead active target tracking problem using distance-only observations. More specifically, assuming that the current time step is k_0, we seek to generate the optimal sensor's trajectory over the next K time steps, so as to minimize the trace of the predicted posterior covariance matrix for the target's position estimate at the $(k_0 + K)$-th time step. Similarly as in Chapter 3, we explicitly consider the sensor's mobility limitations imposed by the maximum-speed and collision-avoidance (i.e., minimum distance-to-target) constraints.

The main contributions of our work are twofold:

• In the problem formulation, we appropriately inflate the measurement-noise variances at future time steps to reflect the increasing uncertainty of the predicted target's position estimates. Additionally, we show that it is intractable to solve the KKT (necessary) optimality conditions so as to compute the (global) optimal solution for the multi-step-ahead active tracking problem.

• We develop an efficient algorithm for computing an approximate solution for the multi-step-ahead active tracking problem. In particular, we apply the nonlinear Gauss-Seidel relaxation to minimize the objective function along one block-coordinate (i.e., the sensing location at one future time step) per iteration. We show that the computational complexity of our proposed algorithm (termed as GSR) is quadratic in the number of time steps considered. We have tested GSR in Monte-Carlo simulations for target tracking under different process noise variances, and shown that it achieves higher tracking accuracy than that of the one-step-ahead motion strategy when the process noise variance decreases.

Following a brief review of existing work in Section 4.2, we present the problem formulation of the multi-step-ahead single-sensor single-target active tracking in Section 4.3. In Section 4.4, we relax the original non-convex optimization problem and

seek its approximate solution through GSR. Extensive simulations are presented in Section 4.5, while the conclusions of this chapter are summarized in Section 4.6.

4.2 Literature review

Multi-step-ahead (also called non-myopic or dynamic) active sensing has been studied and applied to the problems of environmental monitoring and surveillance [31, 94, 95, 96, 97, 98], target detection and tracking [93, 99].

Previous works on multi-step-ahead single-sensor active target tracking have exclusively relied on using bearing-only measurements. Based on the assumptions made regarding the target's motion model, these methods can be classified into two categories. The first class of approaches focuses on analytically determining the optimal sensing trajectory for a deterministic, but restricted, target motion model. In contrast, the second family of methods, applicable to arbitrary target motion models, computes the optimal sensing trajectory numerically. In what follows, we provide more details about each approach.

4.2.1 Analytical solutions for restricted target motion models

In [49], Le Cadre and Laurent-Michel proposed an approximate tracking algorithm, in which a single mobile sensor attempts to minimize the uncertainty about the target's location and velocity over a finite time horizon, by processing relative bearing measurements. Under the assumption that the distance between the sensor and the target is always kept (or approximately) constant, the objective function (i.e., the determinant of the Fisher Information Matrix – FIM) is significantly simplified, and the resulting control law requires that the sensor switches its bearing rate between its upper and lower bound.

In contrast to [49], where the optimization is performed in the discrete time domain, Passerieux and Van Cappel [50] formulated the optimal trajectory generation problem for single-sensor single-target tracking using bearing measurements in continuous time. In this case, however, the target is constrained to move deterministically on a straight line with constant velocity and the objective is to minimize the target's location and velocity uncertainty by maximizing the FIMs' determinant over a finite time horizon.

The authors presented the necessary condition for the continuous-time optimal sensor path based on the Euler equation.

The main drawback of the previous approaches is that the proposed analytical solutions are only valid for specific target-motion models (i.e., the target is restricted either to be at a constant distance from the sensor [49], or to move on a straight line with constant velocity [50]). By imposing such constraints on the target's motion, the objective function is often significantly simplified. In contrast, in our work, we are interested in the general case of arbitrary stochastic target motion models.

4.2.2 Numerical solutions for arbitrary target motion models

Current approaches that impose no requirements on the target motion model seek to solve the underlying optimization problem numerically. In particular, based on the solution strategy employed, these methods can be further classified into two categories: gradient-descent algorithms and grid-based exhaustive search.

Gradient-descent algorithm

Since active sensing can be formulated as an optimization problem, gradient-based algorithms have been applied for iteratively computing the sensor's trajectory.

Oshman and Davidson [52] addressed the single-sensor trajectory optimization problem for static target localization given constraints on the sensor's motion. The objective function employed was the determinant of the FIM and it was minimized, over a finite time horizon, using gradient descent, whose computational complexity is quadratic in the number of time steps considered. However, the impact of the step-size selection on the convergence of the algorithm was not examined. Moreover, the convergence rate of the gradient-based algorithm and the existence of local minima were not considered.

Grid-based exhaustive search

In [57], Logothetis *et al.* studied the single-sensor trajectory optimization from an information theory perspective, where the sensor attempts to reduce the target's location and velocity uncertainty using bearing measurements. The authors employed the determinant of the target's covariance matrix at the end of a finite time horizon as the

cost function, and computed the optimal solution by performing a grid-based exhaustive search. Acknowledging that the computational requirements increase exponentially with the number of time steps considered, the authors presented suboptimal solutions in [100], where the grid-based minimization takes place over only one time step.

In [58], Frew investigated the problem of single-sensor trajectory generation for target tracking using bearing measurements. In this case, motion constraints on the sensor's trajectory are explicitly incorporated in the problem formulation, and the objective function (determinant of the target's covariance matrix) is minimized over a finite time horizon using exhaustive search over a discretized set of candidate sensor headings. Due to the exponential computational requirements of the exhaustive search, relaxation or heuristic pruning were introduced to make the search method amenable to real-time implementations [58].

In [101], Hernandez addressed the problem of determining the optimal maneuver of an observer tracking a target using bearing-only measurements. In this case, the objective function is the maximum diagonal element of the inverse of the FIM. Since exhaustive search has exponential computational complexity in the number of time steps considered, the author proposed a hierarchic pruning scheme to seek suboptimal solutions that require less computational resources.

Recently, Chhetri *et al.* [102] proposed a non-myopic sensor scheduling algorithm for the active target tracking problem. The objective function is the determinant of the predicted posterior covariance matrix of the target's state estimate. The authors discretized the feasible set of candidate headings, and introduced an algorithm based on the branch-and-bound method to search for the optimal solution. However, the worst-case computational complexity of the branch-and-bound method is exponential in the number of time steps considered.

In summary, grid-based exhaustive search can handle both arbitrary target-motion models and the sensor's mobility constraints. Additionally, exhaustive search can find the global minimum in the limit as the cell size goes to zero. However, they have exponential computational complexity in the number of time steps considered, which prohibits their real-time implementation on mobile sensors.

4.3 Problem formulation

Consider a sensor moving in a plane and tracking the position of a moving target by processing distance observations. The problem setup for the multi-step-ahead optimization is similar to that of the multi-sensor optimization (see Section 3.3), where the target's state and its discrete-time dynamic equation are shown in (3.1) and (3.2), respectively. In addition, the motion of the tracking sensor is subject to (i) maximum-speed and (ii) collision-avoidance constraints (see Figure 3.2).

In contrast to the one-step-ahead motion strategy where we only seek the next best sensing location, in this chapter, we focus on determining the optimal motion strategy over a sliding window of fixed length K. Specifically, assuming at the current time step $k_0 = 0$, the target's state $\mathbf{x}_T(0)$ follows a distribution with mean $\hat{\mathbf{x}}_T(0)$ and covariance $\mathbf{P}_0$, our objective is to determine the optimal sensing locations at the next K steps, so as to minimize the target's position uncertainty at the K-th time step, subject to the sensor's maximum-speed and collision-avoidance constraints (see Figure 4.2). To do so, we first compute the predicted target's state estimates and their associated covariances over the next K time steps, described in the following section.

4.3.1 Predicted target's state estimate and prior covariance

The predicted target's state estimate $\hat{\mathbf{x}}_T(k) = \hat{\mathbf{x}}_T(k|0)$ at any future time-step k ($k = 1, \ldots, K$), based on the known target stochastic motion model (3.2), can be computed through the standard KF propagation equation as follows:

$$\hat{\mathbf{x}}_T(k) = \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}_T(k-1) = \mathbf{\Phi}_{k-1:0}\hat{\mathbf{x}}_T(0), \quad k = 1, \ldots, K, \tag{4.1}$$

where $\mathbf{\Phi}_{\ell:j}$ is defined as $\mathbf{\Phi}_{\ell:j} = \mathbf{\Phi}_\ell \mathbf{\Phi}_{\ell-1} \cdots \mathbf{\Phi}_j$ for $j < \ell$, $\mathbf{\Phi}_{\ell:j} = \mathbf{\Phi}_\ell$ for $j = \ell$, and $\mathbf{\Phi}_{\ell:j} = \mathbf{I}_{2N}$ for $j > \ell$.

Similarly, the predicted prior covariance matrix $\mathbf{P}_k^{\ominus} = \mathbf{P}_{k|0},\ k = 1, \ldots, K$, is propagated as

$$\begin{aligned}\mathbf{P}_k^{\ominus} &= \mathbb{E}\left[\left(\mathbf{x}_T(k) - \hat{\mathbf{x}}_T(k)\right)\left(\mathbf{x}_T(k) - \hat{\mathbf{x}}_T(k)\right)^{\mathrm{T}}\right] = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}^{\ominus}\mathbf{\Phi}_{k-1}^{\mathrm{T}} + \mathbf{Q}_{k-1} \\ &= \mathbf{\Phi}_{k-1:0}\mathbf{P}_0\mathbf{\Phi}_{k-1:0}^{\mathrm{T}} + \sum_{\ell=0}^{k-1} \mathbf{\Phi}_{k-1:k-\ell}\,\mathbf{Q}_{k-1-\ell}\,\mathbf{\Phi}_{k-1:k-\ell}^{\mathrm{T}}\,. \end{aligned} \tag{4.2}$$

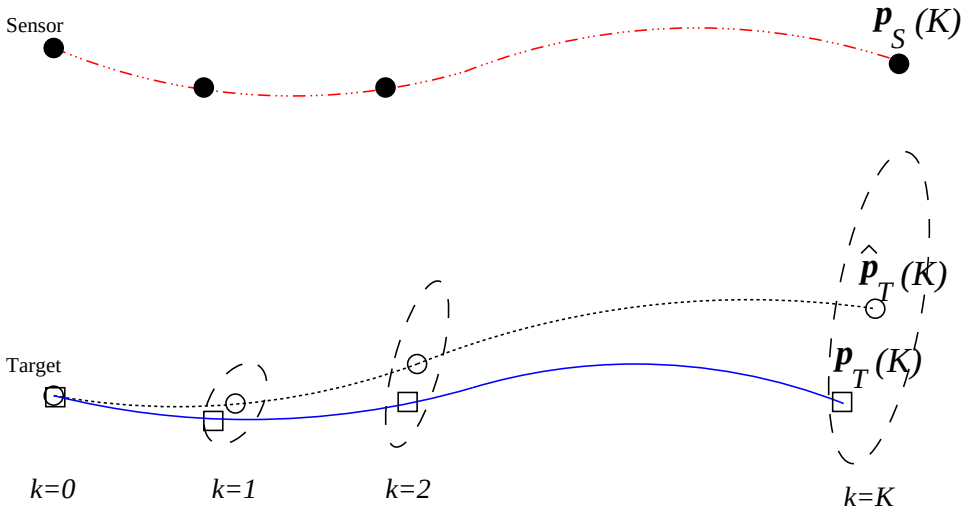


Figure 4.2: Illustration of the multi-step-ahead single-sensor active target tracking problem: The target's true trajectory is shown by the blue solid line, while its predicted trajectory is denoted by the dotted black line. Due to process noise, the predicted target's position at time-step k, $\hat{\mathbf{p}}_T(k)$, denoted by a circle, always differs from its true value $\mathbf{p}_T(k)$, shown as a square box. Additionally, the 3σ ellipse computed from the predicted prior covariance $\mathbf{P}_k^{\ominus}$, which characterizes the uncertainty of the estimate $\hat{\mathbf{p}}_T(k)$, is ever growing due to the process noise covariance $\mathbf{Q}_j$, $0 \leq j \leq k-1$. Our objective is to determine the optimal trajectory (shown by the red dashed line) for the tracking sensor over a sliding window of fixed length K (i.e., $\mathbf{p}_S(k)$, $k = 1, \ldots, K$, shown as solid circles or disks), so as to minimize the target's position uncertainty at the Kth time step, subject to the sensor's mobility constraints.

4.3.2 Predicted posterior covariance

We assume that the sensor can measure its relative distance to the target at every (future) time-step k ($k = 1, \ldots, K$). In what follows, $\mathbf{p}_T(k) = [x_T(k)\ y_T(k)]^{\mathrm{T}}$ and $\mathbf{p}_S(k) = [x_S(k)\ y_S(k)]^{\mathrm{T}}$ denote the true positions of the target and the sensor at time-step k, respectively, expressed in the global frame of reference. Furthermore, to simplify the notation, we introduce the variable $\mathbf{p}_k$ to denote the estimated (predicted) relative

position between the sensor and the target at time-step k, i.e.,

$$\mathbf{p}_k = \mathbf{p}_S(k) - \hat{\mathbf{p}}_T(k),\ k = 1, \ldots, K, \tag{4.3}$$

where $\hat{\mathbf{p}}_T(k) = \mathbf{W}_\mathbf{p}\hat{\mathbf{x}}_T(k)$ and $\mathbf{W}_\mathbf{p} = \begin{bmatrix}\mathbf{I}_2 & \mathbf{0}_{2\times 2(N-1)}\end{bmatrix}^{\mathrm{T}}$ is a projection matrix.

The distance measurement at time-step k is

$$z(k) = \|\mathbf{p}_T(k) - \mathbf{p}_S(k)\| + n(k),\ k = 1, \ldots, K,$$

which is corrupted by a zero-mean white (measurement) noise $n(k)$ with variance σ_k^2.

Similar to the multi-sensor case, the measurement matrix at time-step k is [see (3.16)]

$$\mathbf{h}_k = \begin{bmatrix}\frac{-\mathbf{p}_k^{\mathrm{T}}}{\|\mathbf{p}_k\|} & \mathbf{0}_{1\times 2(N-1)}\end{bmatrix}^{\mathrm{T}} = -\mathbf{W}_\mathbf{p}\frac{\mathbf{p}_k}{\|\mathbf{p}_k\|},\ k = 1, \ldots, K. \tag{4.4}$$

In this work, our objective is to minimize the target's position uncertainty at time-step K, quantified by the trace of the predicted posterior (updated) covariance for the target's position estimate. The posterior covariance for the target's position estimate $\hat{\mathbf{p}}_T(K)$, corresponding to the upper left 2×2 sub-matrix [see (3.1)] of the posterior covariance $\mathbf{P}_K^{\oplus}$ (of dimensions $2N \times 2N$) for the target's state estimate $\hat{\mathbf{x}}_T(K)$, is

$$\mathbf{P}_K^{\oplus}(1:2, 1:2) = \mathbf{W}_\mathbf{p}^{\mathrm{T}}\mathbf{P}_K^{\oplus}\mathbf{W}_\mathbf{p}. \tag{4.5}$$

It is well-known that $\mathbf{P}_K^{\oplus}$ can be computed analytically as a sub-matrix of the inverse of the Hessian matrix $\mathbf{H}$ [1] , a symmetric positive definite (SPD) matrix of dimensions $2N(K+1) \times 2N(K+1)$. More specifically, we have

$$\mathbf{P}_K^{\oplus} = \mathbf{W}^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{W}, \tag{4.6}$$

where the projection matrix $\mathbf{W}$ is defined as $\mathbf{W} = \begin{bmatrix}\mathbf{0}_{2N\times 2NK} & \mathbf{I}_{2N}\end{bmatrix}^{\mathrm{T}}$.

The Hessian matrix $\mathbf{H}$ has the following expression [104]

$$\begin{aligned}\mathbf{H} &= \mathbf{A} + \mathbf{blkdiag}\left(\mathbf{0}_{2N\times 2N},\ \bar{\sigma}_1^{-2}\mathbf{W}_\mathbf{p}\frac{\mathbf{p}_1\mathbf{p}_1^{\mathrm{T}}}{\mathbf{p}_1^{\mathrm{T}}\mathbf{p}_1}\mathbf{W}_\mathbf{p}^{\mathrm{T}},\ \ldots,\ \bar{\sigma}_K^{-2}\mathbf{W}_\mathbf{p}\frac{\mathbf{p}_K\mathbf{p}_K^{\mathrm{T}}}{\mathbf{p}_K^{\mathrm{T}}\mathbf{p}_K}\mathbf{W}_\mathbf{p}^{\mathrm{T}}\right)\\ &= \mathbf{A} + \sum_{k=1}^{K}\bar{\sigma}_k^{-2}\mathbf{W}_k\frac{\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k}\mathbf{W}_k^{\mathrm{T}},\end{aligned} \tag{4.7}$$

[1] The Hessian matrix $\mathbf{H}$ results from formulating this as a maximum-a-posterior (MAP) estimation problem, where the state vector to be estimated is $\mathbf{x}_T(0:K) = \left[\mathbf{x}_T^{\mathrm{T}}(0)\ldots\mathbf{x}_T^{\mathrm{T}}(K)\right]^{\mathrm{T}}$. Note that the EKF is equivalent to the MAP estimator when only one iteration of the Gauss-Newton algorithm is applied for solving it [103].

where $\mathbf{W}_k = \begin{bmatrix} \mathbf{0}_{2\times 2Nk} & \mathbf{W}_{\mathbf{p}}^{\mathrm{T}} & \mathbf{0}_{2\times 2N(K-k)} \end{bmatrix}^{\mathrm{T}},\ k = 1, \ldots, K.$

The $2N(K+1) \times 2N(K+1)$ SPD matrix $\mathbf{A}$ in (4.7), is independent of the relative position $\mathbf{p}_k,\ k = 1, \ldots, K$, and has the following banded structure with both upper and lower bandwidths of $2N$

$$\mathbf{A} = \tag{4.8}$$
$$\begin{bmatrix} \mathbf{P}_0^{-1} + \mathbf{\Phi}_0^{\mathrm{T}}\mathbf{Q}_0^{-1}\mathbf{\Phi}_0 & -\mathbf{\Phi}_0^{\mathrm{T}}\mathbf{Q}_0^{-1} & & & & \\ -\mathbf{Q}_0^{-1}\mathbf{\Phi}_0 & \mathbf{Q}_0^{-1} + \mathbf{\Phi}_1^{\mathrm{T}}\mathbf{Q}_1^{-1}\mathbf{\Phi}_1 & -\mathbf{\Phi}_1^{\mathrm{T}}\mathbf{Q}_1^{-1} & & & \\ & -\mathbf{Q}_1^{-1}\mathbf{\Phi}_1 & \mathbf{Q}_1^{-1} + \mathbf{\Phi}_2^{\mathrm{T}}\mathbf{Q}_2^{-1}\mathbf{\Phi}_2 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \mathbf{Q}_{K-2}^{-1} + \mathbf{\Phi}_{K-1}^{\mathrm{T}}\mathbf{Q}_{K-1}^{-1}\mathbf{\Phi}_{K-1} & -\mathbf{\Phi}_{K-1}^{\mathrm{T}}\mathbf{Q}_{K-1}^{-1} \\ & & & & -\mathbf{Q}_{K-1}^{-1}\mathbf{\Phi}_{K-1} & \mathbf{Q}_{K-1}^{-1} \end{bmatrix}$$

Note that from (4.7) and (4.8), the banded structure of $\mathbf{A}$ also implies that the Hessian $\mathbf{H}$ has the same banded structure with both upper and lower bandwidths of $2N$.

Combining (4.5), (4.6) and (4.7), we arrive at the objective function with respect to the relative positions $\mathbf{p}_k,\ k = 1, \ldots, K$, for the multi-step-ahead single-sensor single-target active tracking problem, i.e.,

$$\mathrm{tr}\left(\mathbf{P}_K^{\oplus}(1:2, 1:2)\right) = \mathrm{tr}\left(\mathbf{W}_K^{\mathrm{T}}\left(\mathbf{A} + \sum_{k=1}^{K} \bar{\sigma}_k^{-2}\mathbf{W}_k \frac{\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k}\mathbf{W}_k^{\mathrm{T}}\right)^{-1}\mathbf{W}_K\right), \tag{4.9}$$

where $\mathbf{W}_K = \mathbf{W}_{\mathbf{p}}\mathbf{W} = \begin{bmatrix} \mathbf{0}_{2\times 2NK} & \mathbf{W}_{\mathbf{p}}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$.

4.3.3 Measurement variance inflation

Typically, the variance $\bar{\sigma}_k^2,\ k = 1, \ldots, K$, in (4.7) is set to be

$$\bar{\sigma}_k^2 = \mathrm{Cov}\left(z(k)|\mathbf{x}_T(k)\right) = \sigma_k^2, \tag{4.10}$$

under the assumption that the target's true state at time-step k, i.e., $\mathbf{x}_T(k)$, is known. However, at time-step 0, the only available information is the predicted target's state $\hat{\mathbf{x}}_T(k)$ at time-step k, $k = 1, \ldots, K$, computed in (4.1).

Furthermore, note that the k-th term $\bar{\sigma}_k^{-2}\mathbf{W}_k \frac{\mathbf{p}_k\mathbf{p}_k^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k}\mathbf{W}_k^{\mathrm{T}}$ which appears in the summation (4.7) corresponds to the expected information contributed from the distance measurement at time-step k. This information is a (matrix) function of $\mathbf{p}_k$, while $\mathbf{p}_k$

depends on two variables [see (4.3)]: (i) the sensing location $\mathbf{p}_S(k)$ which we have control over and (ii) the predicted target's state $\hat{\mathbf{x}}_T(k)$ [or its position $\hat{\mathbf{p}}_T(k)$] that is a random variable.

To compute $\mathbf{H}$ from (4.7), we assume that the target's true position $\mathbf{p}_T(k)$ at time-step k is the same as its predicted position $\hat{\mathbf{p}}_T(k)$. However, regardless of how accurate the tracking model [see (3.2)] is, due to process noise, $\hat{\mathbf{p}}_T(k)$ will always differ from $\mathbf{p}_T(k)$ (see Figure 4.2). In order to reflect the estimation uncertainty of $\hat{\mathbf{p}}_T(k)$, and take into account the fact that the distance measurement $z(k)$ at any future time-step $k,\ k = 1, \ldots, K$, is unavailable at time-step 0, we propose to modify (4.10) as

$$\bar{\sigma}_k^2 = \mathrm{Cov}\left(z(k)|\mathbf{x}_T(0)\right) = \sigma_k^2 + \frac{1}{\mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k}\mathbf{p}_k^{\mathrm{T}}\mathbf{W}_{\mathbf{p}}^{\mathrm{T}}\mathbf{P}_k^{\ominus}\mathbf{W}_{\mathbf{p}}\mathbf{p}_k,\ k = 1, \ldots, K. \tag{4.11}$$

Unfortunately, (4.11) is a function that explicitly depends on $\mathbf{p}_k$, and hence, the control parameter $\mathbf{p}_S(k)$. In order to eliminate its dependence on $\mathbf{p}_S(k)$, we therefore seek an upper bound of the right-hand-side of (4.11). To do so, we notice that $\frac{1}{\mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k}\mathbf{p}_k^{\mathrm{T}}\mathbf{W}_{\mathbf{p}}^{\mathrm{T}}\mathbf{P}_k^{\ominus}\mathbf{W}_{\mathbf{p}}\mathbf{p}_k \leq \frac{1}{\mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k}\mathbf{p}_k^{\mathrm{T}}\lambda_{\max}\left(\mathbf{W}_{\mathbf{p}}^{\mathrm{T}}\mathbf{P}_k^{\ominus}\mathbf{W}_{\mathbf{p}}\right)\mathbf{I}_2\mathbf{p}_k = \lambda_{\max}\left(\mathbf{W}_{\mathbf{p}}^{\mathrm{T}}\mathbf{P}_k^{\ominus}\mathbf{W}_{\mathbf{p}}\right)$, and thus we propose to inflate the distance measurement noise variance σ_k^2 as

$$\bar{\sigma}_k^2 = \sigma_k^2 + \lambda_{\max}\left(\mathbf{W}_{\mathbf{p}}^{\mathrm{T}}\mathbf{P}_k^{\ominus}\mathbf{W}_{\mathbf{p}}\right),\ k = 1, \ldots, K. \tag{4.12}$$

In the simulations (see Section 4.5), we will demonstrate the effectiveness of the measurement-noise-variance inflation proposed in (4.12).

4.3.4 Problem statement

As evident from (4.3) and (4.9), the target's position uncertainty at time-step K will depend, through the relative position $\mathbf{p}_k$, on all the sensing locations at the next K time steps $\mathbf{p}_S(k),\ k = 1, \ldots, K$. Similar to the multi-sensor optimization problem, the sensor is subject to (i) maximum-speed and (ii) collision-avoidance constraints (see Figure 3.2), i.e.,

$$\|\mathbf{p}_S(k) - \mathbf{p}_S(k-1)\| \leq r_{k-1},$$
$$\|\mathbf{p}_S(k) - \hat{\mathbf{p}}_T(k)\| \geq \rho_k.$$

Substituting $\mathbf{p}_k$ [see (4.3)] in the above two inequalities, yields

$$\|\mathbf{p}_k - \mathbf{p}_{k-1} - \mathbf{c}_{k-1}\| \leq r_{k-1}, \tag{4.13}$$

$$\|\mathbf{p}_k\| \geq \rho_k, \tag{4.14}$$

where $\mathbf{c}_{k-1}$ is a known 2-D vector defined as $\mathbf{c}_{k-1} = \hat{\mathbf{p}}_T(k) - \hat{\mathbf{p}}_T(k-1),\ k = 1, \ldots, K$.

Note that once $\mathbf{p}_k,\ k = 1, \ldots, K$, is determined, the sensing location at time-step k, $\mathbf{p}_S(k),\ k = 1, \ldots, K$, can be obtained through (4.1) and (4.3). Therefore, the multi-step-ahead problem is equivalent to determining the set $\{\mathbf{p}_k,\ k = 1, \ldots, K\}$, that minimizes the trace of the (predicted) target's position estimate (posterior) covariance matrix [see (4.9)], under the constraints specified in (4.13)–(4.14):

- Optimization Problem 10 (Π_{10})

$$\underset{\mathbf{p}_1,\ldots,\mathbf{p}_K}{\text{minimize}} \quad f(\mathbf{p}_1, \ldots, \mathbf{p}_K) = \text{tr}\left(\mathbf{W}_K^{\text{T}}\left(\mathbf{A} + \sum_{k=1}^{K} \bar{\sigma}_k^{-2}\mathbf{W}_k \frac{\mathbf{p}_k\mathbf{p}_k^{\text{T}}}{\mathbf{p}_k^{\text{T}}\mathbf{p}_k}\mathbf{W}_k^{\text{T}}\right)^{-1}\mathbf{W}_K\right) \tag{4.15}$$

$$\text{subject to} \quad \|\mathbf{p}_k - \mathbf{p}_{k-1} - \mathbf{c}_{k-1}\|^2 \leq r_{k-1}^2, \tag{4.16}$$

$$\|\mathbf{p}_k\|^2 \geq \rho_k^2, \quad k = 1, \ldots, K, \tag{4.17}$$

with $\bar{\sigma}_k^2$ defined in (4.12).

Remark 16. *Similar to the multi-sensor optimization problem, the multi-step-ahead active target tracking is a non-convex optimization problem, since both the objective function and the feasible set are generally non-convex. Furthermore, the multi-step-ahead optimization problem poses additional challenges.*

- *Lemma 4 in Section 3.3.3, which significantly simplifies the covariance update of the target's position (i.e., only requires inverting a 2×2 matrix $\mathbf{P}_{k+1|k,11}$), is no longer valid for the multi-step-ahead optimization problem when $K \geq 2$. In contrast, the size of the Hessian matrix $\mathbf{H}$ being inverted in* (4.15) *grows linearly in K. Additionally, the stochastic model employed for describing the target's motion, i.e., the matrices $\mathbf{\Phi}_k,\ k = 1, \ldots, K$, will explicitly affect the objective function* (4.15) *through $\mathbf{A}$ [see* (4.8)*].*

- *Recall that the mobility constraints in the multi-sensor one-step-ahead active target tracking problem* [*see* (3.34)–(3.35)] *are decoupled. In contrast, for the multi-step-ahead single-sensor optimization problem, due to the maximum-speed constraints which involve the sensing locations at the consecutive time-step* $k-1$ *and* k ($k = 1, \ldots, K$)*, the optimization variables* $\{\mathbf{p}_k,\ k = 1, \ldots, K\}$ *are no longer decoupled* [*see* (4.16)].

4.3.5 Problem analysis

We first examine the structure of the objective function f in Π_{10} [see (4.15)]. In the following derivations, we use the definition $\mathbf{p}_k = \begin{bmatrix} x_k & y_k \end{bmatrix}^{\mathrm{T}},\ k = 1, \ldots, K$.

Firstly, it is straightforward to conclude that each element of the Hessian matrix $\mathbf{H}$ [see (4.7)] is either a constant or a rational function in x_k and y_k, $k = 1, \ldots, K$. Since additions and multiplications of rational functions yield rational functions, we conclude that the objective function f is rational with respect to $x_1, y_1, \ldots, x_K, y_K$, and can be written in a generic rational form as

$$f(\mathbf{p}_1, \ldots, \mathbf{p}_K) = \frac{\sum_i c_i \prod_{k=1}^{K} x_k^{\alpha_k^i} y_k^{\beta_k^i}}{\sum_j d_j \prod_{k=1}^{K} x_k^{\gamma_k^j} y_k^{\delta_k^j}}, \tag{4.18}$$

where $\alpha_k^i, \beta_k^i, \gamma_k^j, \delta_k^j$ are non-negative integers for all i, j and $k = 1, \ldots, K$, and c_i and d_j are known real numbers whose values depend on $\mathbf{A}$ and $\bar{\sigma}_k^2, k = 1, \ldots, K$.

Interestingly, by exploiting the structure of (4.15), it can be shown that the degree of each monomial in (4.18), i.e., $\sum_{k=1}^{K} \left(\alpha_k^i + \beta_k^i\right)$ and $\sum_{k=1}^{K} \left(\gamma_k^j + \delta_k^j\right)$, is always $2K$. More specifically, $\alpha_k^i + \beta_k^i = 2$ and $\gamma_k^j + \delta_k^j = 2$ for all i, j and $k = 1, \ldots, K$. We summarize this result in the following proposition.

Proposition 17. *For the multi-step-ahead single-sensor single-target active tracking using distance-only observations, the cost function defined by* (4.15) *is a rational function with respect to the optimization variables* $x_1, y_1, \ldots, x_K, y_K$*, i.e.,*

$$f(\mathbf{p}_1, \ldots, \mathbf{p}_K) = \frac{\sum_i c_i \prod_{k=1}^{K} x_k^{\alpha_k^i} y_k^{\beta_k^i}}{\sum_i d_i \prod_{k=1}^{K} x_k^{\alpha_k^i} y_k^{\beta_k^i}}. \tag{4.19}$$

Furthermore, the degree of each monomial in (4.19) *is* $2K$*. More specifically,* $\forall\ i$*,*

$$\alpha_k^i + \beta_k^i = 2, \quad k = 1, \ldots, K.$$

Proof. To prove Proposition 17, it is equivalent to show that by treating all $\mathbf{p}_\ell, \ell \neq k$ as (fixed) parameters, the cost function f with respect to $\mathbf{p}_k$ can be expressed as

$$f(\mathbf{p}_k) = \frac{\mathbf{p}_k^{\mathrm{T}} \left(\kappa_{-k}\mathbf{D}_{-k} - \mathbf{C}_{-k}\right) \mathbf{p}_k}{\mathbf{p}_k^{\mathrm{T}}\mathbf{D}_{-k}\mathbf{p}_k} = \kappa_{-k} - \frac{\mathbf{p}_k^{\mathrm{T}}\mathbf{C}_{-k}\mathbf{p}_k}{\mathbf{p}_k^{\mathrm{T}}\mathbf{D}_{-k}\mathbf{p}_k}, \tag{4.20}$$

where κ_{-k} is a scalar, $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ are 2×2 matrices with subscript "$-k$" highlighting that they are functions of $\{\mathbf{p}_\ell, \ell \neq k\}$, and are independent of the variable $\mathbf{p}_k$.

To show (4.20), we use the definition of f in (4.15). More specifically, we define

$$\mathbf{A}_{-k} = \mathbf{A} + \sum_{\ell \neq k} \bar{\sigma}_\ell^{-2} \mathbf{W}_\ell \frac{\mathbf{p}_\ell \mathbf{p}_\ell^{\mathrm{T}}}{\mathbf{p}_\ell^{\mathrm{T}}\mathbf{p}_\ell} \mathbf{W}_\ell^{\mathrm{T}}, \tag{4.21}$$

which is a matrix function independent of $\mathbf{p}_k$.

Then,

$$f(\mathbf{p}_k) = \mathrm{tr}\left(\mathbf{W}_K^{\mathrm{T}} \left(\mathbf{A}_{-k} + \bar{\sigma}_k^{-2} \mathbf{W}_k \frac{\mathbf{p}_k \mathbf{p}_k^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k} \mathbf{W}_k^{\mathrm{T}} \right)^{-1} \mathbf{W}_K \right). \tag{4.22}$$

Based on the matrix inversion lemma [105],

$$\left(\mathbf{A}_{-k} + \bar{\sigma}_k^{-2} \mathbf{W}_k \frac{\mathbf{p}_k \mathbf{p}_k^{\mathrm{T}}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k} \mathbf{W}_k^{\mathrm{T}} \right)^{-1} = \mathbf{A}_{-k}^{-1} - \frac{\mathbf{A}_{-k}^{-1}\mathbf{W}_k \mathbf{p}_k \mathbf{p}_k^{\mathrm{T}} \mathbf{W}_k^{\mathrm{T}} \mathbf{A}_{-k}^{-1}}{\mathbf{p}_k^{\mathrm{T}}\mathbf{W}_k^{\mathrm{T}}\mathbf{A}_{-k}^{-1}\mathbf{W}_k\mathbf{p}_k + \bar{\sigma}_k^2 \mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k}. \tag{4.23}$$

Substituting (4.23) into (4.22), and use the identity $\mathrm{tr}\left(\mathbf{X}^{\mathrm{T}}\mathbf{Y}\right) = \mathrm{tr}\left(\mathbf{Y}\mathbf{X}^{\mathrm{T}}\right)$ for arbitrary matrices $\mathbf{X}$ and $\mathbf{Y}$ of the same dimensions, we can simplify (4.22) as

$$f(\mathbf{p}_k) = \mathrm{tr}\left(\mathbf{W}_K^{\mathrm{T}}\mathbf{A}_{-k}^{-1}\mathbf{W}_K\right) - \frac{\mathbf{p}_k^{\mathrm{T}}\left(\mathbf{W}_k^{\mathrm{T}}\mathbf{A}_{-k}^{-1}\mathbf{W}_K\mathbf{W}_K^{\mathrm{T}}\mathbf{A}_{-k}^{-1}\mathbf{W}_k\right)\mathbf{p}_k}{\mathbf{p}_k^{\mathrm{T}}\left(\mathbf{W}_k^{\mathrm{T}}\mathbf{A}_{-k}^{-1}\mathbf{W}_k + \bar{\sigma}_k^2\mathbf{I}_2\right)\mathbf{p}_k}. \tag{4.24}$$

Comparing (4.24) and (4.20), we immediately arrive at

$$\kappa_{-k} = \mathrm{tr}\left(\mathbf{W}_K^{\mathrm{T}}\mathbf{A}_{-k}^{-1}\mathbf{W}_K\right), \tag{4.25}$$

$$\mathbf{C}_{-k} = \mathbf{W}_k^{\mathrm{T}}\mathbf{A}_{-k}^{-1}\mathbf{W}_K\mathbf{W}_K^{\mathrm{T}}\mathbf{A}_{-k}^{-1}\mathbf{W}_k, \tag{4.26}$$

$$\mathbf{D}_{-k} = \mathbf{W}_k^{\mathrm{T}}\mathbf{A}_{-k}^{-1}\mathbf{W}_k + \bar{\sigma}_k^2\mathbf{I}_2. \tag{4.27}$$

Since $\mathbf{A}_{-k}$ (and hence $\mathbf{A}_{-k}^{-1}$) is a matrix function of the variables $\{\mathbf{p}_\ell, \ell \neq k\}$ [see (4.21)], from the above relationships, we conclude that κ_{-k}, $\mathbf{C}_{-k}$, and $\mathbf{D}_{-k}$ are independent of $\mathbf{p}_k$. This concludes the proof of Proposition 17. □

From Proposition 17, we conclude that both numerator and denominator of the objective function f [see (4.19)] involve a summation of 3^K monomials. Since the constraints of Π_{10} [see (4.16)–(4.17)] are polynomial inequalities with respect to $\mathbf{p}_k,\ k = 1,\ldots,K$, we conclude that the KKT (necessary) optimality conditions of Π_{10} can be equivalently transformed into a system of multivariate polynomial equations with respect to $\mathbf{p}_1,\ldots,\mathbf{p}_K$ and $2K$ scalar Lagrangian multipliers [since there are $2K$ scalar inequalities defined by (4.16)–(4.17)], whose degree and the number of unknowns increase linearly in K. Unfortunately, the computational complexity of solving multivariate polynomial systems is double exponential in both the degree and the number of unknowns. Thus, it is computationally intractable to calculate the global optimal solution of Π_{10} by directly solving its KKT optimality conditions.

In order to design efficient algorithms that operate in real time, appropriate relaxations of Π_{10} become necessary. In what follows, we first derive the closed-form solution for the single-step problem, i.e., optimizing the objective function f with respect to $\mathbf{p}_k$ only (see Section 4.4.1). We then propose a Gauss-Seidel relaxation (GSR), also called cyclic block-coordinate descent algorithm, to compute an approximate solution for the multi-step-ahead optimization problem Π_{10} (see Section 4.4.2), whose computational complexity is quadratic in the number of time steps considered.

4.4 Solution strategy

4.4.1 Solution strategy for single-step optimization

Motivated by the simplicity of the cost function f in $\mathbf{p}_k$ [see (4.20)], we first study the single-step optimization problem with respect to $\mathbf{p}_k$, by fixing all other variables $\mathbf{p}_\ell,\ \ell \neq k$, in Π_{10}. Since κ_{-k} is independent of $\mathbf{p}_k$ [see (4.25)], we have

$$\arg\min_{\mathbf{p}_k}\ f(\mathbf{p}_k) = \arg\max_{\mathbf{p}_k}\ \frac{\mathbf{p}_k^{\mathrm{T}}\mathbf{C}_{-k}\mathbf{p}_k}{\mathbf{p}_k^{\mathrm{T}}\mathbf{D}_{-k}\mathbf{p}_k}.$$

Therefore, the single-step optimization problem with respect to $\mathbf{p}_k$ can be equivalently formulated as

- Optimization Problem 11 (Π_{11})

$$\underset{\mathbf{p}_k}{\text{maximize}} \quad \frac{\mathbf{p}_k^{\mathrm{T}}\mathbf{C}_{-k}\mathbf{p}_k}{\mathbf{p}_k^{\mathrm{T}}\mathbf{D}_{-k}\mathbf{p}_k} \tag{4.28}$$

$$\text{subject to} \quad \|\mathbf{p}_k - \bar{\mathbf{c}}_{k-1}\|^2 \leq r_{k-1}^2, \tag{4.29}$$

$$\|\mathbf{p}_k\|^2 \geq \rho_k^2, \tag{4.30}$$

where $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ are defined in (4.26) and (4.27), respectively, and $\bar{\mathbf{c}}_{k-1} = \mathbf{p}_{k-1} + \mathbf{c}_{k-1}$ is a known 2-D vector.

To solve Π_{11}, we note that due to the (strict) positive-definiteness of $\mathbf{D}_{-k}$ [see (4.27)], the cost function of Π_{11} [see (4.28)] is a generalized Rayleigh quotient, whose value only depends on the polar angle of $\mathbf{p}_k$ (i.e., φ_k) and is independent of the radius of $\mathbf{p}_k$ (i.e., $\|\mathbf{p}_k\|$). More specifically,

$$\frac{\mathbf{p}_k^{\mathrm{T}}\mathbf{C}_{-k}\mathbf{p}_k}{\mathbf{p}_k^{\mathrm{T}}\mathbf{D}_{-k}\mathbf{p}_k} = \frac{\mathbf{v}_k^{\mathrm{T}}\mathbf{C}_{-k}\mathbf{v}_k}{\mathbf{v}_k^{\mathrm{T}}\mathbf{D}_{-k}\mathbf{v}_k}, \tag{4.31}$$

where $\mathbf{v}_k = \frac{\mathbf{p}_k}{\|\mathbf{p}_k\|} = \begin{bmatrix}\cos\varphi_k & \sin\varphi_k\end{bmatrix}^{\mathrm{T}}$ is a unit vector in 2-D.

On the other hand, from planar geometry, it is straightforward to conclude that the feasible set of $\mathbf{p}_k$ defined by (4.29)–(4.30) can be expressed as an interval (or box) constraint on φ_k, i.e., $\varphi_k \in [\varphi_k^{\min}\ ,\ \varphi_k^{\max}]$, where $\varphi_k^{\min}$ and $\varphi_k^{\max}$ are determined based on the configuration of the feasible set of $\mathbf{p}_k$. Similarly to the single-sensor problem addressed in Section 3.5, there are four cases of the feasible set that we need to address, shown in Figures 3.6(a)–3.6(d).

- Case I: $0 \leq \rho_k \leq \|\bar{\mathbf{c}}_{k-1}\| - r_{k-1}$

  As shown in Figure 3.6(a), the feasible set of $\mathbf{p}_k$ is defined by the maximum-speed constraint [see (4.29)], while the collision-avoidance constraint [see (4.30)] is redundant. Therefore, the upper and lower bounds of φ_k are φ_A and φ_B, the polar angles of A and B, respectively, where A and B are the two tangent points residing in the circle $\|\mathbf{p}_k - \bar{\mathbf{c}}_{k-1}\| = r_{k-1}$. From the geometry of the problem, we have

$$\varphi_k^{\min} = \varphi_A = \varphi_C - \omega, \tag{4.32}$$

$$\varphi_k^{\max} = \varphi_B = \varphi_C + \omega, \tag{4.33}$$

where $\varphi_C = \arctan\left(\frac{c_y}{c_x}\right)$ (with $\bar{\mathbf{c}}_{k-1} = \begin{bmatrix} c_x & c_y \end{bmatrix}^{\mathrm{T}}$) is the polar coordinate of C, the center of the circle $\|\mathbf{p}_k - \bar{\mathbf{c}}_{k-1}\| = r_{k-1}$, and $\omega = \arcsin\left(\frac{r_{k-1}}{\|\bar{\mathbf{c}}_{k-1}\|}\right)$.

- Case II: $\sqrt{\|\bar{\mathbf{c}}_{k-1}\|^2 - r_{k-1}^2} \le \rho_k \le \|\bar{\mathbf{c}}_{k-1}\| + r_{k-1}$
  In this case, the intersection points E and F between the two circles (i.e., $\|\mathbf{p}_k\| = \rho_k$ and $\|\mathbf{p}_k - \bar{\mathbf{c}}_{k-1}\| = r_{k-1}$) are beyond the tangent points A and B. From the geometry of the problem, the upper and lower bounds of φ_k are φ_E and φ_F, the polar angles of E and F, respectively, i.e.,

$$\varphi_k^{\min} = \varphi_E = \varphi_C - \varpi, \tag{4.34}$$

$$\varphi_k^{\max} = \varphi_F = \varphi_C + \varpi, \tag{4.35}$$

  where ϖ can be computed by applying the law of cosines to the triangle OCE (or the triangle OCF) as $\varpi = \arccos\left(\frac{\rho_k^2 + \|\bar{\mathbf{c}}_{k-1}\|^2 - r_{k-1}^2}{2\rho_k \|\bar{\mathbf{c}}_{k-1}\|}\right)$.

- Case III: $\|\bar{\mathbf{c}}_{k-1}\| - r_{k-1} \le \rho_k \le \sqrt{\|\bar{\mathbf{c}}_{k-1}\|^2 - r_{k-1}^2}$
  Since the intersection points E and F are before the tangent points A and B, the upper and lower bounds of φ_k are the same as in Case I, i.e., $\varphi_k^{\min} = \varphi_A$ and $\varphi_k^{\max} = \varphi_B$ [see (4.32)–(4.33)].

- Case IV: $\|\bar{\mathbf{c}}_{k-1}\| + r_{k-1} \le \rho_k$
  Obviously in this case the feasible set is empty and thus we require the sensor to move to D', as shown in Figure 3.6(d). This ensures that (i) the sensor maintains the largest possible distance from the target so as to avoid collision, and (ii) it satisfies the maximum-speed constraint (4.29).

In summary, the optimization problem Π_{11} over the relative position $\mathbf{p}_k$ is equivalent to the following optimization problem Π_{12} over the polar angle φ_k.

- Optimization Problem 12 (Π_{12})

$$\underset{\varphi_k}{\text{maximize}} \quad f(\varphi_k) = \frac{\mathbf{v}_k^{\mathrm{T}} \mathbf{C}_{-k} \mathbf{v}_k}{\mathbf{v}_k^{\mathrm{T}} \mathbf{D}_{-k} \mathbf{v}_k} \tag{4.36}$$

$$\text{subject to} \quad \mathbf{v}_k = \begin{bmatrix} \cos\varphi_k & \sin\varphi_k \end{bmatrix}^{\mathrm{T}},$$

$$\varphi_k^{\min} \le \varphi_k \le \varphi_k^{\max}, \tag{4.37}$$

where $\varphi_k^{\min}$ and $\varphi_k^{\max}$ are defined by either (4.32)–(4.33) or (4.34)–(4.35).

The optimal solution of Π_{12} can be computed in closed form. More specifically, it is well-known that the optimal unit vector $\mathbf{v}_k^*$ that maximizes the cost function (4.36) under no constraints is simply the unit eigenvector corresponding to the maximum eigenvalue of $\mathbf{D}_{-k}^{-1}\mathbf{C}_{-k}$, i.e.,

$$\mathbf{D}_{-k}^{-1}\mathbf{C}_{-k}\mathbf{v}_k^* = \lambda_{\max}\left(\mathbf{D}_{-k}^{-1}\mathbf{C}_{-k}\right)\mathbf{v}_k^*.$$

Therefore,

$$\underline{\varphi}_k = \arctan\left(\frac{v_y}{v_x}\right),$$
$$\overline{\varphi}_k = \arctan\left(\frac{v_y}{v_x}\right) - \text{sign}(v_x)\pi,$$

with $\mathbf{v}_k^* = \begin{bmatrix} v_x & v_y \end{bmatrix}^{\mathrm{T}}$, are both the globally optimal solutions for the unconstrained optimization problem obtained by removing the interval constraint (4.37) from Π_{12}.

If either $\underline{\varphi}_k$ or $\overline{\varphi}_k$ satisfies the constraint (4.37), then the optimal solution φ_k^* of Π_{12} is either $\underline{\varphi}_k$ or $\overline{\varphi}_k$. Otherwise, since each stationary point of the generalized Rayleigh quotient in 2-D is either the global maximum or minimum, the optimal solution φ_k^* of Π_{12} is attained at either $\varphi_k^{\min}$ or $\varphi_k^{\max}$, whichever yields a larger objective value for Π_{12}.

In summary, the global optimal solution of Π_{12} is

$$\varphi_k^* = \begin{cases} \underline{\varphi}_k \text{ (or } \overline{\varphi}_k), & \text{if } \ \underline{\varphi}_k \text{ (or } \overline{\varphi}_k) \in [\varphi_k^{\min} \ , \ \varphi_k^{\max}], \\ \varphi_k^{\min}, & \text{else if } \ f(\varphi_k^{\min}) \geq f(\varphi_k^{\max}), \\ \varphi_k^{\max}, & \text{else if } \ f(\varphi_k^{\min}) < f(\varphi_k^{\max}). \end{cases} \tag{4.38}$$

Once φ_k^* is obtained, we compute the globally optimal solution $\mathbf{p}_k^*$ of Π_{11} as the closest intersection point towards the origin between the feasible set of Π_{11} and the radial line whose angle is φ_k^*.

In what follows, we analyze the computational complexity of the single-step optimization problem Π_{11}.

Complexity analysis of the single-step optimization

Assuming that $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ are known, calculation of the 2-D unit eigenvector $\mathbf{v}_k^*$, the unconstrained global optima $\underline{\varphi}_k$ and $\overline{\varphi}_k$, as well as the upper and lower bounds $\varphi_k^{\min}$

and $\varphi_k^{\max}$, requires constant number of arithmetic operations. Therefore, given $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$, the computational complexity for determining φ_k^* through (4.38) is constant and independent of K.

The expressions of the 2×2 matrices $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ are provided in (4.26) and (4.27), respectively. From (4.26)–(4.27), computing $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ involves the inversion of $\mathbf{A}_{-k}$, a $2N(K+1)\times 2N(K+1)$ SPD matrix, which in general has cubic computational complexity in the size of $\mathbf{A}_{-k}$, i.e., cubic in both N and K.[2] However, exploiting the fact that $\mathbf{A}_{-k}$ has a banded structure of both upper and lower bandwidths of $2N$, $\mathbf{A}_{-k}^{-1}$ can be efficiently computed through the Cholesky decomposition on $\mathbf{A}_{-k}$ with a cost only quadratic in K (and cubic in N) [105, Section 4.3]. However, $\mathbf{A}_{-k}^{-1}$ is in general a dense matrix and the storage of $\mathbf{A}_{-k}^{-1}$ requires space complexity quadratic in both N and K. Therefore, to compute $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ by using the explicit (and full) expression of $\mathbf{A}_{-k}^{-1}$ has complexity quadratic in the number of time steps considered.

In what follows, we develop an efficient algorithm that reduces the complexity of computing $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ from quadratic to linear (in K). The key idea is to avoid computing the full expression of $\mathbf{A}_{-k}^{-1}$. To proceed, we note that from (4.26)–(4.27), $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ can be computed as

$$\mathbf{C}_{-k} = \mathbf{B}_{-k}^{\mathrm{T}}\mathbf{W}_K\mathbf{W}_K^{\mathrm{T}}\mathbf{B}_{-k}, \tag{4.39}$$

$$\mathbf{D}_{-k} = \mathbf{W}_k^{\mathrm{T}}\mathbf{B}_{-k} + \bar{\sigma}_k^2\mathbf{I}_2, \tag{4.40}$$

where

$$\mathbf{B}_{-k} = \mathbf{A}_{-k}^{-1}\mathbf{W}_k \tag{4.41}$$

is a $2N(K+1)\times 2$ matrix.

Note that once $\mathbf{B}_{-k}$ is known, computing $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ using (4.39)–(4.40) requires constant number of arithmetic operations, due to the sparse structures of $\mathbf{W}_k$ and $\mathbf{W}_K$. In order to avoid computing the full expression of $\mathbf{A}_{-k}^{-1}$ in (4.41), we notice that from (4.41), $\mathbf{B}_{-k}$ can be equivalently obtained by solving the following linear equation

$$\mathbf{A}_{-k}\mathbf{B}_{-k} = \mathbf{W}_k. \tag{4.42}$$

[2] The matrix $\mathbf{A}_{-k}$ itself can be recursively updated from $\mathbf{A}_{-(k-1)}$ in constant number of arithmetic operations.

Solving $\mathbf{B}_{-k}$ from (4.42) can be efficiently achieved through the Cholesky decomposition on $\mathbf{A}_{-k}$. Specifically, by exploiting the banded structure of $\mathbf{A}_{-k}$, the overall complexity of solving (4.42) grows only linearly in K (and cubic in N) [105, Section 4.3.5]. Additionally, the number of memory cells required for storing $\mathbf{B}_{-k}$ also increases linearly with respect to K.

In conclusion, we have shown that computing $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ through (4.42), (4.39)–(4.40) can be achieved with a cost linear in K. Therefore, we conclude that the computational complexity of the single-step optimization problem Π_{11} is linear in the number of time steps considered, with the most demanding (or dominant) operation being the computation of the $2N(K+1) \times 2$ matrix $\mathbf{B}_{-k}$ from (4.42).

For completeness, we outline the algorithmic flow chart for computing the global optimum of the single-step optimization problem Π_{11} in Algorithm 3, along with the computational cost associated with each step.

Algorithm 3 Single-step optimization

Require: The matrices $\mathbf{A}_{-k}$ of dimensions $2N(K+1) \times 2N(K+1)$, $\mathbf{W}_k$ and $\mathbf{W}_K$ of dimensions $2N(K+1) \times 2$, the 2-D vector $\bar{\mathbf{c}}_{k-1}$, the scalars r_{k-1}, ρ_k, and $\bar{\sigma}_k^2$.

Ensure: The global optimum $\mathbf{p}_k^*$ of the optimization problem Π_{11}.

1: Solve for $\mathbf{B}_{-k}$ from the linear equation (4.42) [Complexity: $\mathcal{O}(K)$].

2: Compute $\mathbf{C}_{-k}$ and $\mathbf{D}_{-k}$ from (4.39)–(4.40), and the unit eigenvector $\mathbf{v}_k^*$, as well as the unconstrained global optima $\underline{\varphi}_k$ and $\overline{\varphi}_k$ [Complexity: $\mathcal{O}(1)$].

3: Determine the lower and upper bounds $\varphi_k^{\min}$ and $\varphi_k^{\max}$ based on either (4.32)–(4.33), or (4.34)–(4.35) [Complexity: $\mathcal{O}(1)$].

4: Calculate φ_k^* from (4.38) [Complexity: $\mathcal{O}(1)$].

5: **return** The global optimum $\mathbf{p}_k^*$ (whose polar angle is φ_k^*) of the optimization problem Π_{11} [Complexity: $\mathcal{O}(1)$].

4.4.2 Solution strategy for the multi-step optimization

Motivated by the simplicity of the closed-form solution for the single-step optimization problem Π_{11} (see Section 4.4.1), a straightforward approach to solve the original multi-step-ahead optimization problem Π_{10} is to iteratively minimize the objective function of Π_{10} [see (4.15)] for each optimization variable $\mathbf{p}_k,\ k = 1, \ldots, K$, separately. Specifically,

the solution of Π_{10} is acquired by employing the nonlinear Gauss-Seidel algorithm [91, Ch. 3], which requires to solve the following single-step optimization problem at each iteration step.

- Optimization Problem 13 (Π_{13})

$$\begin{aligned}
&\underset{\mathbf{p}_k^{(\ell+1)}}{\text{minimize}}\ \operatorname{tr}\left(\mathbf{W}_K^{\mathrm{T}}\left(\mathbf{A}_{-k}^{(\ell+1)}+\bar{\sigma}_k^{-2}\mathbf{W}_k\frac{\mathbf{p}_k^{(\ell+1)}\left(\mathbf{p}_k^{(\ell+1)}\right)^{\mathrm{T}}}{\left(\mathbf{p}_k^{(\ell+1)}\right)^{\mathrm{T}}\mathbf{p}_k^{(\ell+1)}}\mathbf{W}_k^{\mathrm{T}}\right)^{-1}\mathbf{W}_K\right)\\
&\text{subject to}\ \left\|\mathbf{p}_k^{(\ell+1)}-\bar{\mathbf{c}}_{k-1}^{(\ell+1)}\right\|^2\le r_{k-1}^2,\\
&\qquad\qquad\left\|\mathbf{p}_k^{(\ell+1)}\right\|^2\ge\rho_k^2,
\end{aligned}$$

where $\mathbf{p}_k^{(\ell+1)}$ is the sought new optimal value of $\mathbf{p}_k$ at iteration $\ell+1$, while $\mathbf{p}_i^{(\ell+1)},\ i=1,\ldots,k-1$, and $\mathbf{p}_j^{(\ell)},\ j=k+1,\ldots,K$, are the remaining optimization variables, which considered fixed during this step, and were computed sequentially during the previous iterations. Furthermore, $\bar{\mathbf{c}}_{k-1}^{(\ell+1)}=\mathbf{p}_{k-1}^{(\ell+1)}+\mathbf{c}_{k-1}$, and $\mathbf{A}_{-k}^{(\ell+1)}$ is defined as follows:

$$\begin{aligned}
\mathbf{A}_{-k}^{(\ell+1)}&=\mathbf{A}+\sum_{i=1}^{k-1}\bar{\sigma}_i^{-2}\mathbf{W}_i\frac{\mathbf{p}_i^{(\ell+1)}\left(\mathbf{p}_i^{(\ell+1)}\right)^{\mathrm{T}}}{\left(\mathbf{p}_i^{(\ell+1)}\right)^{\mathrm{T}}\mathbf{p}_i^{(\ell+1)}}\mathbf{W}_i^{\mathrm{T}}+\sum_{j=k+1}^{K}\bar{\sigma}_j^{-2}\mathbf{W}_j\frac{\mathbf{p}_j^{(\ell)}\left(\mathbf{p}_j^{(\ell)}\right)^{\mathrm{T}}}{\left(\mathbf{p}_j^{(\ell)}\right)^{\mathrm{T}}\mathbf{p}_j^{(\ell)}}\mathbf{W}_j^{\mathrm{T}}\\
&=\mathbf{A}_{-(k-1)}^{(\ell+1)}+\bar{\sigma}_{k-1}^{-2}\mathbf{W}_{k-1}\frac{\mathbf{p}_{k-1}^{(\ell+1)}\left(\mathbf{p}_{k-1}^{(\ell+1)}\right)^{\mathrm{T}}}{\left(\mathbf{p}_{k-1}^{(\ell+1)}\right)^{\mathrm{T}}\mathbf{p}_{k-1}^{(\ell+1)}}\mathbf{W}_{k-1}^{\mathrm{T}}-\bar{\sigma}_k^{-2}\mathbf{W}_k\frac{\mathbf{p}_k^{(\ell)}\left(\mathbf{p}_k^{(\ell)}\right)^{\mathrm{T}}}{\left(\mathbf{p}_k^{(\ell)}\right)^{\mathrm{T}}\mathbf{p}_k^{(\ell)}}\mathbf{W}_k^{\mathrm{T}}.
\end{aligned}\tag{4.43}$$

The optimization process in the above Gauss-Seidel relaxation (GSR) algorithm (sequentially optimizing over $\mathbf{p}_1,\mathbf{p}_2,\ldots,\mathbf{p}_K$) is repeated until the maximum allowed number of iterations is reached (here set to 4), or the change in the objective function [see (4.15)] is less than 1%, whichever occurs first. Note that since the optimization process for each variable $\mathbf{p}_k$ has complexity of $\mathcal{O}(K)$, the overall computational complexity for GSR per iteration is only quadratic in the number of time steps considered, i.e., $\mathcal{O}(K^2)$. Finally, the algorithmic process for the multi-step-ahead single-sensor single-target active tracking using GSR is outlined in Algorithm 4.

Algorithm 4 Multi-step-ahead single-sensor single-target active tracking

Require: $\mathbf{p}_k = \bar{\mathbf{c}}_0,\ k = 1, \ldots, K$.

Ensure: $\mathbf{p}_k = \mathbf{p}_k^{(\ell+1)},\ k = 1, \ldots, K$ {Minimize (4.15)}.

repeat

$\mathbf{A}_{-0}^{(\ell+1)} = \mathbf{A}_{-K}^{(\ell)}$.

for $k = 1$ to K **do**

Calculate $\mathbf{A}_{-k}^{(\ell+1)}$ from $\mathbf{A}_{-(k-1)}^{(\ell+1)}$ [see (4.43)].

Determine $\mathbf{p}_k^{(\ell+1)}$ from Algorithm 3.

end for

until maximum number of iterations is reached or change in the objective function [see (4.15)] is less than 1%.

4.5 Simulation results

We have conducted extensive simulations to evaluate and compare the performance of the multi-step-ahead motion strategy based on GSR to the following two methods:

• *Grid-Based Exhaustive Search* (GBES). In this case, we discretize the feasible set and perform an exhaustive search over all possible combinations of $\mathbf{p}_k,\ k = 1, \ldots, K$, to find the one that minimizes $f(\mathbf{p}_1, \ldots, \mathbf{p}_K)$ [see (4.15)]. Ideally, GBES should return the global optimal solution and it could be used as a benchmark for evaluating GSR, if the grid size is sufficiently small. However, this is difficult to guarantee in practice since its computational complexity is exponential in the number of time steps considered. Hence, implementing GBES becomes prohibitive when the number of time steps K increases and/or the size of the grid cells decreases. In our simulations, we are only able to perform GBES for $K \leq 3$.

• *One-step-ahead Motion Strategy* (OSA). Here, we implement the one-step-ahead motion strategy to determine the best sensing location at the next time step. Note that its globally optimal solution can be computed by following the strategy outlined in Section 3.5.

4.5.1 Simulation setup

Similar to the simulation setup in the multi-sensor case, we adopt a zero-acceleration target motion model

$$\dot{\mathbf{x}}_T(t) = \mathbf{F}\ \mathbf{x}_T(t) + \mathbf{G}\ \mathbf{w}(t), \tag{4.44}$$

where $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{x}_T(t)$ are defined in (3.83), and $\mathbf{w}(t) = [w_x(t)\ w_y(t)]^{\mathrm{T}}$ is a zero-mean white Gaussian noise vector with covariance $\mathbb{E}\left[\mathbf{w}(t)\mathbf{w}^{\mathrm{T}}(t')\right] = q\mathbf{I}_2\delta(t-t')$. Note that by modifying the scalar q, we have the ability to control the accuracy of the target's stochastic motion model (4.44). In our implementation, we discretize the continuous-time system model [see (4.44)] with time step $\delta t = 0.1$ sec.

The initial true state of the target is $\mathbf{x}_T(0) = [0\ 0\ -8\ 6]^{\mathrm{T}}$. The initial estimate for the target's state is $\hat{\mathbf{x}}_T(0) = \hat{\mathbf{x}}_T(0|0) = [2\ -2\ 0\ 0]^{\mathrm{T}}$. At the beginning of the simulations, the sensor is randomly placed within a circle of radius 5 m, which is at a distance of about 20 m from the target's initial position. The maximum speed for the sensor is set to 12 m/sec, i.e., the largest distance that the sensor can travel during any time step is 1.2 m. The minimum distance between the target and the sensor is set to 0.2 m. The standard deviation of the sensor's distance-measurement noise is set to $\sigma_k = 0.2$ m ($k = 1, \ldots, K$). The duration of the simulations is 15 seconds (i.e., 150 time steps). At every time step, we calculate the sensor's trajectory over one (OSA), or multiple (GBES and GSR) time steps.

4.5.2 Target tracking over 3 time steps ahead

We first investigate active target tracking over a sliding window of $K = 3$ time steps ahead. The scalar q is set to 10^{-6} m^2/sec^5.

To illustrate the importance of the measurement noise variance inflation (see (4.12) in Section 4.3.3), we first show in Figure 4.3 the time evolution of the trace of the target's position posterior covariance, averaged over 50 Monte-Carlo trials, where we do not modify σ_k^2 (i.e., setting $\bar{\sigma}_k^2 = \sigma_k^2 = 0.04$ m^2, $k = 1, \ldots, K$). As evident from Figure 4.3, after 50 time steps (i.e., 5 seconds), the performance of GSR is comparable to that of GBES and outperforms OSA. However, GSR performs significantly worse than GBES and OSA, during the first 50 time steps. The target's position uncertainty even increases

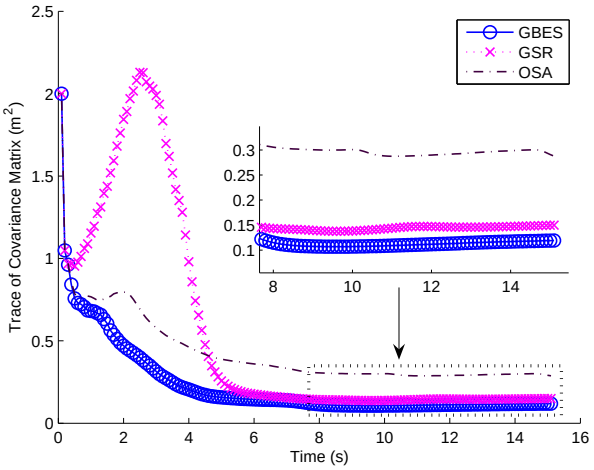


Figure 4.3: [Three time steps ahead; Monte Carlo simulations without measurement noise variance inflation] Average trace of the target's position posterior covariance matrix. Comparison between GBES, GSR, and OSA.

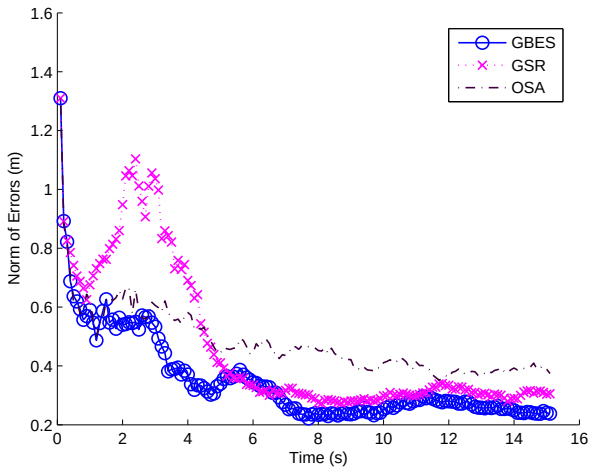


Figure 4.4: [Three time steps ahead; Monte Carlo simulations without measurement noise variance inflation] Average 2-norm of the error of the target's position posterior estimates. Comparison between GBES, GSR, and OSA.

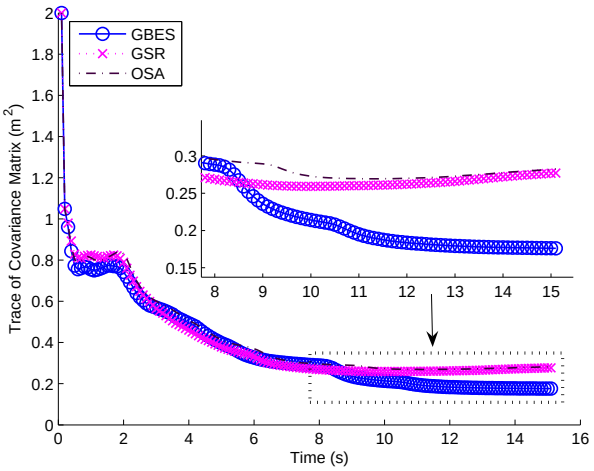


Figure 4.5: [Three time steps ahead; Monte Carlo simulations with measurement noise variance inflation] Average trace of the target's position posterior covariance matrix. Comparison between GBES, GSR, and OSA.

for the first 30 time steps when employing GSR. Furthermore, Figure 4.4 demonstrates the large estimation error between the target's true position and its posterior estimate during the first 30 time steps, when the sensor follows the trajectory generated by GSR.

Figures 4.5–4.6 depict the trace of the target's position posterior covariance matrix, and the 2-norm of the actual error between the target's position estimate and its true value, averaged over 50 Monte-Carlo trials, when the term $\bar{\sigma}_k^2,\ k = 1, \ldots, K$, is modified based on (4.12). As evident, the estimates produced by GSR are significantly improved as compared to the previous case (see Figures 4.3–4.4). More specifically, the performance of GSR is comparable to both GBES and OSA for the first 50 time steps (see Figure 4.5). In addition, GSR achieves the highest tracking accuracy among three methods over the first 60 time steps (see Figure 4.6). These results demonstrate the effectiveness of the proposed rule [see (4.12)] for the measurement noise variance inflation, which appropriately reflects the uncertainty in the predicted target's state estimate $\hat{\mathbf{x}}_T(k),\ k = 1, \ldots, K$.

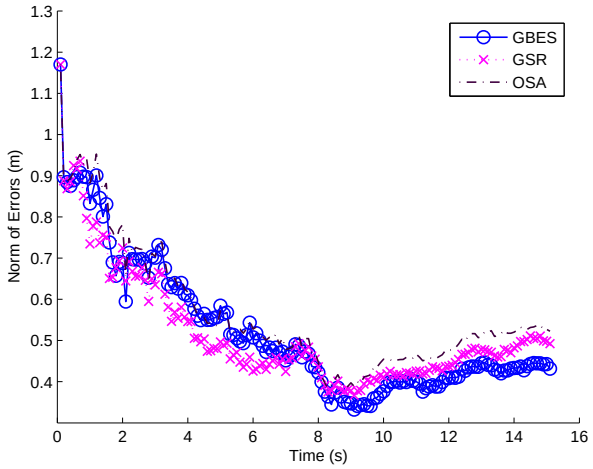


Figure 4.6: [Three time steps ahead; Monte Carlo simulations with measurement noise variance inflation] Average 2-norm of the error of the target's position posterior estimates. Comparison between GBES, GSR, and OSA.

4.5.3 Target tracking over 5 time steps ahead

We hereafter examine the performance of the GSR motion strategy for generating the sensor's optimal trajectory over $K = 5$ time steps ahead. Similar to the case of $K = 3$, the scalar q is set to 10^{-6} $\mathrm{m}^2/\mathrm{sec}^5$. Note that due to its exponential computational complexity, we are unable to implement GBES for $K = 5$.

The trace of the target's position posterior covariance matrix averaged over 50 Monte-Carlo trials is shown in Figure 4.7, and the 2-norm of the actual error between the target's position estimate and its true value is plotted in Figure 4.8. As evident, although GSR performs slightly worse as compared to OSA during the first 10 time steps, it achieves higher tracking accuracy over OSA for the remaining time steps. In other words, compared to OSA, GSR sacrifices its tracking performance at the beginning, so as to attain better tracking accuracy over a longer time horizon.

Figures 4.9(a)–4.9(b) illustrate the actual and estimated trajectories of the target, along with the trajectory of the sensor, during the first 50 time steps, when employing as motion strategy GSR and OSA, respectively, in a single simulation. As evident, the target's position uncertainty when employing GSR is significantly smaller than that

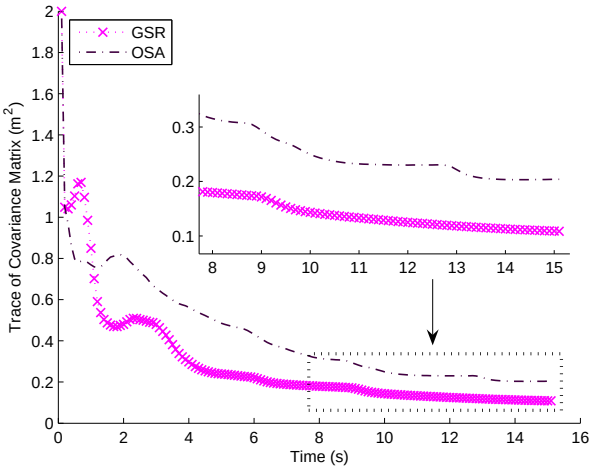


Figure 4.7: [Five time steps ahead; Monte Carlo simulations with measurement noise variance inflation] Average trace of the target's position posterior covariance matrix. Comparison between GSR and OSA.

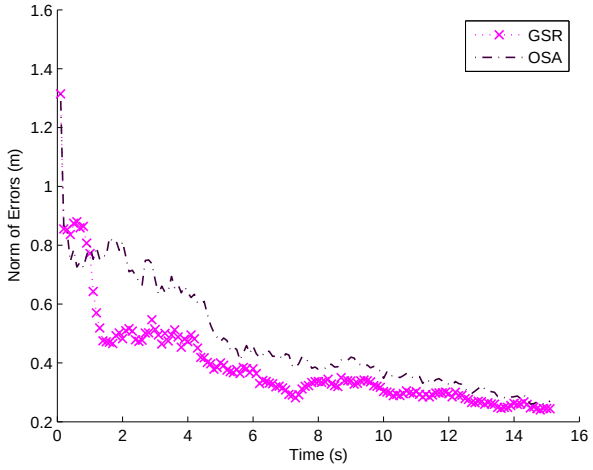


Figure 4.8: [Five time steps ahead; Monte Carlo simulations with measurement noise variance inflation] Average 2-norm of the error of the target's position posterior estimates. Comparison between GSR and OSA.

Table 4.1: $[q = 1]$ Target's position RMS error (m)

K	GBES	GSR	OSA
2	1.7219	1.7128	1.7117
3	1.7929	1.8173	1.7845
4	N/A	1.8250	1.8432
5	N/A	3.1239	1.8859
6	N/A	1.8014	1.7665
7	N/A	2.0377	1.9742
8	N/A	1.8979	1.7694

of OSA after the first 10 time steps. Furthermore, the EKF estimates of the target's position when employing the GSR motion strategy are consistent.

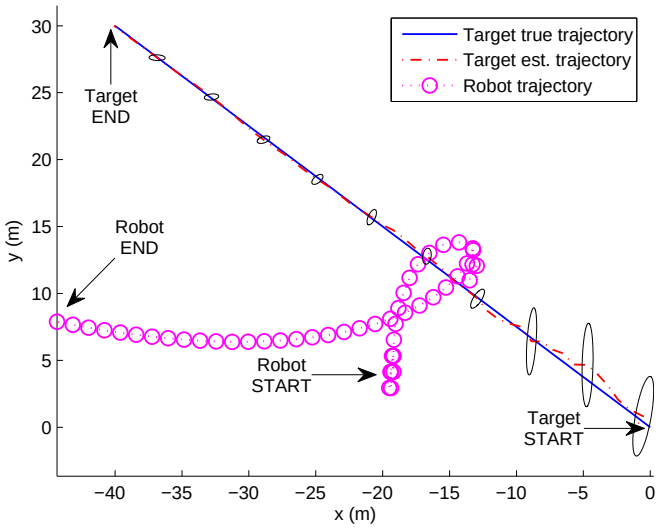
4.5.4 Root mean square error

We have also evaluated the performance of GSR under different levels of the process noise variance. The metric used to evaluate the tracking performance is the root mean square (RMS) error of the target's position estimates. More specifically, the RMS error averaged over all $m = 150$ time steps and $n = 50$ Monte Carlo runs is defined as

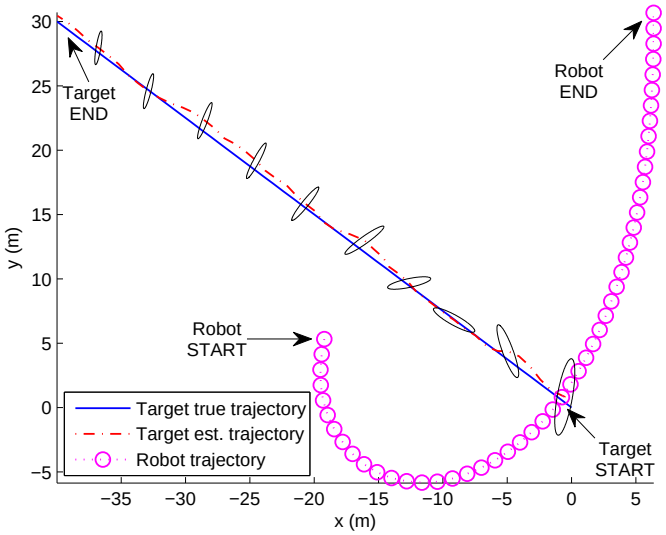
$$\text{RMS error} = \frac{1}{m}\sum_{i=1}^{m}\sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(\mathbf{p}_T^{(j)}(i) - \hat{\mathbf{p}}_T^{(j)}(i|i)\right)^{\mathrm{T}}\left(\mathbf{p}_T^{(j)}(i) - \hat{\mathbf{p}}_T^{(j)}(i|i)\right)},$$

where $\mathbf{p}_T^{(j)}(i)$ and $\hat{\mathbf{p}}_T^{(j)}(i|i)$ are the target's true position and its posterior estimate, respectively, at time-step i for the j-th Monte Carlo run.

Table 4.1 summarizes the RMS errors for GBES, GSR, and OSA, by varying K from 2 to 8. As mentioned previously, GBES can only be implemented for $K \leq 3$. As evident from Table 4.1, for large process noise variance (i.e., $q = 1$), the OSA motion strategy in most cases outperforms the multi-step-ahead motion strategies (i.e., GBES and GSR), except for $K = 4$. This is expected since we are unable to reliably predict the target's motion for longer time periods when the process-noise variance is large [see Figure 4.1(a)]. In such case, the OSA motion strategy is preferred.



(a)



(b)

Figure 4.9: [Five time steps ahead; single simulation with measurement noise variance inflation] The trajectory of the sensor, and the actual and estimated trajectories of the target, when employing as motion strategy (a) GSR and (b) OSA. The ellipses denote the 3σ bounds for the target's position uncertainty at the corresponding time steps.

Table 4.2: $[q = 10^{-6}]$ Target's position RMS error (m)

K	GBES	GSR	OSA
2	0.6162	0.6292	0.6137
3	0.7201	0.6980	0.7694
4	N/A	0.5737	0.6801
5	N/A	0.4856	0.5998
6	N/A	0.6096	0.7584
7	N/A	0.5592	0.6789
8	N/A	0.5828	0.6319

However, by reducing q from 1 to 10^{-6}, and hence we are able to reliably predict the target's state over multiple time steps [see Figure 4.1(b)], then GSR outperforms OSA in most cases except for $K = 2$, as demonstrated in Table 4.2. In conclusion, the results in Table 4.2 confirm that when the process-noise variance is small, generating the optimal trajectory over multiple time steps can achieve higher tracking accuracy as compared to determining the next best sensing location in the one-step-ahead motion strategy.

4.6 Summary

In this chapter, we have studied the problem of optimal trajectory generation over multiple time steps for a single sensor, under mobility constraints, tracking a single target using distance-only observations. In our problem formulation, we explicitly take into account the increasing uncertainty of the predicted target's state estimates by appropriately inflating the measurement-noise variance at the future time steps. In order to provide an efficient, real-time solution for the multi-step-ahead problem, we have relaxed the original non-convex optimization problem and proposed an iterative algorithm, termed nonlinear Gauss-Seidel relaxation (GSR), that leverages the closed-form solution of the single-step optimization and computes an approximate solution to the original multi-step-ahead problem with computational complexity only quadratic in

the number of time steps considered. We have tested the GSR algorithm in Monte-Carlo simulations, and shown that it achieves higher tracking accuracy as compared to that of the one-step-ahead motion strategy when the process-noise variance is small.

Chapter 5

Complexity Reduction in Cooperative Localization (CL)

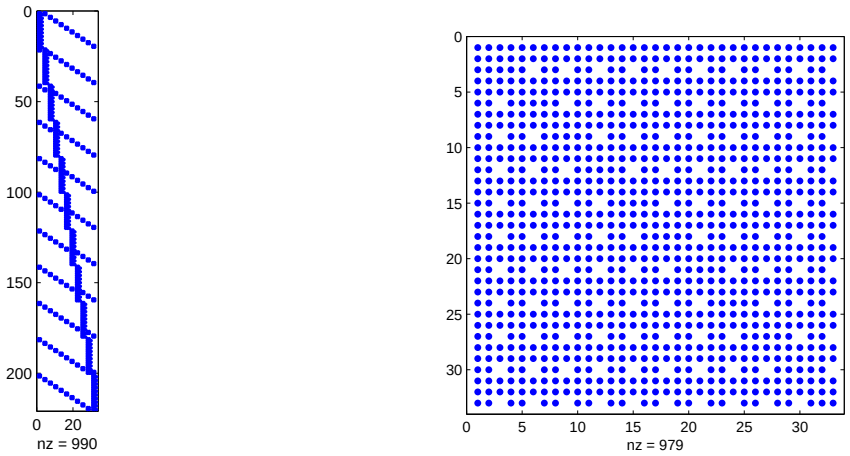
5.1 Introduction

Multi-robot teams (sensor networks) have recently attracted significant interest in the research community because of their robustness, efficiency, versatility, and potential applications, such as environmental monitoring [78], surveillance [79], human-robot interaction [81], and target tracking [85]. Regardless of the application, every robot in the team must be able to accurately localize itself in an unknown environment to ensure successful execution of its tasks. While each robot can independently estimate its own pose (position and orientation) by integrating its linear and rotational velocities, e.g., from wheel encoders [106], the uncertainty of the estimates generated using this technique (dead-reckoning) increases fast, and eventually renders them unreliable. Although one can overcome this limitation by equipping every robot with absolute positioning sensors, such as the Global Positioning System (GPS) receivers, GPS signals are unreliable in urban environments and unavailable in space and underwater. On the other hand, by performing cooperative localization (CL), where communicating robots use relative measurements (distance, bearing, and orientation) to jointly estimate the robots' poses, the accuracy of the robot pose estimates can significantly improve [107], even in the absence of GPS.

As shown in [106], in CL, each robot can process its own proprioceptive measurements independently and distributively. However, since CL involves joint-state estimation, the processing of exteroceptive measurements (i.e., robot-to-robot measurements) requires the robots to communicate with each other and update the covariance matrix corresponding to all pose estimates in a centralized fashion. Using the extended Kalman filter (EKF) framework, the time complexity for processing each exteroceptive measurement is $\mathcal{O}(N^2)$ (N being the number of robots) [106]. Since at every time step the maximum possible number of exteroceptive measurements is $(N-1)N$, the overall time complexity for updating the estimates' covariance, in the worst case, becomes $\mathcal{O}(N^4)$ per time step. As N grows, this high (time) complexity may prohibit real-time performance.

In this chapter, we investigate the computational complexity (i.e., time complexity and space complexity) of the centralized EKF-based CL algorithm, considering the most challenging case where the total number of relative measurements per time step is $(N-1)N$. Since the relative observations only involve pairs of robots, the measurement (Jacobian) matrix $\mathbf{H}$ arising in CL is inherently sparse (see Section 5.3.2). Figure 5.1(a) illustrates the sparsity of $\mathbf{H}$ for $N = 11$. By exploiting this sparsity, we are able to show that the time complexity of covariance update can be reduced to $\mathcal{O}(N^3)$ when employing the Information filter (see Section 5.3.3). To further improve the numerical stability of the Information filter, we present an EKF update approach that compresses all relative measurements through QR factorization (also called QR decomposition) of $\mathbf{H}$ (see Section 5.3.4). However, the traditional Householder QR algorithm (also called the Standard Householder QR) has the drawback of introducing non-zero fill-ins and thus destroying the sparsity of $\mathbf{H}$ at every iteration. As a result, the QR factorization of $\mathbf{H}$ when using the Standard Householder QR requires time complexity of $\mathcal{O}(N^4)$ and space complexity of $\mathcal{O}(N^3)$ (see Section 5.4). In this work, we develop the Modified Householder QR algorithm (see Section 5.5), which fully exploits the sparsity of $\mathbf{H}$ and achieves the QR decomposition of $\mathbf{H}$ with time complexity and space complexity of $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2)$, respectively. As a result, the overall computational complexity of the EKF-based CL, through QR factorization, is reduced from $\mathcal{O}(N^4)$ to $\mathcal{O}(N^3)$ for time complexity and from $\mathcal{O}(N^3)$ to $\mathcal{O}(N^2)$ for space complexity.

Following a brief review of related work in Section 5.2, we present the formulation of



(a) Measurement matrix $\mathbf{H}$ (b) $\mathbf{H}^{\mathrm{T}}\mathbf{H}$

Figure 5.1: Sparsity patterns of (a) measurement matrix $\mathbf{H}$ and (b) $\mathbf{H}^{\mathrm{T}}\mathbf{H}$ for $N = 11$, where the blue dots denote the non-zero elements. Note that due to the structure of $\mathbf{H}$, the square matrix $\mathbf{H}^{\mathrm{T}}\mathbf{H}$ is not necessarily sparse, even though $\mathbf{H}$ is sparse. Therefore, the upper triangular matrix $\mathbf{R}$ computed from the QR decomposition of $\mathbf{H}$ is generally dense.

the EKF-based CL, as well as the EKF update approach based on QR factorization of $\mathbf{H}$ in Section 5.3. In Section 5.4, we describe the Standard Householder QR algorithm, and analyze its computational complexity when factorizing $\mathbf{H}$. We then present the Modified Householder QR in Section 5.5, followed by its time and space complexity analysis. Simulation results are presented in Section 5.6, followed by the summary in Section 5.7.

5.2 Literature review

Multi-robot cooperative localization (CL) has received considerable attention in the literature (e.g., [108, 109, 110, 111]). Various system architectures, such as centralized [106, 112] or distributed [113, 114, 115, 116], have been proposed for CL. These system architectures use various estimation algorithms, such as the EKF [106], the maximum-likelihood estimator (MLE) [112], the maximum-a-posteriori estimator (MAP) [116], and particle filters [109]. In this section, we focus our discussion on centralized or distributed approaches that fuse data using MLE/MAP (see Section 5.2.1) or EKF (see Section 5.2.2). The latter case is the main focus of our work. In what follows, we denote as K the total number of time steps considered in MLE/MAP, N the number of robots in the team, and examine the worst-case computational complexity where the total number of robot-to-robot measurements per time step is $(N-1)N$.

5.2.1 MLE or MAP-based estimators

Howard *et al.* [112] presented a centralized MLE-based algorithm to address the CL problem. The solution of the underlying non-convex optimization problem, formulated by maximizing the conditional probability density function (pdf), is computed iteratively using the conjugate gradient (CG) algorithm. Since the cost at each iteration in CG is of $\mathcal{O}(KN^2)$, the worst-case time complexity of the proposed algorithm is $\mathcal{O}(K^2N^3)$.

Contrary to [112], where all measurements are transmitted and processed at a fusion center, the same authors in [115] proposed to solve the MLE in a distributed fashion. The original non-convex optimization problem is decomposed into N sub-problems, one for each robot. In particular, the ith robot (i.e., robot-i) minimizes only a part of the

cost function that involves terms corresponding to its own proprioceptive measurements, as well as the relative measurements between robot-i and other robots. The optimization process in this case approximates the other robots' poses as constants. However, there exists no proof that this approximate approach guarantees convergence to a local minimum. Additionally, the computational complexity of the proposed algorithm is not addressed in the paper.

Similar to [112], Dellaert *et al.* [117] addressed the MAP-based centralized CL by incorporating prior information about the initial poses of the robots. Contrary to [112], the resulting non-convex optimization problem is solved iteratively by the Levenberg-Marquardt (LM) algorithm. The solution of the system of linear equations at each iteration of the LM method is determined through a sparse QR solver. Although some insights were provided about the selection of the initial estimates, the algorithm does not guarantee convergence to the global optimum. Furthermore, a shortcoming of the sparse QR solver is that its performance is case-dependent, and no information about its worst-case computational complexity is provided.

Recently, Nerurkar *et al.* [116] introduced a fully distributed CG algorithm to address the MAP-based CL. Similar to [117], the resulting non-convex optimization problem is solved iteratively using the LM algorithm. In contrast to [117], the incremental solution at each iteration of the LM method is distributively computed using the CG algorithm. The time complexity per robot and iteration of the LM method is of $\mathcal{O}((KN)^2)$, hence the overall processing cost for all robots per LM iteration is of $\mathcal{O}(K^2N^3)$.

In summary, the worst-case time complexity of the MLE-based CL, as shown in [112], is quadratic in K and cubic in N. On the other hand, the robots' state estimates, computed distributively in [115], are only approximate. In [117], the worst-case computational complexity of the sparse QR solver is not provided. Furthermore, a limitation of [116] is that it requires synchronous communication among the robot team. Finally, the main drawback of the aforementioned MLE or MAP-based estimators is that they are all formulated based upon the Information filter, which squares the condition number of the measurement matrix and renders the estimators numerically less robust and stable.

5.2.2 EKF-based estimators

In [106], Roumeliotis and Bekey showed that within the EKF framework, each robot can independently propagate its own state and covariance in a fully distributed fashion. However, during the update step, the observing robot needs to broadcast its relative measurements to the rest of the robots in the team, and update the covariance matrix corresponding to all pose estimates in a centralized fashion. In this approach, the time complexity for processing each exteroceptive measurement is $\mathcal{O}(N^2)$. Therefore, in the worst case, the overall time complexity for sequentially updating state and covariance becomes $\mathcal{O}(N^4)$ per time step. In order to reduce the computational requirements of the EKF-based CL, several approximate algorithms have been proposed.

Panzieri *et al.* [118] presented a fully-decentralized algorithm based on the Interlaced EKF [119], where each robot only processes relative measurements taken by itself to update its own pose estimate, while treating the other robots' poses as deterministic parameters. The time complexity per robot and time step is $\mathcal{O}(N)$ [hence the overall time complexity per time step is $\mathcal{O}(N^2)$] when relative measurements are processed sequentially. Similarly to [118], Karam *et al.* [120] proposed a distributed EKF-based method for CL. However, contrary to [118], here the robots are restricted to exchange their state estimates with others within communication range. The time complexity per robot and time step is $\mathcal{O}(N^2)$, resulting in an overall time complexity $\mathcal{O}(N^3)$ per time step.

Martinelli [121] developed a distributed approach for CL that uses hierarchical EKF filters to estimate the robots' poses. In particular, the N robots in a team are divided into L groups, each comprising M robots ($N = LM$). Every group contains a leader who processes all the relative measurements between any two robots in that group and only updates the pose estimates of the robots belonging to its group. A team leader is in charge of processing all observations between any two robots from different groups and only updates the pose estimates of the L group leaders. The time complexity per time step for each group leader and the team leader are $\mathcal{O}(M^4)$ and $\mathcal{O}(N(N-M)L^2)$, respectively.

The main drawback of the aforementioned approximate algorithms (see [118, 120, 121]) is that in order to reduce the computational complexity of the EKF-based CL, these approaches ignore cross-correlations amongst robots, which often leads to overly

optimistic and inconsistent estimates. In this work, we present an algorithm that reduces the time complexity of the EKF-based CL to $\mathcal{O}(N^3)$, without introducing any approximations, but, instead, by taking advantage of the specific sparse structure of the measurement matrix.

5.3 Problem formulation

Consider a group of N mobile robots performing CL in 2-D by processing relative distance and bearing measurements. In this work, we study the case of global localization, i.e., the pose of each robot is expressed with respect to a fixed (global) frame of reference.

The pose of the ith robot (or robot-i) at time-step k is denoted as $\mathbf{x}_k^i = [(\mathbf{p}_k^i)^{\mathrm{T}}\ \varphi_k^i]^{\mathrm{T}}$, where $\mathbf{p}_k^i = [x_k^i\ y_k^i]^{\mathrm{T}}$ and φ_k^i represent the global position and orientation of robot-i at time-step k, respectively. The state vector for the robot team at time-step k is defined as $\mathbf{x}_k = [(\mathbf{x}_k^1)^{\mathrm{T}}\ (\mathbf{x}_k^2)^{\mathrm{T}}\ \ldots\ (\mathbf{x}_k^N)^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{3N}$. We assume that each robot is equipped with proprioceptive sensors (e.g., wheel encoders), which measure its linear and rotational velocities, as well as exteroceptive sensors (e.g., laser scanners), which can detect, identify, and measure the relative distance and bearing to other robots.

5.3.1 State propagation

The discrete-time state propagation equation for robot-i from time-step $k-1$ to k is

$$\mathbf{x}_k^i = \mathbf{f}_{k-1}^i(\mathbf{x}_{k-1}^i, \mathbf{u}_{k-1}^i, \mathbf{w}_{k-1}^i),\ i = 1, \ldots, N,$$

where the control input $\mathbf{u}_{k-1}^i = [v_{k-1}^i\ \omega_{k-1}^i]^{\mathrm{T}}$, consisting of the linear velocity measurement v_{k-1}^i and rotational velocity measurement ω_{k-1}^i recorded by the robot-i's odometric sensors, is corrupted by zero-mean, white Gaussian process noise $\mathbf{w}_{k-1}^i = [w_{k-1,v}^i\ w_{k-1,\omega}^i]^{\mathrm{T}}$ with covariance $\mathbf{C}_{\mathbf{w}}^i$.

In this work, we employ the extended Kalman filter (EKF) for recursively estimating the robot's pose $\mathbf{x}_k^i,\ i = 1, \ldots, N$. Thus the estimate of robot-i's pose is propagated by

$$\hat{\mathbf{x}}_{k|k-1}^i = \mathbf{f}_{k-1}^i(\hat{\mathbf{x}}_{k-1|k-1}^i, \mathbf{u}_{k-1}^i, \mathbf{0}_{2\times 1}),\ i = 1, \ldots, N,$$

where $\hat{\mathbf{x}}_{\ell|j}$ is the state estimate at time-step ℓ, after measurements up to time-step j have been processed.

The covariance matrix corresponding to the state estimate $\hat{\mathbf{x}}_{k|k-1}$ is propagated as

$$\mathbf{P}_{k|k-1} = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}_{k-1}^{\mathrm{T}} + \mathbf{G}_{k-1}\mathbf{C}_{\mathbf{w}}\mathbf{G}_{k-1}^{\mathrm{T}}, \tag{5.1}$$

where $\mathbf{\Phi}_{k-1} = \mathrm{diag}(\mathbf{\Phi}_{k-1}^{1}, \ldots, \mathbf{\Phi}_{k-1}^{N})$ and $\mathbf{G}_{k-1} = \mathrm{diag}(\mathbf{G}_{k-1}^{1}, \ldots, \mathbf{G}_{k-1}^{N})$, with $\mathbf{\Phi}_{k-1}^{i} = \boldsymbol{\nabla}_{\mathbf{x}_{k-1}^{i}}\mathbf{f}_{k-1}^{i}$ and $\mathbf{G}_{k-1}^{i} = \boldsymbol{\nabla}_{\mathbf{w}_{k-1}^{i}}\mathbf{f}_{k-1}^{i},\ i = 1, \ldots, N$. The overall process noise covariance is $\mathbf{C}_{\mathbf{w}} = \mathrm{diag}(\mathbf{C}_{\mathbf{w}}^{1}, \ldots, \mathbf{C}_{\mathbf{w}}^{N})$.

Note that due to the block diagonal structures of $\mathbf{\Phi}_{k-1}$ and $\mathbf{G}_{k-1}$, the overall time and space complexity of (5.1) are $\mathcal{O}(N^2)$ [106].

5.3.2 Measurement model

At time-step k, the relative distance and bearing observations recorded by the exteroceptive sensors of robot-i measuring robot-j $(1 \leq i \neq j \leq N)$ are given by

$$\mathbf{z}_{k}^{i,j} = \mathbf{h}_{k}^{i,j}(\mathbf{x}_{k}^{i}, \mathbf{x}_{k}^{j}) + \mathbf{n}_{k}^{i,j}, \tag{5.2}$$

where $\mathbf{h}_{k}^{i,j}(\mathbf{x}_{k}^{i}, \mathbf{x}_{k}^{j}) = [d_{k}^{i,j}\ \theta_{k}^{i,j}]^{\mathrm{T}}$, with

$$d_{k}^{i,j} = \|\mathbf{p}_{k}^{j} - \mathbf{p}_{k}^{i}\|_2, \quad \text{and} \quad \theta_{k}^{i,j} = \arctan\left(\frac{y_{k}^{j} - y_{k}^{i}}{x_{k}^{j} - x_{k}^{i}}\right) - \varphi_{k}^{i}$$

denoting the true distance and bearing from robot-i to robot-j at time-step k, respectively. $\mathbf{n}_{k}^{i,j} = [n_{k,d}^{i,j}\ n_{k,\theta}^{i,j}]^{\mathrm{T}}$ is zero-mean white Gaussian measurement noise with covariance $\mathbf{C}_{\mathbf{n}}^{i,j}$. Without loss of generality, we assume $\mathbf{C}_{\mathbf{n}}^{i,j} = \mathbf{I}_2$ $(1 \leq i \neq j \leq N)$ in the remainder of the chapter.

The measurement-error equation for $\mathbf{z}_{k}^{i,j}$, obtained by linearizing (5.2) around the current best state estimate $\hat{\mathbf{x}}_{k|k-1}$, is

$$\begin{aligned} \tilde{\mathbf{z}}_{k|k-1}^{i,j} &= \mathbf{z}_{k}^{i,j} - \mathbf{h}_{k}^{i,j}(\hat{\mathbf{x}}_{k|k-1}^{i}, \hat{\mathbf{x}}_{k|k-1}^{j}) \\ &\approx \mathbf{\Psi}_{k}^{i,j}\,\tilde{\mathbf{x}}_{k|k-1}^{i} + \mathbf{\Upsilon}_{k}^{i,j}\,\tilde{\mathbf{x}}_{k|k-1}^{j} + \mathbf{n}_{k}^{i,j} = \mathbf{H}_{k}^{i,j}\,\tilde{\mathbf{x}}_{k|k-1} + \mathbf{n}_{k}^{i,j}, \end{aligned} \tag{5.3}$$

where

$$\boldsymbol{\Psi}_k^{i,j} = \boldsymbol{\nabla}_{\mathbf{x}_k^i} \mathbf{h}_k^{i,j} = \begin{bmatrix} -\frac{(\hat{\mathbf{p}}_{k|k-1}^j - \hat{\mathbf{p}}_{k|k-1}^i)^{\mathrm{T}}}{\|\hat{\mathbf{p}}_{k|k-1}^j - \hat{\mathbf{p}}_{k|k-1}^i\|_2} & 0 \\ -\frac{(\hat{\mathbf{p}}_{k|k-1}^j - \hat{\mathbf{p}}_{k|k-1}^i)^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{\|\hat{\mathbf{p}}_{k|k-1}^j - \hat{\mathbf{p}}_{k|k-1}^i\|_2^2} & -1 \end{bmatrix}, \tag{5.4}$$

$$\boldsymbol{\Upsilon}_k^{i,j} = \boldsymbol{\nabla}_{\mathbf{x}_k^j} \mathbf{h}_k^{i,j} = \begin{bmatrix} \frac{(\hat{\mathbf{p}}_{k|k-1}^j - \hat{\mathbf{p}}_{k|k-1}^i)^{\mathrm{T}}}{\|\hat{\mathbf{p}}_{k|k-1}^j - \hat{\mathbf{p}}_{k|k-1}^i\|_2} & 0 \\ \frac{(\hat{\mathbf{p}}_{k|k-1}^j - \hat{\mathbf{p}}_{k|k-1}^i)^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{\|\hat{\mathbf{p}}_{k|k-1}^j - \hat{\mathbf{p}}_{k|k-1}^i\|_2^2} & 0 \end{bmatrix}, \tag{5.5}$$

and $\mathbf{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$. Furthermore, note that the measurement matrix $\mathbf{H}_k^{i,j}$ (of dimensions $2 \times 3N$) has a sparse structure (without loss of generality, we assume $i < j$), i.e.,

$$\mathbf{H}_k^{i,j} = \begin{bmatrix} \mathbf{0}_{2\times 3(i-1)} & \boldsymbol{\Psi}_k^{i,j} & \mathbf{0}_{2\times 3(j-i-1)} & \boldsymbol{\Upsilon}_k^{i,j} & \mathbf{0}_{2\times 3(N-j)} \end{bmatrix}.$$

In fact, there are at most 9 nonzero elements in $\mathbf{H}_k^{i,j}$ (5 from $\boldsymbol{\Psi}_k^{i,j}$ and 4 from $\boldsymbol{\Upsilon}_k^{i,j}$).

In this chapter, we consider the most challenging, in terms of computational requirements, scenario where the sensing range of the exteroceptive sensors is sufficiently large so that each robot can detect, identify, and measure the relative distance and bearing to the remaining $N-1$ robots at every time step.[1] Thus, the total number of relative measurements per time step is $(N-1)N$.

The measurement-error equation for the robot team, obtained by stacking all the measurement residuals $\tilde{\mathbf{z}}_{k|k-1}^{i,j}$ $(1 \leq i \neq j \leq N)$ [see (5.3)] into a column vector, is

$$\tilde{\mathbf{z}}_{k|k-1} \approx \mathbf{H}_k \tilde{\mathbf{x}}_{k|k-1} + \mathbf{n}_k,$$

where $\tilde{\mathbf{z}}_{k|k-1} = [(\tilde{\mathbf{z}}_{k|k-1}^{1,2})^{\mathrm{T}} \ldots (\tilde{\mathbf{z}}_{k|k-1}^{i,j})^{\mathrm{T}} \ldots (\tilde{\mathbf{z}}_{k|k-1}^{N,N-1})^{\mathrm{T}}]^{\mathrm{T}}$ is the measurement residual error vector of dimension $2(N-1)N$, and $\mathbf{n}_k = [(\mathbf{n}_k^{1,2})^{\mathrm{T}} \ldots (\mathbf{n}_k^{i,j})^{\mathrm{T}} \ldots (\mathbf{n}_k^{N,N-1})^{\mathrm{T}}]^{\mathrm{T}}$ is zero-mean, white Gaussian measurement noise with covariance $\mathbf{C}_{\mathbf{n}} = \mathbf{I}_{2(N-1)N}$.

The overall measurement matrix $\mathbf{H}_k = [(\mathbf{H}_k^{1,2})^{\mathrm{T}} \ldots (\mathbf{H}_k^{i,j})^{\mathrm{T}} \ldots (\mathbf{H}_k^{N,N-1})^{\mathrm{T}}]^{\mathrm{T}}$, whose dimensions are $2(N-1)N \times 3N$, has the following sparse structure

[1] For the general case when the number of measurements is less than $(N-1)N$ due to the limited sensing range of each robot, our proposed method (see Section 5.5) can be readily applied without modifications.

$$\mathbf{H}_k = \begin{bmatrix} \boldsymbol{\Psi}_k^{1,2} & \boldsymbol{\Upsilon}_k^{1,2} & & & & & \\ \boldsymbol{\Upsilon}_k^{2,1} & \boldsymbol{\Psi}_k^{2,1} & & & & & \\ \vdots & & & & & & \\ \boldsymbol{\Psi}_k^{1,N} & & & & & & \boldsymbol{\Upsilon}_k^{1,N} \\ \boldsymbol{\Upsilon}_k^{N,1} & & & & & & \boldsymbol{\Psi}_k^{N,1} \\ & \boldsymbol{\Psi}_k^{2,3} & \boldsymbol{\Upsilon}_k^{2,3} & & & & \\ & \boldsymbol{\Upsilon}_k^{3,2} & \boldsymbol{\Psi}_k^{3,2} & & & & \\ & \vdots & & & & & \\ & \boldsymbol{\Psi}_k^{2,N} & & & & & \boldsymbol{\Upsilon}_k^{2,N} \\ & \boldsymbol{\Upsilon}_k^{N,2} & & & & & \boldsymbol{\Psi}_k^{N,2} \\ & & \vdots & \vdots & \vdots & & \\ & & & & \boldsymbol{\Psi}_k^{N-1,N} & \boldsymbol{\Upsilon}_k^{N-1,N} & \\ & & & & \boldsymbol{\Upsilon}_k^{N,N-1} & \boldsymbol{\Psi}_k^{N,N-1} & \end{bmatrix}. \tag{5.6}$$

Remark 18. *We highlight two important properties of* $\mathbf{H}_k$*. Firstly, due to its sparse structure* [*see* (5.6)]*, each row of* $\mathbf{H}_k$ *has at most 5 nonzero elements* [*see* (5.4) – (5.5)]*. Per column, there exist at most* $4(N-1)$ *non-zeros, e.g., the nonzero elements of the first column of* $\mathbf{H}_k$ *originate from the first columns of the matrices* $\boldsymbol{\Psi}_k^{1,j}$ *and* $\boldsymbol{\Upsilon}_k^{j,1}$*,* $j = 2, \ldots, N$*, where each column of* $\boldsymbol{\Psi}_k^{1,j}$ *(or* $\boldsymbol{\Upsilon}_k^{j,1}$*) has at most 2 nonzero elements* [*see* (5.4) – (5.5)]*. Therefore* $\mathbf{nnz}(\mathbf{H}_k) \sim \mathcal{O}(N^2)$*, where* $\mathbf{nnz}(\mathbf{H}_k)$ *denotes the number of nonzero elements of* $\mathbf{H}_k$*, and the storage of* $\mathbf{H}_k$ *can be efficiently achieved using* $\mathcal{O}(N^2)$ *memory cells. Secondly, since all the exteroceptive measurements are relative observations between each pair of robots, and absolute pose measurements* (*such as GPS*) *are unavailable,* $\mathbf{H}_k$ *is rank deficient. In particular, it has been shown that* $\mathbf{rank}(\mathbf{H}_k) = 3(N-1)$ [122].

5.3.3 State and covariance update

Once all the relative measurements $\mathbf{z}_k^{i,j}$, $1 \leq i \neq j \leq N$, become available, the state estimate and its covariance can be updated as

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{z}}_{k|k-1}, \tag{5.7}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^{\mathrm{T}}, \tag{5.8}$$

where $\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}\mathbf{S}_k^{-1}$ is the Kalman gain and $\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}} + \mathbf{C}_{\mathbf{n}}$ is the measurement residual covariance.

Unfortunately, the time complexity and space complexity of the covariance update using (5.8) are $\mathcal{O}(N^6)$ and $\mathcal{O}(N^4)$, respectively, due to the inversion and storage of a dense matrix $\mathbf{S}_k$ whose dimensions are $2(N-1)N \times 2(N-1)N$. Therefore, the real-time implementation of (5.7)–(5.8) becomes prohibitive as the number of robots, N, increases.

It is possible to avoid the inversion and storage of $\mathbf{S}_k$ by processing the $(N-1)N$ exteroceptive measurements sequentially. In this case, since the time complexity for processing every relative measurement $\mathbf{z}_k^{i,j}$ is $\mathcal{O}(N^2)$, the total time complexity per update step is $\mathcal{O}(N^4)$ [106]. Additionally, the space complexity reduces to $\mathcal{O}(N^2)$. Note, however, that due to the nonlinearity of the measurement model [see (5.2)], the estimates obtained by sequential updates are less accurate than these computed by concurrent updates [see (5.7)–(5.8)].

Another alternative approach for updating the state and covariance is to employ the Information filter [123, Chapter 5, Section 5.6], i.e., for $\mathbf{C_n} = \mathbf{I}_{2(N-1)N}$,

$$\mathbf{P}_{k|k} = \left(\mathbf{P}_{k|k-1}^{-1} + \mathbf{H}_k^{\mathrm{T}}\mathbf{H}_k\right)^{-1}, \tag{5.9}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{P}_{k|k}\mathbf{H}_k^{\mathrm{T}}\tilde{\mathbf{z}}_{k|k-1}. \tag{5.10}$$

In order to analyze the computational complexity of the EKF-based CL when employing the Information filter [see (5.9)–(5.10)], we first show the following lemma, which establishes that computing the dense square matrix $\mathbf{H}_k^{\mathrm{T}}\mathbf{H}_k$ [see Figure 5.1(b)] and the vector $\mathbf{P}_{k|k}\mathbf{H}_k^{\mathrm{T}}\tilde{\mathbf{z}}_{k|k-1}$ only requires $\mathcal{O}(N^2)$ arithmetic operations, due to the sparse structure of $\mathbf{H}_k$.

Lemma 19. *The time complexity of evaluating the square matrix $\mathbf{H}_k^{\mathrm{T}}\mathbf{H}_k$ and the vector $\mathbf{P}_{k|k}\mathbf{H}_k^{\mathrm{T}}\tilde{\mathbf{z}}_{k|k-1}$ is $\mathcal{O}(N^2)$.*

Proof. To proceed, we rewrite $\mathbf{H}_k = \begin{bmatrix}\mathbf{H}_{k_1} & \dots & \mathbf{H}_{k_N}\end{bmatrix}$, where $\mathbf{H}_{k_i}$ is a $2(N-1)N \times 3$ submatrix consisting of the $(3i-2)$th, $(3i-1)$th, and $3i$th columns of $\mathbf{H}_k$, for $i = 1, \dots, N$. Based on this partition, we have $\mathbf{H}_k^{\mathrm{T}}\mathbf{H}_k = \left[\mathbf{H}_{k_i}^{\mathrm{T}}\mathbf{H}_{k_j}\right]_{i,j}$, where $\mathbf{H}_{k_i}^{\mathrm{T}}\mathbf{H}_{k_j}$ $(i, j = 1, \dots, N)$ are 3×3 block matrices.

Next we observe that the sparse structure of $\mathbf{H}_k$ [see (5.6)] implies that (for $1 \leq i \neq$

$j \leq N$)

$$\mathbf{H}_{k_i}^{\mathrm{T}}\mathbf{H}_{k_i} = \sum_{\eta=1,\ \eta\neq i}^{N} \left((\mathbf{\Psi}_k^{i,\eta})^{\mathrm{T}}\mathbf{\Psi}_k^{i,\eta} + (\mathbf{\Upsilon}_k^{\eta,i})^{\mathrm{T}}\mathbf{\Upsilon}_k^{\eta,i}\right), \tag{5.11}$$

$$\mathbf{H}_{k_i}^{\mathrm{T}}\mathbf{H}_{k_j} = (\mathbf{\Psi}_k^{i,j})^{\mathrm{T}}\mathbf{\Upsilon}_k^{i,j} + (\mathbf{\Upsilon}_k^{j,i})^{\mathrm{T}}\mathbf{\Psi}_k^{j,i}. \tag{5.12}$$

Since the right-hand-side (RHS) of (5.11)–(5.12) involves multiplications and additions of constant-dimension matrices (i.e., $\mathbf{\Psi}_k^{i,j}, \mathbf{\Upsilon}_k^{i,j}, \mathbf{\Psi}_k^{j,i}, \mathbf{\Upsilon}_k^{j,i}$), the time complexity of evaluating each term of (5.11) and (5.12) are $\mathcal{O}(N)$ and $\mathcal{O}(1)$, respectively. Therefore the overall time complexity of evaluating $\mathbf{H}_k^{\mathrm{T}}\mathbf{H}_k$ is quadratic in N.

Furthermore, computing the vector $\mathbf{P}_{k|k}\mathbf{H}_k^{\mathrm{T}}\tilde{\mathbf{z}}_{k|k-1}$ has time complexity $\mathcal{O}(N^2)$. To proceed, we first compute the $3N$ dimensional vector $\tilde{\mathbf{y}} = \mathbf{H}_k^{\mathrm{T}}\tilde{\mathbf{z}}_{k|k-1}$. Note that the jth element of $\tilde{\mathbf{y}}$, $\tilde{y}_j$, is the inner product of the vector $\tilde{\mathbf{z}}_{k|k-1}$ and the jth column of $\mathbf{H}_k$. Since each column of $\mathbf{H}_k$ has at most $4(N-1)$ nonzero elements (see Remark 18), the processing requirement of computing $\tilde{y}_j$ is $\mathcal{O}(N)$, despite the fact that $\tilde{\mathbf{z}}_{k|k-1}$ is a $2(N-1)N$ dense vector. Thus, the time complexity of calculating $\tilde{\mathbf{y}}$ is $\mathcal{O}(N^2)$. Next, we compute $\mathbf{P}_{k|k}\tilde{\mathbf{y}}$, a matrix (of dimensions $3N \times 3N$) and vector (of dimension $3N$) multiplication, in $\mathcal{O}(N^2)$ basic operations. Hence, the overall time complexity of computing $\mathbf{P}_{k|k}\mathbf{H}_k^{\mathrm{T}}\tilde{\mathbf{z}}_{k|k-1}$ is $\mathcal{O}(N^2)$. □

Based on Lemma 19, we conclude that the most computationally demanding operations of (5.9)–(5.10) are the inversions of two matrices (i.e., $\mathbf{P}_{k|k-1}$ and $\mathbf{P}_{k|k-1}^{-1}+\mathbf{H}_k^{\mathrm{T}}\mathbf{H}_k$), whose dimensions are linear in N. Thus, the total time complexity of the state and covariance updates using (5.9)–(5.10) is $\mathcal{O}(N^3)$, a significant complexity reduction as compared to the standard EKF updates using (5.7)–(5.8). Additionally, since both $\mathbf{P}_{k|k-1}$ and $\mathbf{H}_k$ require $\mathcal{O}(N^2)$ memory cells (recall that $\mathbf{nnz}(\mathbf{H}_k) \sim \mathcal{O}(N^2)$ and only the non-zero elements of $\mathbf{H}_k$ need to be stored), the space complexity of updating the state and covariance using (5.9)–(5.10) is reduced from $\mathcal{O}(N^4)$ to $\mathcal{O}(N^2)$. Theorem 20 summarizes the above analysis.

Theorem 20. *The EKF update in CL, when employing the Information filter* [*see* (5.9)–(5.10)]*, has time complexity and space complexity of* $\mathcal{O}(N^3)$ *and* $\mathcal{O}(N^2)$*, respectively.*

Unfortunately, (5.9) often suffers from numerical instability, due to $\kappa(\mathbf{H}_k^{\mathrm{T}}\mathbf{H}_k) =$

$\kappa^2(\mathbf{H}_k)$ (where $\kappa(\mathbf{H}_k)$ is the condition number of $\mathbf{H}_k$), which renders the algorithm (5.9)–(5.10) numerically less robust [124].

5.3.4 State and covariance update through QR factorization

An alternative strategy that effectively overcomes the numerical instability of the Information filter [see (5.9)–(5.10)] is to perform an additional preprocessing step and reduce the dimension of the measurements. This is achieved through thin QR factorization with column pivoting of $\mathbf{H}_k$ [124], i.e.,

$$\mathbf{H}_k = \mathbf{Q}_{\mathbf{H}_k}\mathbf{R}_{\mathbf{H}_k}\mathbf{\Pi}_{\mathbf{H}_k}^{\mathrm{T}}, \tag{5.13}$$

where $\mathbf{Q}_{\mathbf{H}_k}$ is a $2(N-1)N\times 3(N-1)$ matrix with orthonormal columns (i.e., $\mathbf{Q}_{\mathbf{H}_k}^{\mathrm{T}}\mathbf{Q}_{\mathbf{H}_k} = \mathbf{I}_{3(N-1)}$), and $\mathbf{R}_{\mathbf{H}_k}$ is a $3(N-1)N \times 3N$ upper triangular matrix with the absolute value of its diagonal elements arranged in decreasing order.[2] Furthermore, $\mathbf{\Pi}_{\mathbf{H}_k}$ is a (column) permutation matrix generated from column-pivoting techniques, satisfying $\mathbf{\Pi}_{\mathbf{H}_k}\mathbf{\Pi}_{\mathbf{H}_k}^{\mathrm{T}} = \mathbf{\Pi}_{\mathbf{H}_k}^{\mathrm{T}}\mathbf{\Pi}_{\mathbf{H}_k} = \mathbf{I}_{3N}$. Note that in order to ensure the orthogonality of $\mathbf{Q}_{\mathbf{H}_k}$, it is necessary to apply column pivoting on $\mathbf{H}_k$ [105, Section 5.4.1], due to the fact that $\mathbf{H}_k$ is rank deficient (see Remark 18).

Substituting (5.13) into (5.9), we obtain the EKF update equations based on QR factorization, i.e.,

$$\begin{aligned}\mathbf{P}_{k|k} &= \left(\mathbf{P}_{k|k-1}^{-1} + \mathbf{\Pi}_{\mathbf{H}_k}\mathbf{R}_{\mathbf{H}_k}^{\mathrm{T}}\mathbf{R}_{\mathbf{H}_k}\mathbf{\Pi}_{\mathbf{H}_k}^{\mathrm{T}}\right)^{-1} \qquad (5.14)\\ &= \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1}\mathbf{\Pi}_{\mathbf{H}_k}\mathbf{R}_{\mathbf{H}_k}^{\mathrm{T}}\mathbf{\Sigma}_k^{-1}\mathbf{R}_{\mathbf{H}_k}\mathbf{\Pi}_{\mathbf{H}_k}^{\mathrm{T}}\mathbf{P}_{k|k-1},\\ \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{P}_{k|k}\mathbf{H}_k^{\mathrm{T}}\tilde{\mathbf{z}}_{k|k-1}, \qquad (5.15)\end{aligned}$$

where $\mathbf{\Sigma}_k = \mathbf{R}_{\mathbf{H}_k}\mathbf{\Pi}_{\mathbf{H}_k}^{\mathrm{T}}\mathbf{P}_{k|k-1}\mathbf{\Pi}_{\mathbf{H}_k}\mathbf{R}_{\mathbf{H}_k}^{\mathrm{T}} + \mathbf{I}_{3(N-1)}$, and the second equality in (5.14) is established by the matrix inversion lemma.

Note that in contrast to $\mathbf{S}_k$ [see (5.8)], whose dimensions are $2(N-1)N\times 2(N-1)N$, the matrix $\mathbf{\Sigma}_k$ in (5.14) has dimensions only $3(N-1)\times 3(N-1)$. Thus, assuming both $\mathbf{R}_{\mathbf{H}_k}$ and $\mathbf{\Pi}_{\mathbf{H}_k}$ are given, the total time complexity and space complexity of the state and covariance updates using (5.14)–(5.15) are $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2)$, respectively, the

[2] Recall that $\mathbf{rank}(\mathbf{H}_k) = 3(N-1)$ (see Remark 18). In practice, however, due to the accumulation of numerical roundoff errors arising from finite precision matrix computations, it is possible that the number of rows of $\mathbf{R}_{\mathbf{H}_k}$ (and the number of columns of $\mathbf{Q}_{\mathbf{H}_k}$) is between $3(N-1)$ and $3N$.

same order of computational complexity as when using (5.9)–(5.10). Most importantly, $\kappa(\mathbf{R}_{\mathbf{H}_k}) = \kappa(\mathbf{H}_k)$ [105]. Hence, the numerical stability of (5.14)–(5.15) is significantly better as compared to (5.9)–(5.10).

Therefore, in order to ensure that the time complexity and space complexity of the state and covariance updates through QR factorization [see (5.14)–(5.15)] remain $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2)$, we conclude that the maximum number of arithmetic operations and memory usage to implement (5.13) should be less or equal to $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2)$, respectively.

5.4 Standard Householder QR

There exist several methods for performing QR factorization, such as the Cholesky decomposition (CHO), the modified Gram-Schmidt process (MGS), the Givens rotations (GIV), and the Householder transformations (also called Householder reflections) [105, Section 5.2]. Due to its simplicity and numerical stability, we adopt the traditional column-pivoted QR factorization algorithm utilizing Householder transformations, which is termed as the Standard Householder QR in this chapter.[3] In what follows, we provide a brief overview of Householder reflections (see Section 5.4.1), as well as of the Standard Householder QR algorithm (see Section 5.4.2). The main conclusion of this section is that the QR decomposition of $\mathbf{H}_k$, when employing the Standard Householder QR, has time complexity of $\mathcal{O}(N^4)$ and space complexity of $\mathcal{O}(N^3)$ (see Section 5.4.3).

For clarity, the time-step index k is dropped from (5.13) throughout the rest of the chapter, with $m_{\mathbf{H}} = 2(N-1)N$ and $n_{\mathbf{H}} = 3N$ denoting the number of rows and columns of $\mathbf{H}$, respectively, as well as $l_{\mathbf{H}} = \mathbf{rank}(\mathbf{H}) = 3(N-1)$.

[3] We have conducted computational complexity analysis when employing CHO, MGS, and GIV. More specifically, as shown in Appendix B, the time complexity and space complexity of CHO are $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2)$, respectively, due to the sparse structure of $\mathbf{H}_k$. However, CHO is not applicable since it requires $\mathbf{H}_k^{\mathrm{T}}\mathbf{H}_k$ to be positive definite, or equivalently, $\mathbf{H}_k$ to be full column rank [105]. On the other hand, both MGS and GIV require $\mathcal{O}(N^4)$ arithmetic operations and $\mathcal{O}(N^3)$ memory cells (see Appendix B).

5.4.1 Householder reflections

We first introduce an orthogonal and symmetric Householder (reflection) matrix $\mathbf{Q} = \mathbf{I} - \beta\mathbf{v}\mathbf{v}^{\mathrm{T}}$, where $\beta = 2/\|\mathbf{v}\|_2^2$, and the nonzero vector $\mathbf{v}$ is called the Householder vector associated with $\mathbf{Q}$.

Exploiting the structure of $\mathbf{Q}$, we have

$$\mathbf{Q}\mathbf{H} = (\mathbf{I} - \beta\mathbf{v}\mathbf{v}^{\mathrm{T}})\mathbf{H} = \mathbf{H} - \mathbf{v}\big(\beta\mathbf{H}^{\mathrm{T}}\mathbf{v}\big)^{\mathrm{T}}. \tag{5.16}$$

In essence, a Householder update or transformation [see (5.16)] is a matrix-outer-product update $\mathbf{H} - \mathbf{v}\boldsymbol{\zeta}^{\mathrm{T}}$ where $\boldsymbol{\zeta} = \beta\mathbf{H}^{\mathrm{T}}\mathbf{v}$ [105, Section 5.1.4].

Now suppose for a given matrix $\mathbf{H}$, our objective is to seek a Householder matrix $\mathbf{Q}$ (or equivalently, its associated Householder vector $\mathbf{v}$), such that the first column of $\mathbf{Q}\mathbf{H}$ has zeros below its first component. In other words, letting $\mathbf{u}$ denote the first column of $\mathbf{H}$, we seek a vector $\mathbf{v}$ and a scalar r, such that[4]

$$\mathbf{Q}\mathbf{u} = \mathbf{u} - \beta(\mathbf{v}^{\mathrm{T}}\mathbf{u})\mathbf{v} = r\mathbf{e}_1. \tag{5.17}$$

The solution of (5.17) is provided by [105, Section 5.1.2]

$$r = -\mathrm{sign}(u_1)\|\mathbf{u}\|_2 \ , \ \beta = \big(\|\mathbf{u}\|_2^2 + |u_1|\|\mathbf{u}\|_2\big)^{-1}, \tag{5.18}$$

$$\mathbf{v} = \mathbf{u} - r\mathbf{e}_1, \tag{5.19}$$

where u_1 is the first element of $\mathbf{u}$. For notational convenience, we use $\mathbf{u}_{-1}$ to denote the vector obtained by removing the first element of $\mathbf{u}$, i.e., $\mathbf{u} = [u_1 \ \mathbf{u}_{-1}^{\mathrm{T}}]^{\mathrm{T}}$.

Remark 21. *A key observation of* (5.19) *is that the vectors* $\mathbf{v}$ *and* $\mathbf{u}$ *only differ by the first element* ($v_1 = u_1 - r$). *In other words,* $\mathbf{v}_{-1} = \mathbf{u}_{-1}$. *This property will play a pivotal role in the development of the Modified Householder* QR *algorithm in Section 5.5.*

5.4.2 Description of the Standard Householder QR

In the previous section, we have described an approach to generate a Householder matrix $\mathbf{Q}$ such that $\mathbf{Q}\mathbf{H}$ eliminates all except the first element of the first column of $\mathbf{H}$. The Standard Householder QR algorithm (with column pivoting) [105, Algorithm 5.4.1]

[4] $\mathbf{e}_j$ is the unit (column) vector with a 1 in the jth coordinate and 0's elsewhere.

extends the above strategy by applying a sequence of Householder matrices (multiplying $\mathbf{H}$ from left by $\mathbf{Q}$), as well as a sequence of column permutation matrices (multiplying $\mathbf{H}$ from right by $\mathbf{\Pi}$), to recursively transform $\mathbf{H}$ into an upper triangular form $\mathbf{R}_{\mathbf{H}}$ of dimensions $l_{\mathbf{H}} \times n_{\mathbf{H}}$, whose diagonal elements $r_i,\ i = 1, \ldots, l_{\mathbf{H}}$, satisfy $|r_i| \geq |r_j|$ for $1 \leq i < j \leq l_{\mathbf{H}}$.

More specifically, suppose that after $(\ell - 1)$ iterations, the resulting intermediate matrix

$$\mathbf{H}^{(\ell-1)} = \mathbf{Q}_{\ell-1} \cdots \mathbf{Q}_1 \mathbf{H} \mathbf{\Pi}_1 \cdots \mathbf{\Pi}_{\ell-1} \tag{5.20}$$

has the following structure

$$\mathbf{H}^{(\ell-1)} = \begin{bmatrix} \mathbf{H}_1^{(\ell-1)} & \mathbf{H}_2^{(\ell-1)} \end{bmatrix} = \begin{bmatrix} \mathbf{H}_{1,1}^{(\ell-1)} & \mathbf{H}_{1,2}^{(\ell-1)} \\ \mathbf{0} & \mathbf{H}_{2,2}^{(\ell-1)} \end{bmatrix}, \tag{5.21}$$

where $\mathbf{H}_1^{(\ell-1)}$ consists of the first $\ell - 1$ columns of $\mathbf{H}^{(\ell-1)}$, and $\mathbf{H}_{1,1}^{(\ell-1)}$ is an upper triangular matrix of dimensions $(\ell - 1) \times (\ell - 1)$ with its diagonal elements arranged in decreasing order of absolute value. To facilitate the subsequent derivations, $\mathbf{H}_{1,2}^{(\ell-1)}$ and $\mathbf{H}_{2,2}^{(\ell-1)}$ are further partitioned into

$$\mathbf{H}_{1,2}^{(\ell-1)} = \begin{bmatrix} \mathbf{t} & \overline{\mathbf{H}}_{1,2}^{(\ell-1)} \end{bmatrix}, \tag{5.22}$$

$$\mathbf{H}_{2,2}^{(\ell-1)} = \begin{bmatrix} \mathbf{u} & \overline{\mathbf{H}}_{2,2}^{(\ell-1)} \end{bmatrix} = \begin{bmatrix} u_1 & \bar{\mathbf{s}}^{\mathrm{T}} \\ \mathbf{u}_{-1} & \underline{\mathbf{H}}_{2,2}^{(\ell-1)} \end{bmatrix}, \tag{5.23}$$

where $\mathbf{t}$ is the first column of $\mathbf{H}_{1,2}^{(\ell-1)}$, $\mathbf{u}$ denotes the first column of $\mathbf{H}_{2,2}^{(\ell-1)}$, and the row vector $\bar{\mathbf{s}}^{\mathrm{T}}$ represents the first row of $\overline{\mathbf{H}}_{2,2}^{(\ell-1)}$.

At the ℓth iteration, the Standard Householder QR first performs (i) column pivoting (i.e., right-multiplying $\mathbf{H}^{(\ell-1)}$ with $\mathbf{\Pi}_\ell$), followed by (ii) a Householder update by left-multiplying $\mathbf{H}^{(\ell-1)}\mathbf{\Pi}_\ell$ with $\mathbf{Q}_\ell$, resulting in an intermediate matrix $\mathbf{H}^{(\ell)} = \mathbf{Q}_\ell \mathbf{H}^{(\ell-1)} \mathbf{\Pi}_\ell = \mathbf{Q}_\ell \cdots \mathbf{Q}_1 \mathbf{H} \mathbf{\Pi}_1 \cdots \mathbf{\Pi}_\ell$. Next, we briefly describe these two steps.

Column pivoting

Here we compute the squared norm of every column of $\mathbf{H}_{2,2}^{(\ell-1)}$, denoted as $c_\ell, \ldots, c_{n_{\mathbf{H}}}$, and seek $\ell^* = \arg\max_j \{c_j,\ j = \ell, \ldots, n_{\mathbf{H}}\}$ and $c^* = c_{\ell^*}$. In fact, c_j $(j = \ell, \ldots, n_{\mathbf{H}})$

can be updated recursively through a formula discovered by Businger and Golub [125] (see Algorithm 5, Line 11), which significantly reduces the computational requirements associated with column pivoting [105, Section 5.4.1]. The permutation matrix $\mathbf{\Pi}_\ell$ is generated by exchanging the unit vectors $\mathbf{e}_\ell$ and $\mathbf{e}_{\ell^*}$ in $\mathbf{I}_{n_\mathbf{H}}$. Furthermore, $\mathbf{H}^{(\ell-1)}\mathbf{\Pi}_\ell$ can be found by simply swapping the ℓth and ℓ^*th columns of $\mathbf{H}^{(\ell-1)}$. Since the computational overhead of column pivoting is not a dominant factor (see Section 5.4.3), in what follows, we assume $\mathbf{\Pi}_\ell = \mathbf{I}_{n_\mathbf{H}}$ for simplicity.

Householder update

Here our objective is to find $\mathbf{Q}_\ell = \mathbf{I}_{m_\mathbf{H}} - \beta_\ell \mathbf{v}_\ell \mathbf{v}_\ell^{\mathrm{T}}$ such that the first ℓ columns of $\mathbf{H}^{(\ell)} = \mathbf{Q}_\ell \mathbf{H}^{(\ell-1)}\mathbf{\Pi}_\ell = (\mathbf{I}_{m_\mathbf{H}} - \beta_\ell \mathbf{v}_\ell \mathbf{v}_\ell^{\mathrm{T}})\mathbf{H}^{(\ell-1)}\mathbf{\Pi}_\ell$ are upper triangular. Since $\mathbf{H}_{1,1}^{(\ell-1)}$ is already upper triangular, the Householder matrix $\mathbf{Q}_\ell$ simply eliminates all except the first component of $\mathbf{u}$ [see (5.23)], while leaving $\mathbf{H}_1^{(\ell-1)}$ intact. This is achieved by selecting $\mathbf{v}_\ell = [\mathbf{0}_{1\times(\ell-1)}\ \mathbf{v}^{\mathrm{T}}]^{\mathrm{T}}$, with [105, Section 5.2.1]

$$\mathbf{v} = \mathbf{u} + \mathrm{sign}(u_1)(c^*)^{1/2}\mathbf{e}_1, \tag{5.24}$$

where $c^* = \|\mathbf{u}\|_2^2$ and $\beta_\ell = (c^* + |u_1|(c^*)^{1/2})^{-1}$. Note that $c^* = c_{\ell^*}$ is readily available after the column pivoting stage and needs not to be recomputed through $c^* = \mathbf{u}^{\mathrm{T}}\mathbf{u}$.

Following the notations defined in (5.22)–(5.23), we are able to establish Proposition 22 regarding the structure of the intermediate matrix $\mathbf{H}^{(\ell)}$.

Proposition 22. *$\mathbf{H}^{(\ell)}$ has the following block structure*

$$\mathbf{H}^{(\ell)} = \begin{bmatrix}\mathbf{H}_1^{(\ell)} & \mathbf{H}_2^{(\ell)}\end{bmatrix} = \begin{bmatrix}\mathbf{H}_{1,1}^{(\ell)} & \mathbf{H}_{1,2}^{(\ell)} \\ \mathbf{0} & \mathbf{H}_{2,2}^{(\ell)}\end{bmatrix}, \tag{5.25}$$

where $\mathbf{H}_1^{(\ell)}$ corresponds to the first ℓ columns of $\mathbf{H}^{(\ell)}$, $\mathbf{H}_{1,1}^{(\ell)} = \begin{bmatrix}\mathbf{H}_{1,1}^{(\ell-1)} & \mathbf{t} \\ \mathbf{0} & r_\ell\end{bmatrix}$ is an upper triangular matrix of dimensions $\ell \times \ell$, and $\mathbf{H}_{1,2}^{(\ell)} = \begin{bmatrix}\big(\overline{\mathbf{H}}_{1,2}^{(\ell-1)}\big)^{\mathrm{T}} & \mathbf{s}\end{bmatrix}^{\mathrm{T}}$, with

$$r_\ell = -\mathrm{sign}(u_1)(c^*)^{1/2}, \tag{5.26}$$

$$\mathbf{s} = \bar{\mathbf{s}} - v_1\boldsymbol{\delta}, \tag{5.27}$$

$$\mathbf{H}_{2,2}^{(\ell)} = \underline{\mathbf{H}}_{2,2}^{(\ell-1)} - \mathbf{u}_{-1}\boldsymbol{\delta}^{\mathrm{T}}, \tag{5.28}$$

where $\boldsymbol{\delta} = \beta_\ell\big(\overline{\mathbf{H}}_{2,2}^{(\ell-1)}\big)^{\mathrm{T}}\mathbf{v}$.

Proof. The proof is straightforward. First, from the definition of $\mathbf{v}_\ell$, we have $\mathbf{Q}_\ell = \mathbf{diag}(\mathbf{I}_{\ell-1}, \mathbf{Q})$, where $\mathbf{Q} = \mathbf{I}_{m_\mathbf{H}-\ell+1} - \beta_\ell \mathbf{v}\mathbf{v}^{\mathrm{T}}$. Under the assumption $\mathbf{\Pi}_\ell = \mathbf{I}_{n_\mathbf{H}}$, we have

$$\mathbf{H}^{(\ell)} = \mathbf{Q}_\ell \mathbf{H}^{(\ell-1)} \mathbf{\Pi}_\ell = \begin{bmatrix} \mathbf{H}_{1,1}^{(\ell-1)} & \mathbf{H}_{1,2}^{(\ell-1)} \\ \mathbf{0} & \mathbf{Q}\mathbf{H}_{2,2}^{(\ell-1)} \end{bmatrix} = \begin{bmatrix} \mathbf{H}_{1,1}^{(\ell-1)} & \mathbf{t} & \overline{\mathbf{H}}_{1,2}^{(\ell-1)} \\ \mathbf{0} & \mathbf{Q}\mathbf{u} & \mathbf{Q}\overline{\mathbf{H}}_{2,2}^{(\ell-1)} \end{bmatrix}. \tag{5.29}$$

Since $\mathbf{v}$ is chosen to eliminate all except the first component of $\mathbf{u}$ [see (5.24)], we have $\mathbf{Q}\mathbf{u} = r_\ell \mathbf{e}_1$ with the expression of r_ℓ provided in (5.26) [see (5.18)]. On the other hand [see (5.16)],

$$\mathbf{Q}\overline{\mathbf{H}}_{2,2}^{(\ell-1)} = \overline{\mathbf{H}}_{2,2}^{(\ell-1)} - \mathbf{v}\Big(\beta_\ell \big(\overline{\mathbf{H}}_{2,2}^{(\ell-1)}\big)^{\mathrm{T}} \mathbf{v}\Big)^{\mathrm{T}} = \begin{bmatrix} \bar{\mathbf{s}}^{\mathrm{T}} \\ \underline{\mathbf{H}}_{2,2}^{(\ell-1)} \end{bmatrix} - \begin{bmatrix} v_1 \\ \mathbf{v}_{-1} \end{bmatrix} \boldsymbol{\delta}^{\mathrm{T}} = \begin{bmatrix} \bar{\mathbf{s}}^{\mathrm{T}} - v_1 \boldsymbol{\delta}^{\mathrm{T}} \\ \underline{\mathbf{H}}_{2,2}^{(\ell-1)} - \mathbf{u}_{-1} \boldsymbol{\delta}^{\mathrm{T}} \end{bmatrix}, \tag{5.30}$$

where the last equality comes from $\mathbf{v}_{-1} = \mathbf{u}_{-1}$ [see Remark 21]. Comparing (5.29)–(5.30) with (5.25) and matching corresponding terms, we arrive at (5.27)–(5.28). □

In summary, we have briefly described (i) column pivoting and (ii) Householder update at the ℓth iteration of the Standard Householder QR. The algorithm terminates when $c^* < \epsilon$, where ϵ is a small positive scalar. Since the total number of iterations is $l_\mathbf{H} = \mathbf{rank}(\mathbf{H})$, the output $\mathbf{R}_\mathbf{H}$ is selected as the first $l_\mathbf{H}$ rows of $\mathbf{H}^{(l_\mathbf{H})}$. Furthermore, $\mathbf{\Pi}_\mathbf{H}$ is represented and updated through a permutation vector $[\pi_1 \ \dots \ \pi_{n_\mathbf{H}}]$, i.e., $\mathbf{\Pi}_\mathbf{H} = [\mathbf{e}_{\pi_1} \ \dots \ \mathbf{e}_{\pi_{n_\mathbf{H}}}]$ (see [105]). For completeness, the flow chart of the Standard Householder QR [105, Algorithm 5.4.1] is summarized in Algorithm 5.

5.4.3 Complexity analysis of the Standard Householder QR

We now analyze the computational requirements of factorizing $\mathbf{H}$ when employing the Standard Householder QR. The main result is that the time complexity and space complexity are $\mathcal{O}(N^4)$ and $\mathcal{O}(N^3)$, respectively, which makes the time and space complexity of the EKF-based CL also $\mathcal{O}(N^4)$ and $\mathcal{O}(N^3)$, respectively.

To proceed, we note that the overall computational cost of the QR decomposition is contributed from two sources: (i) column pivoting and (ii) Householder update (see Section 5.4.2). More specifically, Lines 1–4, 6, 10–13 in Algorithm 5 correspond to column pivoting, while Lines 7–9 correspond to a Householder update. In what follows, we examine the computational cost associated with each source separately, and then draw conclusion about the overall computational complexity.

Algorithm 5 Standard Householder QR

Require: $\mathbf{H}^{(0)} = \mathbf{H}$ of dimensions $m_{\mathbf{H}} \times n_{\mathbf{H}}$ ($m_{\mathbf{H}} \geq n_{\mathbf{H}}$).

Ensure: $\mathbf{R}_{\mathbf{H}}$ of dimensions $l_{\mathbf{H}} \times n_{\mathbf{H}}$ and $\mathbf{\Pi}_{\mathbf{H}}$ of dimensions $n_{\mathbf{H}} \times n_{\mathbf{H}}$.

1: **for** $i = 1$ **to** $n_{\mathbf{H}}$, **do**
2: Initialize $\pi_i = i$, and compute c_i, the squared norm of the ith column of $\mathbf{H}^{(0)}$.
3: **end for**
4: Set $\ell = 1$, determine $\ell^* = \arg\max_j \{c_j,\, j = \ell, \ldots, n_{\mathbf{H}}\}$ and $c^* = c_{\ell^*}$.
5: **while** $c^* > \epsilon$ **do**
6: Swap the ℓth and ℓ^*th columns of $\mathbf{H}^{(\ell-1)}$, exchange c_ℓ and c_{ℓ^*}, and interchange π_ℓ with π_{ℓ^*}.
7: Calculate $\beta_\ell, \mathbf{v}$ from $\mathbf{u}$ using (5.24), and r_ℓ from c^* using (5.26).
8: Compute $\boldsymbol{\delta}$ and $\mathbf{s} = [s_1 \ \ldots \ s_{n_{\mathbf{H}}-\ell}]^{\mathrm{T}}$ [see (5.27)].
9: Update the sub-matrix $\mathbf{H}_{2,2}^{(\ell)}$ [see (5.28)].
10: **for** $j = \ell + 1$ **to** $n_{\mathbf{H}}$, **do**
11: Modify $c_j \Leftarrow c_j - s_{j-\ell}^2$.
12: **end for**
13: Seek $\ell^* = \arg\max_j \{c_j,\, j = \ell + 1, \ldots, n_{\mathbf{H}}\}$ and $c^* = c_{\ell^*}$, and then $\ell \Leftarrow \ell + 1$.
14: **end while**
15: **return** $\mathbf{\Pi}_{\mathbf{H}} = [\mathbf{e}_{\pi_1} \ \ldots \ \mathbf{e}_{\pi_{n_{\mathbf{H}}}}]$, and $\mathbf{R}_{\mathbf{H}}$, selected as the first $l_{\mathbf{H}}$ rows of $\mathbf{H}^{(l_{\mathbf{H}})}$, where the total number of iterations $l_{\mathbf{H}} = \mathbf{rank}(\mathbf{H})$.

Complexity of column pivoting

The following proposition summarizes the computational requirements associated with column pivoting (see Lines 1–4, 6, 10–13 in Algorithm 5).

Proposition 23. *Due to the sparsity of* $\mathbf{H}$*, the time complexity and space complexity associated with column pivoting are* $\mathcal{O}(N^2)$ *and* $\mathcal{O}(N)$*, respectively.*

Proof. First, we investigate the space complexity. Note that the extra variables introduced for implementing column pivoting are $[c_1 \ \dots \ c_{n_{\mathbf{H}}}]$ and $[\pi_1 \ \dots \ \pi_{n_{\mathbf{H}}}]$, both of dimension $n_{\mathbf{H}}$, and thus the space complexity is $\mathcal{O}(N)$.

Second, for time complexity, we divide the overall arithmetic operations into two parts, namely (i) initial generation of $c_j, j = 1, \dots, n_{\mathbf{H}}$, in Lines 1–4, and (ii) sequential update of $c_j, j = \ell + 1, \dots, n_{\mathbf{H}}$, corresponding to Lines 6, 10–13 at the ℓth iteration of the while-loop.

During the initialization stage, calculating every $c_j, j = 1, \dots, n_{\mathbf{H}}$, involves $\mathcal{O}(N)$ operations, since each column of $\mathbf{H}$ has at most $4(N-1)$ non-zeros (see Remark 18). Furthermore, seeking the maximum element among $n_{\mathbf{H}} = 3N$ scalars can be achieved in $3N-1$ steps. Thus, the initial generation of $c_j,\ j = 1, \dots, n_{\mathbf{H}}$, and c^* can be fulfilled in $\mathcal{O}(N^2)$ basic steps.

To analyze the time complexity during the update phase at the ℓth iteration, we first note that the exchange of the ℓth and ℓ^*th columns of $\mathbf{H}^{(\ell-1)}$, as well as c_ℓ and c_{ℓ^*} (in Line 6) can be efficiently implemented using pointers, without physically swapping real data in the memory. On the other hand, given $\mathbf{s}$, the total number of operations for updating $c_j, j = \ell + 1, \dots, n_{\mathbf{H}}$ (see Lines 10–12) is $n_{\mathbf{H}} - \ell$, and the maximum-seeking operation in Line 13 can be achieved in $n_{\mathbf{H}} - \ell$ basic steps. Hence, the number of basic operations for the update phase at the ℓth iteration is of order $n_{\mathbf{H}} - \ell$. Therefore, the total time complexity associated with the sequential update is $\mathcal{O}(N^2)$.

In summary, we conclude that the overall time complexity of column pivoting is $\mathcal{O}(N^2)$. □

Remark 24. *As it will become evident later on, the computational complexity of column pivoting is at least one order of magnitude less than that of the Householder update.*

For this reason, without loss of generality, in what follows we assume that $\mathbf{\Pi}_\ell = \mathbf{I}_{n_\mathbf{H}}, \ell = 1, \ldots, l_\mathbf{H}$, and thus $\mathbf{\Pi}_\mathbf{H} = \mathbf{I}_{n_\mathbf{H}}$ with $[\pi_1 \ \ldots \ \pi_{n_\mathbf{H}}] = [1 \ \ldots \ n_\mathbf{H}]$.

Complexity of the Householder update

Next, we address the computational complexity of the Householder update in the Standard Householder QR (see Lines 7–9 in Algorithm 5). For presentation clarity, we introduce a matrix $\mathbf{\Lambda}$ of dimensions $m_\mathbf{\Lambda} \times n_\mathbf{\Lambda}$, with $m_\mathbf{\Lambda} = (N-1)N/2 = m_\mathbf{H}/4$ and $n_\mathbf{\Lambda} = N = n_\mathbf{H}/3$ [see (5.31)].

$$\mathbf{\Lambda} = \begin{bmatrix} \psi^{1,2} & \upsilon^{1,2} & & & & & \\ \psi^{1,3} & & \upsilon^{1,3} & & & & \\ \vdots & & & \ddots & & & \\ \psi^{1,N} & & & & & & \upsilon^{1,N} \\ & \psi^{2,3} & \upsilon^{2,3} & & & & \\ & \psi^{2,4} & & \upsilon^{2,4} & & & \\ & \vdots & & & \ddots & & \\ & \psi^{2,N} & & & & & \upsilon^{2,N} \\ & & \vdots & \cdots & & & \vdots \\ & & & & \psi^{N-1,N} & & \upsilon^{N-1,N} \end{bmatrix}. \tag{5.31}$$

Specifically, $\mathbf{\Lambda}$ is constructed by first removing from $\mathbf{H}$ the measurement matrices $\mathbf{H}^{i,j}$ for all $j < i$. Second, every pair of matrices $\mathbf{\Psi}^{i,j}$ and $\mathbf{\Upsilon}^{i,j}$ $(i < j)$ of dimensions 2×3 in $\mathbf{H}$ [see (5.6)] are replaced by nonzero scalars $\psi^{i,j}, \upsilon^{i,j}$ in $\mathbf{\Lambda}$ [see (5.31)]. Since both $m_\mathbf{H}, m_\mathbf{\Lambda}$ are quadratic in N, $n_\mathbf{H}, n_\mathbf{\Lambda}$ are linear in N, and most importantly, the matrices $\mathbf{\Psi}^{i,j}, \mathbf{\Upsilon}^{i,j}$ have fixed dimensions independent of N, the computational complexity of the QR decomposition of $\mathbf{\Lambda}$ in (5.31) is of the same order as that of the QR decomposition of $\mathbf{H}$ in (5.6). Thus, for simplicity, from now on we will focus on the computational requirements of factorizing $\mathbf{\Lambda}$.

To facilitate the ensuing discussion, we introduce a useful lemma which shows that the number of arithmetic operations of the Householder update at the ℓth iteration is proportional to $\mathbf{nnz}(\mathbf{u})$, where $\mathbf{u}$ is the first column of $\mathbf{\Lambda}_{2,2}^{(\ell-1)}$ obtained by removing the first $\ell - 1$ rows and columns of $\mathbf{\Lambda}^{(\ell-1)}$.

Lemma 25. *Let* $\mathbf{u}$ *represents the first column of* $\mathbf{\Lambda}_{2,2}^{(\ell-1)}$ *and suppose* $\mathbf{nnz}(\mathbf{u}) = \tau_{\ell-1}$*, then the number of basic arithmetic operations for implementing Lines 7–9 in Algorithm 5 at the* ℓ*th iteration of the Standard Householder* QR *is of* $\mathcal{O}((n_{\mathbf{\Lambda}} - \ell)\tau_{\ell-1})$.

Proof. First, we notice that $c^* = \|\mathbf{u}\|_2^2$ is readily available from the column pivoting stage (see Line 4 for $\ell = 1$ and Line 13 for $\ell > 1$ in Algorithm 5), and needs not to be recomputed. From (5.24), (5.26) and Remark 21, it is clear that the calculations of $\mathbf{v}, \beta_\ell$, and r_ℓ (in Line 7) require constant number of operations. Furthermore, $\mathbf{u}$ and $\mathbf{v}$ have exactly the same sparsity pattern, and $\mathbf{nnz}(\mathbf{v}) = \tau_{\ell-1}$.

Second, we examine the number of basic operations required for computing the matrix-vector product $\boldsymbol{\delta} = \beta_\ell \big(\overline{\mathbf{\Lambda}}_{2,2}^{(\ell-1)}\big)^{\mathrm{T}}\mathbf{v}$, whose element δ_j is simply the inner product of $\beta_\ell\mathbf{v}$ and the jth column of $\overline{\mathbf{\Lambda}}_{2,2}^{(\ell-1)}, j = 1, \ldots, n_{\mathbf{\Lambda}} - \ell$. Since $\mathbf{nnz}(\mathbf{v}) = \tau_{\ell-1}$, computing δ_j only requires $\tau_{\ell-1}$ scalar multiplications and $\tau_{\ell-1} - 1$ scalar additions. Therefore, computing $\boldsymbol{\delta}$ involves $\mathcal{O}((n_{\mathbf{\Lambda}} - \ell)\tau_{\ell-1})$ operations. Once $\boldsymbol{\delta}$ is computed, updating $\mathbf{s} = [s_1 \ \ldots \ s_{n_{\mathbf{\Lambda}}-\ell}]^{\mathrm{T}}$ is straightforward with $n_{\mathbf{\Lambda}} - \ell$ basic steps [see (5.27)]. Thus, the overall time complexity of Line 8 is $\mathcal{O}((n_{\mathbf{\Lambda}} - \ell)\tau_{\ell-1})$.

Third, (5.28) is an outer-product update, or rank-one modification. Note that this rank-one update process has a cost of $(n_{\mathbf{\Lambda}} - \ell)(\tau_{\ell-1} - 1)$. In particular, we only need to modify those rows of $\underline{\mathbf{\Lambda}}_{2,2}^{(\ell-1)}$ whose indices correspond to the indices of the nonzero elements of $\mathbf{u}_{-1}$.

In summary, the time complexity of the Householder update at the ℓth iteration, dominated by the computation of $\boldsymbol{\delta}$ and the rank-one update of $\mathbf{\Lambda}_{2,2}^{(\ell)}$, is $\mathcal{O}((n_{\mathbf{\Lambda}} - \ell)\tau_{\ell-1})$. □

Based on Lemma 25, the overall time complexity associated with the Householder update is of order $\sum_{\ell=1}^{n_{\mathbf{\Lambda}}}(n_{\mathbf{\Lambda}} - \ell)\tau_{\ell-1}$. On the other hand, the subsequent symbolic analysis reveals that the storage of the intermediate matrix $\mathbf{\Lambda}^{(\ell)}$, i.e., $\mathbf{nnz}(\mathbf{\Lambda}^{(\ell)})$, $\ell = 1, \ldots, n_{\mathbf{\Lambda}}$, is the most dominant factor in terms of memory-cell usage. Furthermore, both τ_ℓ and $\mathbf{nnz}(\mathbf{\Lambda}^{(\ell)})$ are closely related to a quantity $\bar{\tau}_\ell$, denoting the number of non-zeros in each column of $\mathbf{\Lambda}_2^{(\ell)}$ obtained by removing the first ℓ columns of $\mathbf{\Lambda}^{(\ell)}$.[5] More

[5] As the following analysis shows, each column of $\mathbf{\Lambda}_2^{(\ell)}$ always has the same number of non-zeros, denoted as $\bar{\tau}_\ell$ [see (5.34)]. Hence, $\mathbf{nnz}(\mathbf{\Lambda}_2^{(\ell)}) = (n_{\mathbf{\Lambda}} - \ell)\bar{\tau}_\ell$.

specifically, from (5.22)–(5.23), we have

$$\bar{\tau}_{\ell-1} = \mathbf{nnz}(\mathbf{t}) + \mathbf{nnz}(\mathbf{u}) \quad \Rightarrow \quad \tau_{\ell-1} = \mathbf{nnz}(\mathbf{u}) = \bar{\tau}_{\ell-1} - (\ell - 1), \tag{5.32}$$

where $\mathbf{nnz}(\mathbf{t}) = \ell - 1$ since the vector $\mathbf{t}$ of dimension $\ell - 1$ is in general dense.

Similarly, based on the matrix partition in (5.21) and recalling that $\mathbf{\Lambda}_{1,1}^{(\ell)}$ is a dense upper triangular matrix of dimensions $\ell \times \ell$, the relationship between $\mathbf{nnz}(\mathbf{\Lambda}^{(\ell)})$ and $\bar{\tau}_\ell$ can be readily established as follows, i.e.,

$$\mathbf{nnz}(\mathbf{\Lambda}^{(\ell)}) = \mathbf{nnz}(\mathbf{\Lambda}_1^{(\ell)}) + \mathbf{nnz}(\mathbf{\Lambda}_2^{(\ell)}) \;\Rightarrow\; \mathbf{nnz}(\mathbf{\Lambda}^{(\ell)}) = \ell(\ell+1)/2 + (n_{\mathbf{\Lambda}} - \ell)\bar{\tau}_\ell. \tag{5.33}$$

In what follows, our objective is to seek the expression of $\bar{\tau}_\ell$ ($\ell = 0, \ldots, n_{\mathbf{\Lambda}} - 1$) in order to determine the time and space complexity associated with the Householder update. To do so, we perform a symbolic analysis which is based solely on the nonzero pattern of $\mathbf{\Lambda}$ [see (5.31)]. Additionally, we provide the explicit structures of $\mathbf{\Lambda}^{(1)}$ and $\mathbf{\Lambda}^{(2)}$ [see (5.35)–(5.36)], the intermediate results after the first and second iteration of the Householder update, respectively, which clearly demonstrate that the Householder update destroys the original sparsity of $\mathbf{\Lambda}$. We then deduce the expression of $\bar{\tau}_\ell$ and the sparsity pattern of $\mathbf{\Lambda}^{(\ell)}$ for arbitrary ℓ through induction. The main result of our analysis is summarized as

$$\bar{\tau}_\ell = \sum_{i=1}^{\ell+1}(N - i), \quad \ell = 0, \ldots, n_{\mathbf{\Lambda}} - 1. \tag{5.34}$$

To proceed, we note that initially, $\mathbf{\Lambda}_2^{(0)} = \mathbf{\Lambda}^{(0)} = \mathbf{\Lambda}$ has a sparse structure shown in (5.31). Since each column of $\mathbf{\Lambda}$ has $N - 1$ non-zeros, thus $\bar{\tau}_0 = N - 1$. It is straightforward to verify that the right-hand side of (5.34) is $N - 1$ by substituting $\ell = 0$, and hence (5.34) holds for $\ell = 0$.

• At the first iteration of the Householder update:

Now we examine the nonzero pattern of the intermediate matrix $\mathbf{\Lambda}^{(1)}$ after the first Householder matrix $\mathbf{Q}_1$ has been applied. Based on (5.28), we conclude that the first $N - 1$ elements of each column of $\mathbf{\Lambda}_2^{(1)}$ become nonzero [see (5.35)] due to the nonzero elements of $\mathbf{u}$ (the first column of $\mathbf{\Lambda}_{2,2}^{(0)} = \mathbf{\Lambda}^{(0)}$), i.e., $\psi^{1,j}, j = 2, \ldots, N$. Therefore the original sparsity of $\mathbf{\Lambda}^{(0)}$ is destroyed and $\mathbf{\Lambda}^{(1)} = \mathbf{Q}_1\mathbf{\Lambda}^{(0)}$ has the following dense

structure[6]

$$\mathbf{\Lambda}^{(1)} = \begin{bmatrix} \times & \times & \boxtimes & \boxtimes & \cdots & \boxtimes \\ & \boxtimes & \times & \boxtimes & \cdots & \boxtimes \\ & \vdots & \vdots & \vdots & \ddots & \vdots \\ & \boxtimes & \boxtimes & \boxtimes & \cdots & \times \\ & \psi^{2,3} & \upsilon^{2,3} & & & \\ & \psi^{2,4} & & \upsilon^{2,4} & & \\ & \vdots & & & \ddots & \\ & \psi^{2,N} & & & & \upsilon^{2,N} \\ & & \vdots & \cdots & & \vdots \\ & & & & \psi^{N-1,N} & \upsilon^{N-1,N} \end{bmatrix}. \tag{5.35}$$

Furthermore, in contrast to $N-1$ nonzero elements per column of $\mathbf{\Lambda}^{(0)}$, we observe that every column of $\mathbf{\Lambda}_2^{(1)}$ has

$$\bar{\tau}_1 = \bar{\tau}_0 + (N-2) = \sum_{i=1}^{2}(N-i)$$

non-zeros [see (5.35)], which confirms the validity of (5.34) for $\ell = 1$. These extra $N-2$ non-zero fill-ins introduced to each column of $\mathbf{\Lambda}_2^{(1)}$ will impact the computational requirements of the Householder update at subsequent steps.

• At the second iteration of the Householder update:

Similarly, after the second Householder matrix $\mathbf{Q}_2$ has been applied, the resulting matrix $\mathbf{\Lambda}^{(2)} = \mathbf{Q}_2\mathbf{\Lambda}^{(1)}$ has a even denser structure shown in (5.36). Specifically, $\mathbf{Q}_2$ further destroys the sparse pattern of $\mathbf{\Lambda}^{(1)}$ by introducing extra nonzero elements through $\psi^{2,j}, j = 3, \ldots, N$, to the corresponding rows of the sub-matrix $\mathbf{\Lambda}_2^{(1)}$. This results in every column of $\mathbf{\Lambda}_2^{(2)}$ having

$$\bar{\tau}_2 = \bar{\tau}_1 + (N-3) = \sum_{i=1}^{3}(N-i)$$

non-zeros [see (5.36)], verifying (5.34) for $\ell = 2$. Again, the extra nonzero fill-ins introduced at the second iteration will increase the computational cost of the Householder

[6] We use " × " to denote a nonzero element, whereas " ⊠ " highlights a nonzero fill-in.

update at subsequent iterations.

$$\mathbf{\Lambda}^{(2)} = \begin{bmatrix} \times & \times & \times & \times & \cdots & \times \\ & \times & \times & \times & \cdots & \times \\ & & \vdots & \vdots & \ddots & \vdots \\ & & \times & \times & \cdots & \times \\ & & \times & \boxtimes & \cdots & \boxtimes \\ & & \boxtimes & \times & \cdots & \boxtimes \\ & & \vdots & \vdots & \ddots & \vdots \\ & & \boxtimes & \boxtimes & \cdots & \times \\ & & \psi^{3,4} & \upsilon^{3,4} & & \\ & & \vdots & \cdots & & \vdots \\ & & & & \psi^{N-1,N} & \upsilon^{N-1,N} \end{bmatrix}. \tag{5.36}$$

• At the ℓth iteration of the Householder update:

Following the above analysis, we prove (5.34) by mathematical induction. Specifically, suppose after $\ell - 1$ iterations, every column of $\mathbf{\Lambda}_2^{(\ell-1)}$ has

$$\bar{\tau}_{\ell-1} = \sum_{i=1}^{\ell}(N-i)$$

nonzero elements. Then, at the ℓth iteration, the Householder matrix $\mathbf{Q}_\ell$ left-multiplying $\mathbf{\Lambda}^{(\ell-1)}$ results in the intermediate matrix $\mathbf{\Lambda}^{(\ell)} = \mathbf{Q}_\ell \mathbf{\Lambda}^{(\ell-1)}$. Compared to $\mathbf{\Lambda}^{(\ell-1)}$, extra non-zeros are introduced into those rows of the sub-matrix $\mathbf{\Lambda}_2^{(\ell)}$, whose indices match the linear indices of the non-zeros $\psi^{\ell,j}, j = \ell+1, \ldots, N$. Hence, the number of nonzero elements of each column of $\mathbf{\Lambda}_2^{(\ell)}$ increases to

$$\bar{\tau}_\ell = \bar{\tau}_{\ell-1} + [N - (\ell+1)] = \sum_{i=1}^{\ell+1}(N-i),$$

which verifies the correctness of (5.34).

• Summary of the complexity of the Householder update:

Once $\bar{\tau}_\ell$ is available, we proceed to calculate the computational requirements associated with the Householder update. Substituting (5.34) into (5.32)–(5.33), we obtain

$$\tau_{\ell-1} = \ell N - \ell^2/2 - 3\ell/2 + 1, \tag{5.37}$$

$$\mathbf{nnz}(\mathbf{\Lambda}^{(\ell)}) = (\ell+1)\ell/2 + (N-\ell)(\ell+1)(2N-\ell-2)/2. \tag{5.38}$$

From (5.37), we conclude that the overall time complexity of the Householder update in the Standard Householder QR is in the order of

$$\sum_{\ell=1}^{n_{\mathbf{\Lambda}}}(n_{\mathbf{\Lambda}}-\ell)\tau_{\ell-1}\sim\mathcal{O}(N^4).$$

Similarly, based on the expression of $\mathbf{nnz}(\mathbf{\Lambda}^{(\ell)})$ in (5.38), we conclude that the overall space complexity of the Householder update, dominated by the memory-cell usage of the intermediate matrix $\mathbf{\Lambda}^{(\ell)}$, is of

$$\max_{1\leq\ell\leq N}\ \mathbf{nnz}(\mathbf{\Lambda}^{(\ell)})\sim\mathcal{O}(N^3).$$

Summary of complexity analysis

In summary, we conclude that the QR factorization of $\mathbf{\Lambda}$ through the Standard Householder QR, requires $\mathcal{O}(N^4)$ basic arithmetic operations, as well as $\mathcal{O}(N^3)$ memory usage. Since the complexity of the QR decomposition of $\mathbf{H}$ [see (5.6)] is of the same order as of $\mathbf{\Lambda}$ [see (5.31)], we conclude that for CL, the overall computational complexity of the EKF update through QR factorization is dominated by the QR decomposition of $\mathbf{H}$, as summarized by the following theorem:

Theorem 26. *The EKF-based CL through QR factorization, when employing the Standard Householder QR algorithm, has time complexity of $\mathcal{O}(N^4)$ and space complexity of $\mathcal{O}(N^3)$.*

5.5 Modified Householder QR

As evident from the symbolic analysis described in Section 5.4.3, the main reason of the high computational cost of the Standard Householder QR is because, at every iteration ℓ, the Standard Householder QR uses its previous intermediate and dense matrix $\mathbf{H}^{(\ell-1)}$ to generate $\mathbf{Q}_\ell$, which inevitably introduces extra non-zero fill-ins in $\mathbf{H}^{(\ell)}$ that further destroys the original sparsity of $\mathbf{H}$. These extra fill-ins increase both the number of arithmetic operations and memory-cell usage. As a result, the overall computational complexity increases to $\mathcal{O}(N^4)$ in time complexity and $\mathcal{O}(N^3)$ in space complexity.

This motivates us to modify the procedures of the Standard Householder QR so as to preserve the sparsity of the original matrix $\mathbf{H}$ and minimize the cost of its QR factorization. Our proposed QR algorithm, namely the Modified Householder QR, is derived from [126], where Kaufman proposed an idea that exploits the sparsity of the original matrix under a sequence of dense Householder transformations. However, in [126], Kaufman assumes that the Householder reflection matrices (or equivalently, the Householder vectors) are known in advance, which is not the case in our scenario. Instead, we first explore the relationships between $\mathbf{u}$ and $\mathbf{v}$ (see Remark 21), and prove that every submatrix $\mathbf{H}_{2,2}^{(\ell-1)},\ \ell = 1, \ldots, l_{\mathbf{H}}$, can be expressed as a linear combination of the original matrix $\mathbf{H}$ (see Proposition 27). Based on this proposition, we introduce key modifications to the Standard Householder QR and develop the Modified Householder QR. In contrast to iteratively updating $\mathbf{H}$ using its previous intermediate but dense results in the Standard Householder QR, Proposition 27 enables us to sequentially transform $\mathbf{H}$ by exploiting its original sparse structure in the Modified Householder QR. As a consequence, the sparsity of $\mathbf{H}$ is preserved, and both the time and space complexity of the QR decomposition of $\mathbf{H}$ when employing the Modified Householder QR are reduced.

In what follows, we first describe the Modified Householder QR in Section 5.5.1, then analyze its computational complexity in Section 5.5.2. Our main result is that the QR factorization of $\mathbf{H}$, when employing the Modified Householder QR, has time complexity $\mathcal{O}(N^3)$ and space complexity $\mathcal{O}(N^2)$.

5.5.1 Description of the Modified Householder QR

In this section, we present a general description of the Modified Householder QR. We do not assume the specific sparse structure of $\mathbf{H}$. Instead, the sparsity of $\mathbf{H}$ [see (5.6)] will be exploited in Section 5.5.2 for evaluating the computational complexity.

To facilitate the description and derivation of the Modified Householder QR, we adopt the same notation used in Section 5.4.2. Furthermore, we use $\mathbf{h}_i$ and $\mathbf{h}_i^{(\ell)}, i = 1, \ldots, n_{\mathbf{H}}$, to denote the ith columns of the original matrix $\mathbf{H}$ and the updated matrix $\mathbf{H}^{(\ell)}$, respectively, with $m_{\mathbf{H}}$ and $n_{\mathbf{H}}$ representing the number of rows and columns of $\mathbf{H}$, and $l_{\mathbf{H}}$ being the rank of $\mathbf{H}$.

As pointed out in Section 5.4.2, the ℓth iteration of the Standard Householder QR is decomposed into two separate phases, column pivoting and Householder update. The

Modified Householder QR performs column pivoting identically to that of the Standard Householder QR, but differs in the implementation of the Householder update. Similarly in Section 5.4.2, due to the minimum computational overhead associated with column pivoting (see Remark 24), in what follows we assume $\mathbf{\Pi}_\ell = \mathbf{I}_{n_\mathbf{H}}, \ell = 1, \ldots, l_\mathbf{H}$, and focus on the implementation of the Householder update in the Modified Householder QR.

In order to describe the essence of the Modified Householder QR, we first establish the following important proposition. More specifically, by recursively applying the property $\mathbf{v}_{-1} = \mathbf{u}_{-1}$, we can show that at each iteration ℓ ($\ell = 1, \ldots, l_\mathbf{H}$), the sub-matrix $\mathbf{H}_{2,2}^{(\ell-1)}$ can be expressed as a linear combination of the original matrix $\mathbf{H}$, summarized in Proposition 27 [127].

Proposition 27. *After $\ell-1$ ($\ell = 1, \ldots, l_\mathbf{H}$) iterations of Householder transformations, every column of the sub-matrix $\mathbf{H}_{2,2}^{(\ell-1)}$ [see (5.21)] can be written as a linear combination of the columns of the original matrix $\mathbf{H}$. More specifically,*

$$\mathbf{H}_{2,2}^{(\ell-1)} = \mathbf{H}_{-(\ell-1)} \begin{bmatrix} \mathbf{\Gamma}^{(\ell-1)} \\ \mathbf{I}_{n_\mathbf{H}-\ell+1} \end{bmatrix}, \tag{5.39}$$

where $\mathbf{H}_{-(\ell-1)} = [(\mathbf{h}_1)_{-(\ell-1)} \ \ldots \ (\mathbf{h}_{n_\mathbf{H}})_{-(\ell-1)}]$ is a sub-matrix of $\mathbf{H}$, obtained by removing its first $\ell-1$ rows. The matrix $\mathbf{\Gamma}^{(\ell-1)} = [\gamma_{i,j}^{(\ell-1)}]$ of dimensions $(\ell-1)\times(n_\mathbf{H}-\ell+1)$ is termed the coefficient matrix.

Proof. From (5.20) and under the assumption $\mathbf{\Pi}_\ell = \mathbf{I}_{n_\mathbf{H}}, \forall \ell$, we have

$$\mathbf{h}_j^{(\ell-1)} = \left(\prod_{i=1}^{\ell-1} \left(\mathbf{I}_{m_\mathbf{H}} - \beta_i \mathbf{v}_i \mathbf{v}_i^\mathrm{T} \right) \right) \mathbf{h}_j, \ j = \ell, \ldots, n_\mathbf{H}. \tag{5.40}$$

Therefore, $\mathbf{h}_j^{(\ell-1)}$ ($j = \ell, \ldots, n_\mathbf{H}$) is a linear combination of $\mathbf{h}_j$ and the Householder vectors $\{\mathbf{v}_i, i = 1, \ldots, \ell-1\}$.

Additionally, a crucial property of the Householder transformation (see Remark 21) is

$$(\mathbf{v}_i)_{-i} = (\mathbf{h}_i^{(i-1)})_{-i}, \ i = 1, \ldots, \ell-1, \tag{5.41}$$

where $\mathbf{v}_{-i}$ denotes the vector obtained by removing the first i components of $\mathbf{v}$. Accordingly, and focusing on the vectors resulting from (5.41) after removing the first $\ell-1$

components of $\mathbf{v}_i$ and $\mathbf{h}_i^{(i-1)}$, respectively, we have

$$(\mathbf{v}_i)_{-(\ell-1)} = (\mathbf{h}_i^{(i-1)})_{-(\ell-1)},\ i = 1, \ldots, \ell-1. \tag{5.42}$$

Hence, the vector $(\mathbf{h}_j^{(\ell-1)})_{-(\ell-1)}$ $(j = \ell, \ldots, n_{\mathbf{H}})$, obtained by removing the first $\ell-1$ components of $\mathbf{h}_j^{(\ell-1)}$ in (5.40), can be expressed as a linear combination of $(\mathbf{h}_j)_{-(\ell-1)}$ and $\{(\mathbf{v}_i)_{-(\ell-1)} = (\mathbf{h}_i^{(i-1)})_{-(\ell-1)}, i = 1, \ldots, \ell-1\}$, i.e.,

$$(\mathbf{h}_j^{(\ell-1)})_{-(\ell-1)} = (\mathbf{h}_j)_{-(\ell-1)} + \sum_{i=1}^{\ell-1} (\mathbf{h}_i^{(i-1)})_{-(\ell-1)}\ \chi_{i,j}^{(\ell-1)},\quad j = \ell, \ldots, n_{\mathbf{H}}, \tag{5.43}$$

for some coefficients $\chi_{i,j}^{(\ell-1)}, i = 1, \ldots, \ell-1, j = \ell, \ldots, n_{\mathbf{H}}$.

Furthermore, notice that every vector $\mathbf{h}_i^{(i-1)}, i = 2, \ldots, \ell-1$, itself is a linear combination of $\mathbf{h}_i$ and the Householder vectors $\{\mathbf{v}_\eta, \eta = 1, \ldots, i-1\}$, i.e., $\mathbf{h}_i^{(i-1)} = \left(\prod_{\eta=1}^{i-1} \left(\mathbf{I}_{m_{\mathbf{H}}} - \beta_\eta \mathbf{v}_\eta \mathbf{v}_\eta^{\mathrm{T}}\right)\right)\mathbf{h}_i$ [see (5.40)], and recall that [see (5.42)]

$$(\mathbf{v}_\eta)_{-(\ell-1)} = (\mathbf{h}_\eta^{(\eta-1)})_{-(\ell-1)},\ \eta = 1, \ldots, i-1.$$

Thus, $(\mathbf{h}_i^{(i-1)})_{-(\ell-1)}$ $(i = 2, \ldots, \ell-1)$ can be expressed as a linear combination of $(\mathbf{h}_i)_{-(\ell-1)}$ and $\{(\mathbf{h}_\eta^{(\eta-1)})_{-(\ell-1)}, \eta = 1, \ldots, i-1\}$, i.e.,

$$(\mathbf{h}_i^{(i-1)})_{-(\ell-1)} = (\mathbf{h}_i)_{-(\ell-1)} + \sum_{\eta=1}^{i-1} (\mathbf{h}_\eta^{(\eta-1)})_{-(\ell-1)}\ \chi_{\eta,i}^{(i-1)},\quad i = 1, \ldots, \ell-1, \tag{5.44}$$

for some coefficients $\chi_{\eta,i}^{(i-1)}, i = 1, \ldots, \ell-1, \eta = 1, \ldots, i-1$.

Substituting (5.44) into (5.43), we therefore conclude by recursion that every vector $(\mathbf{h}_j^{(\ell-1)})_{-(\ell-1)}$, $j = \ell, \ldots, n_{\mathbf{H}}$, can be expressed as a linear combination of $(\mathbf{h}_j)_{-(\ell-1)}$ and $\{(\mathbf{h}_i)_{-(\ell-1)}, i = 1, \ldots, \ell-1\}$, i.e.,

$$(\mathbf{h}_j^{(\ell-1)})_{-(\ell-1)} = (\mathbf{h}_j)_{-(\ell-1)} + \sum_{i=1}^{\ell-1} (\mathbf{h}_i)_{-(\ell-1)}\ \gamma_{i,j}^{(\ell-1)},\quad j = \ell, \ldots, n_{\mathbf{H}}, \tag{5.45}$$

where the coefficients $\gamma_{i,j}^{(\ell-1)}, i = 1, \ldots, \ell-1, j = \ell, \ldots, n_{\mathbf{H}}$, need to be determined at each iteration. In what follows, we will provide a formula [see (5.56)] that updates $\gamma_{i,j}^{(\ell-1)}$ recursively.

Notice that the matrix $\mathbf{H}_{2,2}^{(\ell-1)}$ [see (5.21)] comprises all the vectors $(\mathbf{h}_j^{(\ell-1)})_{-(\ell-1)}$, $j = \ell, \ldots, n_{\mathbf{H}}$. Hence, (5.45) can be summarized into a compact matrix form, shown in (5.39). □

At every iteration $\ell-1$, the Standard Householder QR computes and stores the explicit form of $\mathbf{H}_{2,2}^{(\ell-1)}$. Similar to $\mathbf{\Lambda}_{2,2}^{(\ell-1)}$, $\mathbf{H}_{2,2}^{(\ell-1)}$ is a dense matrix and has $\mathcal{O}(\ell N-\ell^2/2)$ non-zeros per column. In order to preserve the original sparsity of $\mathbf{H}$, and at the same time reduce the memory usage, our modification to the Standard Householder QR is that the explicit form of $\mathbf{H}_{2,2}^{(\ell-1)}$ [or equivalently, the explicit forms of the vectors $(\mathbf{h}_j^{(\ell-1)})_{-(\ell-1)}, j=\ell,\ldots,n_{\mathbf{H}}$] is neither calculated nor stored. Instead, $\mathbf{H}_{2,2}^{(\ell-1)}$ is represented implicitly by the coefficient matrix $\mathbf{\Gamma}^{(\ell-1)}$, which (i) requires at most $\mathcal{O}(N^2)$ memory usage, and (ii) contains all information about $\mathbf{H}_{2,2}^{(\ell-1)}$ [see (5.39)]. Consequently, the Modified Householder QR sequentially transforms $\mathbf{H}$ through the evolution of $\mathbf{\Gamma}^{(\ell)}$. As will become clear later on, in contrast to updating the dense intermediate matrix $\mathbf{H}_{2,2}^{(\ell)}$ in the Standard Householder QR, updating $\mathbf{\Gamma}^{(\ell)}$ in the Modified Householder QR enables us to take full advantage of the original sparse pattern of $\mathbf{H}$ through $\mathbf{H}_{-\ell},\ \ell=1,\ldots,n_{\mathbf{H}}$ [see (5.39)].

In order to facilitate the presentation of the subsequent derivations, $\mathbf{\Gamma}^{(\ell-1)}$ is written as

$$\mathbf{\Gamma}^{(\ell-1)}=\begin{bmatrix}\boldsymbol{\gamma}_\ell^{(\ell-1)} & \overline{\mathbf{\Gamma}}^{(\ell-1)}\end{bmatrix}, \tag{5.46}$$

where $\boldsymbol{\gamma}_\ell^{(\ell-1)}$ denotes the first column of $\mathbf{\Gamma}^{(\ell-1)}$.

In what follows, we address two key issues in the Modified Householder QR. First, computing the matrices $\mathbf{H}_{1,1}^{(\ell)}$, $\mathbf{H}_{1,2}^{(\ell)}$ in (5.25); Second, deriving the recursive rule for updating $\mathbf{\Gamma}^{(\ell)}$ from $\mathbf{\Gamma}^{(\ell-1)}$. Or equivalently, we seek the expression of $\mathbf{\Gamma}^{(\ell)}$ such that

$$\mathbf{H}_{2,2}^{(\ell)}=\mathbf{H}_{-\ell}\begin{bmatrix}\mathbf{\Gamma}^{(\ell)}\\ \mathbf{I}_{n_{\mathbf{H}}-\ell}\end{bmatrix}, \tag{5.47}$$

where $\mathbf{H}_{-\ell}=[(\mathbf{h}_1)_{-\ell}\ \ldots\ (\mathbf{h}_{n_{\mathbf{H}}})_{-\ell}]$ is the sub-matrix of $\mathbf{H}$ obtained by removing its first ℓ rows.

To proceed, we first note that the vector $\mathbf{u}=(\mathbf{h}_\ell^{(\ell-1)})_{-(\ell-1)}$, i.e., the first column of $\mathbf{H}_{2,2}^{(\ell-1)}$, plays an important role in generating the Householder vector and the subsequent process. Hence, we compute the explicit form of $\mathbf{u}$ using (5.39), i.e.,

$$\mathbf{u}=(\mathbf{h}_\ell^{(\ell-1)})_{-(\ell-1)}=(\mathbf{h}_\ell)_{-(\ell-1)}+\mathbf{H}^1_{-(\ell-1)}\,\boldsymbol{\gamma}_\ell^{(\ell-1)}, \tag{5.48}$$

where $\mathbf{H}^1_{-(\ell-1)}=[(\mathbf{h}_1)_{-(\ell-1)}\ \ldots\ (\mathbf{h}_{\ell-1})_{-(\ell-1)}]$ consists of the first $\ell-1$ columns of $\mathbf{H}_{-(\ell-1)}$. Note that we explicitly compute only the vector $\mathbf{u}=(\mathbf{h}_\ell^{(\ell-1)})_{-(\ell-1)}$, the first

column of $\mathbf{H}_{2,2}^{(\ell-1)}$, while the remaining columns $(\mathbf{h}_j^{(\ell-1)})_{-(\ell-1)}, j = \ell+1, \ldots, n_{\mathbf{H}}$ [or equivalently, $\overline{\mathbf{H}}_{2,2}^{(\ell-1)}$ in (5.23)], are not explicitly computed. Instead, they are represented implicitly by $\overline{\mathbf{\Gamma}}^{(\ell-1)}$.

Once the expression of $\mathbf{u}$, computed by (5.48), is known, and c^* is obtained from column pivoting, the Householder vector $\mathbf{v}_\ell$ (or equivalently $\mathbf{v}$), as well as β_ℓ and r_ℓ, are readily available through (5.24) and (5.26). Notice that computing $\mathbf{v}$ from $\mathbf{u}$ only requires updating the first element of $\mathbf{u}$ [see Remark 21].

Now we are ready to present the recursive formulas for computing $\mathbf{H}_{1,1}^{(\ell)}$, $\mathbf{H}_{1,2}^{(\ell)}$, and $\mathbf{\Gamma}^{(\ell)}$.

Computing $\mathbf{H}_{1,1}^{(\ell)}$

Notice that the terms $\mathbf{H}_{1,1}^{(\ell-1)}$ as well as $\mathbf{t}$ are already available after the $(\ell-1)$th iteration of Householder QR. Hence the only unknown in $\mathbf{H}_{1,1}^{(\ell)}$ [see (5.25)] is the scalar r_ℓ, which can be calculated from c^* and u_1 in fixed number of arithmetic operations [see (5.26)].

Computing $\mathbf{H}_{1,2}^{(\ell)}$

Since $\overline{\mathbf{H}}_{1,2}^{(\ell-1)}$ becomes available after the $(\ell-1)$th iteration of Householder QR, updating $\mathbf{H}_{1,2}^{(\ell)}$ is equivalent to calculating the vector $\mathbf{s}$ from (5.27), which requires $\bar{\mathbf{s}}$ and $\boldsymbol{\delta} = \beta_\ell \big(\overline{\mathbf{H}}_{2,2}^{(\ell-1)}\big)^{\mathrm{T}} \mathbf{v}$. Remember that we do not have the explicit forms of $\bar{\mathbf{s}}$ and $\overline{\mathbf{H}}_{2,2}^{(\ell-1)}$. However, using the fact that $\bar{\mathbf{s}}^{\mathrm{T}}$ corresponds to the first row of $\overline{\mathbf{H}}_{2,2}^{(\ell-1)}$ [see (5.23)] and based on (5.39) and (5.46), we can rewrite $\overline{\mathbf{H}}_{2,2}^{(\ell-1)}$ and $\bar{\mathbf{s}}$ as follows

$$\overline{\mathbf{H}}_{2,2}^{(\ell-1)} = \mathbf{H}_{-(\ell-1)}^{2} + \mathbf{H}_{-(\ell-1)}^{1}\, \overline{\mathbf{\Gamma}}^{(\ell-1)}, \tag{5.49}$$

$$\bar{\mathbf{s}} = \boldsymbol{\rho}_2 + \big(\overline{\mathbf{\Gamma}}^{(\ell-1)}\big)^{\mathrm{T}} \boldsymbol{\rho}_1, \tag{5.50}$$

where $\mathbf{H}_{-(\ell-1)}^{2} = [(\mathbf{h}_{\ell+1})_{-(\ell-1)} \ \ldots \ (\mathbf{h}_{n_{\mathbf{H}}})_{-(\ell-1)}]$ comprises the last $n_{\mathbf{H}} - \ell$ columns of $\mathbf{H}_{-(\ell-1)}$, and $\boldsymbol{\rho}_1^{\mathrm{T}}$ and $\boldsymbol{\rho}_2^{\mathrm{T}}$ are the first rows of $\mathbf{H}_{-(\ell-1)}^{1}$ and $\mathbf{H}_{-(\ell-1)}^{2}$, respectively. From (5.49), we compute the vector

$$\boldsymbol{\delta} = \beta_\ell \big(\overline{\mathbf{H}}_{2,2}^{(\ell-1)}\big)^{\mathrm{T}} \mathbf{v} = \boldsymbol{\delta}_2 + \big(\overline{\mathbf{\Gamma}}^{(\ell-1)}\big)^{\mathrm{T}} \boldsymbol{\delta}_1, \tag{5.51}$$

where $\boldsymbol{\delta}_1 = \beta_\ell \big(\mathbf{H}_{-(\ell-1)}^{1}\big)^{\mathrm{T}} \mathbf{v}$ and $\boldsymbol{\delta}_2 = \beta_\ell \big(\mathbf{H}_{-(\ell-1)}^{2}\big)^{\mathrm{T}} \mathbf{v}$.

Substituting (5.50) and (5.51) in (5.27), we arrive at the following update equation for $\mathbf{s}$ (or equivalently, $\mathbf{H}_{1,2}^{(\ell)}$), i.e.,

$$\mathbf{s} = \big[\boldsymbol{\rho}_2 - v_1\boldsymbol{\delta}_2\big] + \big(\overline{\boldsymbol{\Gamma}}^{(\ell-1)}\big)^{\mathrm{T}}\big[\boldsymbol{\rho}_1 - v_1\boldsymbol{\delta}_1\big]. \tag{5.52}$$

Computing $\boldsymbol{\Gamma}^{(\ell)}$

To determine $\boldsymbol{\Gamma}^{(\ell)}$, we begin with (5.47) and (5.28). Recall that we do not have the explicit expression of $\underline{\mathbf{H}}_{2,2}^{(\ell-1)}$. However, since $\underline{\mathbf{H}}_{2,2}^{(\ell-1)}$ corresponds to removing the first row of $\overline{\mathbf{H}}_{2,2}^{(\ell-1)}$ [see (5.23)], from (5.49) we obtain

$$\underline{\mathbf{H}}_{2,2}^{(\ell-1)} = \mathbf{H}_{-\ell}^{2} + \mathbf{H}_{-\ell}^{1}\,\overline{\boldsymbol{\Gamma}}^{(\ell-1)}, \tag{5.53}$$

where $\mathbf{H}_{-\ell}^{1} = [(\mathbf{h}_1)_{-\ell}\ \ldots\ (\mathbf{h}_{\ell-1})_{-\ell}]$ and $\mathbf{H}_{-\ell}^{2} = [(\mathbf{h}_{\ell+1})_{-\ell}\ \ldots\ (\mathbf{h}_{n_{\mathbf{H}}})_{-\ell}]$.

Next we explore the property $\mathbf{v}_{-1} = \mathbf{u}_{-1}$ [see Remark 21]. In particular, based on (5.48), we have

$$\mathbf{v}_{-1} = \mathbf{u}_{-1} = (\mathbf{h}_\ell)_{-\ell} + \mathbf{H}_{-\ell}^{1}\boldsymbol{\gamma}_\ell^{(\ell-1)}. \tag{5.54}$$

Finally, we substitute (5.53), (5.54), and (5.51) into (5.28), and notice that $\mathbf{H}_{-\ell} = [\mathbf{H}_{-\ell}^{1}\ \ (\mathbf{h}_\ell)_{-\ell}\ \ \mathbf{H}_{-\ell}^{2}]$, to arrive at

$$\mathbf{H}_{2,2}^{(\ell)} = \mathbf{H}_{-\ell}^{1}\big(\overline{\boldsymbol{\Gamma}}^{(\ell-1)} - \boldsymbol{\gamma}_\ell^{(\ell-1)}\boldsymbol{\delta}^{\mathrm{T}}\big) - (\mathbf{h}_\ell)_{-\ell}\,\boldsymbol{\delta}^{\mathrm{T}} + \mathbf{H}_{-\ell}^{2} = \mathbf{H}_{-\ell}\begin{bmatrix}\overline{\boldsymbol{\Gamma}}^{(\ell-1)} - \boldsymbol{\gamma}_\ell^{(\ell-1)}\boldsymbol{\delta}^{\mathrm{T}} \\ -\boldsymbol{\delta}^{\mathrm{T}} \\ \mathbf{I}_{n_{\mathbf{H}}-\ell}\end{bmatrix}. \tag{5.55}$$

Comparing (5.55) and (5.47), we immediately obtain the expression of the $\ell \times (n_{\mathbf{H}} - \ell)$ matrix $\boldsymbol{\Gamma}^{(\ell)}$, i.e.,

$$\boldsymbol{\Gamma}^{(\ell)} = \begin{bmatrix}\overline{\boldsymbol{\Gamma}}^{(\ell-1)} - \boldsymbol{\gamma}_\ell^{(\ell-1)}\boldsymbol{\delta}^{\mathrm{T}} \\ -\boldsymbol{\delta}^{\mathrm{T}}\end{bmatrix}. \tag{5.56}$$

Note that the upper part of $\boldsymbol{\Gamma}^{(\ell)}$ is a rank-one modification of the existing matrix $\overline{\boldsymbol{\Gamma}}^{(\ell-1)}$, which has a relatively low time complexity. Furthermore, (5.56) affirms that every vector $(\mathbf{h}_j^{(\ell)})_{-\ell}, j = \ell+1, \ldots, n_{\mathbf{H}}$, is indeed a linear combination of $(\mathbf{h}_j)_{-\ell}$ and $\{(\mathbf{h}_i)_{-\ell}, i = 1, \ldots, \ell\}$.

In summary, we outline the algorithmic flow chart of the Modified Householder QR in Algorithm 6. Note that column pivoting implemented by the Modified Householder QR is identical to that of the Standard Householder QR.

Algorithm 6 Modified Householder QR

Require: $\mathbf{H}^{(0)} = \mathbf{H}$ of dimensions $m_{\mathbf{H}} \times n_{\mathbf{H}}$ ($m_{\mathbf{H}} \geq n_{\mathbf{H}}$).

Ensure: $\mathbf{R}_{\mathbf{H}}$ of dimensions $l_{\mathbf{H}} \times n_{\mathbf{H}}$ and $\mathbf{\Pi}_{\mathbf{H}}$ of dimensions $n_{\mathbf{H}} \times n_{\mathbf{H}}$.

Initialize $\mathbf{\Gamma}^{(0)}$ as an empty matrix.
for $i = 1$ **to** $n_{\mathbf{H}}$, **do**
    Initialize $\pi_i = i$, and compute c_i, the squared norm of the ith column of $\mathbf{H}^{(0)}$.
end for
Set $\ell = 1$, determine $\ell^* = \arg\max_j \{c_j,\, j = \ell, \ldots, n_{\mathbf{H}}\}$ and $c^* = c_{\ell^*}$.
while $c^* > \epsilon$ **do**
    Swap the ℓth and ℓ^*th columns of $\mathbf{H}^{(\ell-1)}$, as well as the first and $(\ell^* - \ell + 1)$th columns of $\mathbf{\Gamma}^{(\ell-1)}$, exchange c_ℓ and c_{ℓ^*}, and then interchange π_ℓ with π_{ℓ^*}.
    Calculate $\mathbf{u}$ from (5.48).
    Compute $\mathbf{v}, \beta_\ell, r_\ell$ and update the sub-matrix $\mathbf{H}_{1,1}^{(\ell)}$.
    Determine $\boldsymbol{\delta}_1, \boldsymbol{\delta}_2$, and $\boldsymbol{\delta}$ using (5.51).
    Calculate $\mathbf{s} = [s_1 \ \ldots \ s_{n_{\mathbf{H}}-\ell}]^{\mathrm{T}}$ from (5.52) and update the sub-matrix $\mathbf{H}_{1,2}^{(\ell)}$.
    Update $\mathbf{\Gamma}^{(\ell)}$ based on (5.56).
    for $j = \ell + 1$ **to** $n_{\mathbf{H}}$, **do**
        Modify $c_j \Leftarrow c_j - s_{j-\ell}^2$.
    end for
    Seek $\ell^* = \arg\max_j \{c_j,\, j = \ell + 1, \ldots, n_{\mathbf{H}}\}$ and $c^* = c_{\ell^*}$, and then $\ell \Leftarrow \ell + 1$.
end while
return $\mathbf{\Pi}_{\mathbf{H}} = [\mathbf{e}_{\pi_1} \ \ldots \ \mathbf{e}_{\pi_{n_{\mathbf{H}}}}]$, and $\mathbf{R}_{\mathbf{H}}$, selected as the first $l_{\mathbf{H}}$ rows of $\mathbf{H}^{(l_{\mathbf{H}})}$, where the total number of iterations $l_{\mathbf{H}} = \mathbf{rank}(\mathbf{H})$.

5.5.2 Complexity analysis of the Modified Householder QR

In Section 5.5.1, we provide a general description of the Modified Householder QR algorithm which is applicable to arbitrary matrices. In this section, we apply the Modified Householder QR to perform the QR factorization of $\mathbf{H}$ [see (5.6)], and analyze its computational complexity by taking into account the sparsity of $\mathbf{H}$.

Since column pivoting is implemented identically by both the Standard and Modified Householder QR algorithms, we conclude that the computational overhead associated with column pivoting in the Modified Householder QR has time complexity of $\mathcal{O}(N^2)$ and space complexity of $\mathcal{O}(N)$ (see Proposition 23). Therefore, in what follows, we analyze the computational complexity associated with the Householder update in the Modified Householder QR, corresponding to Lines 8–12 in Algorithm 6. We first investigate the time complexity, followed by the space complexity.

Time complexity

In this section, we briefly analyze the time complexity associated with the Householder update when applying the Modified Householder QR to factorize $\mathbf{H}$. We claim that its worst-case time complexity is of $\mathcal{O}(N^3)$. To prove this, we will identify the number of flops for every line of Lines 8–12 in Algorithm 6, and show that the total number of arithmetic operations required per iteration of the Modified Householder QR is bounded above by $\mathcal{O}(N^2)$. Since the total number of iterations is $l_{\mathbf{H}} = 3(N-1)$, the overall time complexity is $\mathcal{O}(N^3)$. For clarity, Table 5.1 lists the cost for each step between Line 8 and 12.

• Computational cost of Line 8:

Recall that $\mathbf{nnz}(\mathbf{h}_j) \sim \mathcal{O}(N), j = 1, \ldots, 3N$ [see Remark 18], hence $\mathbf{nnz}((\mathbf{h}_i)_{-(\ell-1)})$ $\sim \mathcal{O}(N)$, $i = 1, \ldots, \ell$. Thus, computing $\mathbf{u}$ from (5.48) requires $\mathcal{O}(\ell N)$ operations. Therefore, the cost of performing Line 8 is bounded above by $\mathcal{O}(N^2)$.

• Computational cost of Line 9:

Once $\mathbf{u}$ becomes available from Line 8, as well as $c^* = \mathbf{u}^{\mathrm{T}}\mathbf{u}$ obtained from column pivoting, determining the scalars β_ℓ and r_ℓ require constant number of arithmetic operations. Furthermore, updating $\mathbf{v}$ from $\mathbf{u}$ only requires the modification of the first element of $\mathbf{u}$, which can be achieved in $\mathcal{O}(1)$ process time. In summary, the cost of

Table 5.1: Time complexity per iteration of the Modified Householder QR

Algorithm 6	Variables	Process Time
Line 8	$\mathbf{u}$	$\mathcal{O}(N^2)$
Line 9	$\beta_\ell, r_\ell, \mathbf{v}$	$\mathcal{O}(1)$
Line 10	$\boldsymbol{\delta}$	$\mathcal{O}(N^2)$
Line 11	$\mathbf{s}$	$\mathcal{O}(N^2)$
Line 12	$\boldsymbol{\Gamma}^{(\ell)}$	$\mathcal{O}(N^2)$

performing Line 9 is of $\mathcal{O}(1)$.

• Computational cost of Line 10:

Since each column of $\mathbf{H}^1_{-(\ell-1)}$ and $\mathbf{H}^2_{-(\ell-1)}$ has $\mathcal{O}(N)$ non-zeros, which is attained by preserving the original sparse structure of $\mathbf{H}$, computing $\boldsymbol{\delta}_1$ and $\boldsymbol{\delta}_2$ has a cost at most of $\mathcal{O}(N^2)$, regardless of the dense structure of $\mathbf{v}$. Additionally, calculating $\boldsymbol{\delta}$ from $\boldsymbol{\delta}_1$ and $\boldsymbol{\delta}_2$ has a cost of $\mathcal{O}(\ell N)$, since (5.51) involves an $(n_{\mathbf{H}}-\ell)\times(\ell-1)$ matrix multiplied by an $(\ell-1)\times 1$ vector and a vector-vector addition of dimension $n_{\mathbf{H}}-\ell$. In summary, the cost of performing Line 10 is of $\mathcal{O}(N^2)$.

• Computational cost of Line 11:

Since computing $\mathbf{s}$ involves a $(\ell-1)\times 1$ vector multiplying with an $(n_{\mathbf{H}}-\ell)\times(\ell-1)$ matrix and a vector-vector addition of dimension $n_{\mathbf{H}}-\ell$ [see (5.52)], the overall cost of performing Line 11 is bounded above by $\mathcal{O}(N^2)$.

• Computational cost of Line 12:

In (5.56), the vector $\boldsymbol{\delta}$ [see (5.51)] is available from Line 10 and does not need to be recomputed. Hence, we only need to focus on the upper part of $\boldsymbol{\Gamma}^{(\ell)}$ [see (5.56)], which is a rank-one update of the existing matrix $\overline{\boldsymbol{\Gamma}}^{(\ell-1)}$. Since the vectors $\boldsymbol{\gamma}_\ell^{(\ell-1)}$ and $\boldsymbol{\delta}$ are of dimensions $\ell-1$ and $n_{\mathbf{H}}-\ell$, respectively, we conclude that the overall cost of performing Line 12 is of $\mathcal{O}(\ell N)$, which is bounded above by $\mathcal{O}(N^2)$.

In summary, we have shown that the number of arithmetic operations required per Householder update is bounded above by $\mathcal{O}(N^2)$ in the Modified Householder QR. Therefore the worst-case time complexity of the QR factorization of $\mathbf{H}$, when employing the Modified Householder QR algorithm, is of $\mathcal{O}(N^3)$.

Space complexity

To address the space complexity, we focus on the storage requirements for the matrices $\mathbf{H}$, $\mathbf{\Gamma}^{(\ell)}, \ell = 1, \ldots, n_{\mathbf{H}}$, and the vector $\mathbf{u}$, which are the most dominant sources in terms of memory cell usage. Recall that from Remark 18 $\mathbf{nnz}(\mathbf{H}) \sim \mathcal{O}(N^2)$. Additionally, notice that the dimension of $\mathbf{u}$ is at most $m_{\mathbf{H}} \sim \mathcal{O}(N^2)$. On the other hand, since the dimensions of $\mathbf{\Gamma}^{(\ell)}$ are $\ell \times (n_{\mathbf{H}} - \ell)$, thus $\max\{\mathbf{nnz}(\mathbf{\Gamma}^{(\ell)}); \ell = 1, \ldots, n_{\mathbf{H}}\} \sim \mathcal{O}(N^2)$. Therefore, we conclude that the space complexity of the QR decomposition of $\mathbf{H}$, when using the Modified Householder QR algorithm, is $\mathcal{O}(N^2)$.

Summary of complexity analysis

In summary, we have shown that the QR factorization of $\mathbf{H}$ when employing the Modified Householder QR has time complexity of $\mathcal{O}(N^3)$ and space complexity of $\mathcal{O}(N^2)$. Therefore, we conclude that the overall time and space complexity of EKF-based CL through QR factorization are $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2)$, respectively, one order of magnitude less than that of using the Standard Householder QR. In conclusion, we summarize the main result of this section in Theorem 28.

Theorem 28. *The EKF-based CL through QR factorization, when employing the Modified Householder QR, has time complexity $\mathcal{O}(N^3)$ and space complexity $\mathcal{O}(N^2)$.*

Furthermore, we would like to point out that the Modified Householder QR algorithm is applicable to any sparse matrix $\mathbf{H}$ of sparsity pattern other than that of the measurement matrix in (5.6). In particular, suppose that the dimensions of $\mathbf{H}$ are $m \times n$ ($m \geq n$), and denote $\tau = \max(\mathbf{nnz}(\mathbf{h}_1), \ldots, \mathbf{nnz}(\mathbf{h}_n))$. Following the analysis conducted in Section 5.5.2, it can be shown that the space complexity of the Modified Householder QR is of $\mathcal{O}(\max\{\min\{\tau n, m\}, n^2\})$, and the time complexity is of $\mathcal{O}(\max\{\tau n^2, n^3\})$, in contrast to $\mathcal{O}(mn^2)$ of the Standard Householder QR [105].

5.6 Simulation results

In the previous section, we have shown that the worst-case time and space complexity of the QR decomposition of the sparse measurement matrix $\mathbf{H}$ in CL [see (5.6)], when employing the Modified Householder QR, is of $\mathcal{O}(N^3)$ and $\mathcal{O}(N^2)$, respectively. In order

to corroborate our theoretical analysis, we have evaluated the running time required by the Modified Householder QR algorithm for a team of N robots performing CL. Specifically, we randomly generated the robots' poses and assumed that each robot is able to detect, identify, and measure both relative distance and bearing to the remaining $N-1$ robots. Hence, $\mathbf{H}$ has dimensions $2(N-1)N \times 3N$.

We have examined the scalability of our algorithm by varying N from 3 to 1000, and for every value of N, we have conducted 120 simulations. We count the CPU running time for a complete QR decomposition [see (5.13)] when employing the Modified Householder QR algorithm (MH-QR). The average running times are summarized in Table 5.2, as well as in Figure 5.2. Furthermore, we compared our results with the CPU running time when employing SuiteSparseQR (SS-QR) [128, 129], the current state-of-the-art QR decomposition package for sparse matrices, which is an implementation of the multi-frontal sparse QR factorization algorithm. All simulations were run on a Linux (kernel 2.6.32) desktop computer with a 2.66 GHz Intel Core-i5 Quadcore CPU and 4 GB of RAM.

The results presented in Table 5.2 and Figure 5.2 illustrate that MH-QR significantly outperforms SS-QR for teams of robots ranging between 3 and 500. Additionally, for large teams of robots (i.e., $N > 500$), we were unable to perform QR decomposition using SS-QR due to memory shortage. In contrast, MH-QR is applicable even when the number of robots increases to 1000, and it successfully performs the QR factorization of $\mathbf{H}$, whose dimensions are about 2 million by 3 thousand, in less than 6 minutes. Additionally, after performing linear regression on the data, we have determined that the logarithm of the average CPU running time required by SS-QR or MH-QR is

$$\begin{aligned} \log t_{\text{SS-QR}} &= 3.854 \times \log N - 18.529, \\ \log t_{\text{MH-QR}} &= 3.240 \times \log N - 16.607. \end{aligned} \tag{5.57}$$

As evident from Figure 5.2 and (5.57), MH-QR has cubic time complexity in the number of robots. We should note that the main reason for the linear coefficient in (5.57) deviating from its ideal value 3 is because as N increases, a significant portion of CPU resources is devoted to memory accessing. One of our future research directions is to further improve the performance of MH-QR and investigate its distributed and parallel implementations.

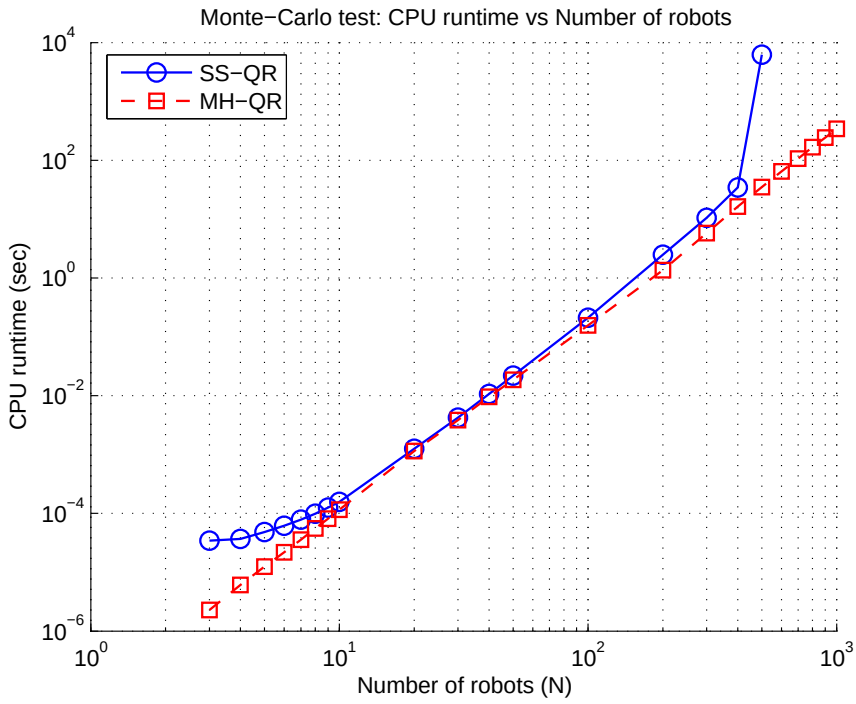


Figure 5.2: [Monte Carlo simulations] Average CPU runtime of QR decomposition of $\mathbf{H}$ over 120 trials. Comparison between SuiteSparseQR (SS-QR) and the Modified Householder QR (MH-QR).

Table 5.2: CPU runtime (sec)

N	SS-QR	MH-QR
3	3.4478×10^{-5}	2.2894×10^{-6}
4	3.6947×10^{-5}	6.0830×10^{-6}
5	4.8356×10^{-5}	1.2463×10^{-5}
6	6.1540×10^{-5}	2.1875×10^{-5}
7	7.8234×10^{-5}	3.5728×10^{-5}
8	9.8992×10^{-5}	5.5599×10^{-5}
9	1.2526×10^{-4}	8.1345×10^{-5}
10	1.5649×10^{-4}	1.1524×10^{-4}
20	1.2550×10^{-3}	1.1416×10^{-3}
30	4.2463×10^{-3}	3.8503×10^{-3}
40	1.0732×10^{-2}	9.5553×10^{-3}
50	2.1999×10^{-2}	1.8594×10^{-2}
100	2.1057×10^{-1}	1.5690×10^{-1}
200	2.4867×10^{0}	1.3535×10^{0}
300	1.0515×10^{1}	5.7690×10^{0}
400	3.4640×10^{1}	1.6350×10^{1}
500	6.2230×10^{3}	3.4965×10^{1}
600	N/A	6.4883×10^{1}
700	N/A	1.0704×10^{2}
800	N/A	1.6659×10^{2}
900	N/A	2.4353×10^{2}
1000	N/A	3.4350×10^{2}

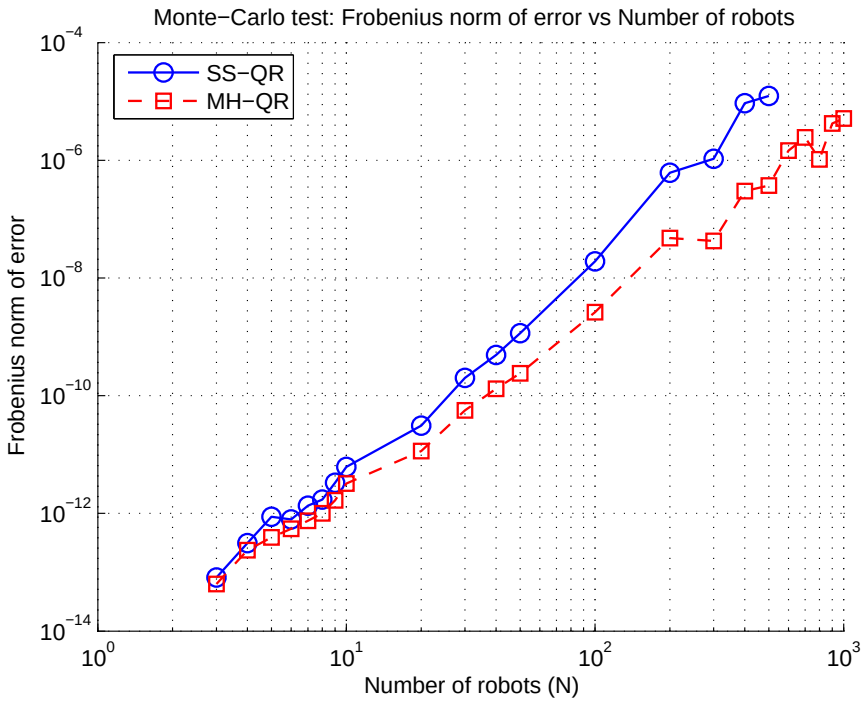


Figure 5.3: [Monte Carlo simulations] Average Frobenius norm of $\mathbf{H}^{\mathrm{T}}\mathbf{H} - \mathbf{R}_{\mathbf{H}}^{\mathrm{T}}\mathbf{R}_{\mathbf{H}}$ over 120 trials. Comparison between SuiteSparseQR (SS-QR) and the Modified Householder QR (MH-QR).

Table 5.3: Frobenius norm of $\mathbf{H}^{\mathrm{T}}\mathbf{H} - \mathbf{R}_{\mathbf{H}}^{\mathrm{T}}\mathbf{R}_{\mathbf{H}}$

N	SS-QR	MH-QR
3	8.1188×10^{-14}	6.2962×10^{-14}
4	3.1230×10^{-13}	2.3758×10^{-13}
5	8.7691×10^{-13}	3.9107×10^{-13}
6	7.8818×10^{-13}	5.4605×10^{-13}
7	1.3542×10^{-12}	7.4759×10^{-13}
8	1.7164×10^{-12}	1.0002×10^{-12}
9	3.3196×10^{-12}	1.6660×10^{-12}
10	6.1658×10^{-12}	3.2150×10^{-12}
20	3.1072×10^{-11}	1.1443×10^{-11}
30	2.0070×10^{-10}	5.6430×10^{-11}
40	4.9004×10^{-10}	1.3051×10^{-10}
50	1.1505×10^{-9}	2.4006×10^{-10}
100	1.9087×10^{-8}	2.6235×10^{-9}
200	6.1207×10^{-7}	4.7511×10^{-8}
300	1.0625×10^{-6}	4.2499×10^{-8}
400	9.3274×10^{-6}	3.0049×10^{-7}
500	1.2407×10^{-5}	3.7327×10^{-7}
600	N/A	1.4607×10^{-6}
700	N/A	2.4464×10^{-6}
800	N/A	1.0321×10^{-6}
900	N/A	4.2254×10^{-6}
1000	N/A	5.1291×10^{-6}

Finally, we have examined the accuracy of MH-QR. In particular, we computed the Frobenius norm of $\mathbf{H}^{\mathrm{T}}\mathbf{H} - \mathbf{R}_{\mathbf{H}}^{\mathrm{T}}\mathbf{R}_{\mathbf{H}}$, which is 0 in the ideal case. We have compared the Frobenius norms when employing SS-QR and MH-QR, averaged over 120 simulations for each N. As evident from Table 5.3 and Figure 5.3, MH-QR attains higher numerical accuracy than SS-QR, which is attributed to column pivoting implemented in MH-QR.

5.7 Summary

In this chapter, we have developed an efficient algorithm for the QR decomposition of the sparse measurement matrix $\mathbf{H}$ arising in CL, namely the Modified Householder QR. The proposed algorithm has been successfully applied to 2-D multi-robot CL. In particular, we have shown that the overall computational complexity of the EKF update through QR factorization, implemented using the Modified Householder QR, is of $\mathcal{O}(N^3)$ in time complexity and $\mathcal{O}(N^2)$ in space complexity. Simulation results demonstrate that the Modified Householder QR algorithm achieves higher numerical accuracy and significantly outperforms SuiteSparseQR in terms of CPU runtime for large teams of robots.

Chapter 6

Concluding Remarks

6.1 Summary of contributions

The work presented in the preceding chapters has focused on providing novel active sensing algorithms for the challenging problems of (i) leader-follower formation control; (ii) target tracking; and on reducing the processing requirements of the EKF-based CL. The key contributions of our work are summarized as follows:

- Active formation control:
  Chapter 2 presented a new approach for maximizing the localization accuracy of a follower robot moving in a leader-follower formation by optimally controlling the follower robot's sensing locations. Specifically, since the follower robot's pose uncertainty increases when it maintains an exact formation, in this work we allow the follower robot to deviate from its desired formation, but confine it within a predefined region. Our objective is to determine the optimal follower robot's sensing location at the next time step in order to minimize the weighted trace of the posterior covariance matrix characterizing the follower robot's pose uncertainty. We developed an algorithm that analytically computes the globally optimal solution (that is, the follower robot's best sensing location), even though the underlying constrained optimization problem is non-convex. The simulation results demonstrate substantial improvement of the follower robot's localization accuracy when employing our optimal motion strategy as compared to alternative

approaches.

- Active target tracking:
  Chapters 3 & 4 introduced efficient algorithms for computing the sensing locations (or trajectories) of a single or multiple tracking sensors that maximize the target's position estimation accuracy.

  Specifically, in Chapter 3, we provided analytical expressions for computing the next best sensing location (i.e., one-step-ahead optimization) for a single sensor, subject to mobility constraints, tracking a moving target using distance-only, bearing-only, or distance-and-bearing observations. We also addressed the problem of determining optimal one-step-ahead motion strategies for multiple heterogeneous sensors, and showed that by imposing maximum-speed constraints on the sensors, the resulting non-convex constrained optimization problem is, in general, NP-Hard. In order to provide an efficient, real-time solution for the multi-sensor case, we relaxed the original NP-Hard problem and proposed an iterative, linear complexity, algorithm, that leverages the single-sensor analytical solution, to find an approximate solution to the original problem.

  In Chapter 4, we addressed the problem of generating an optimal trajectory over a finite time horizon (i.e., multi-step-ahead optimization) for a single sensor, under mobility constraints, tracking a moving target using distance-only observations. Unlike the one-step-ahead optimization, our objective here is to minimize the uncertainty of the target's position estimates at the end of a finite time interval. Moreover, in the problem formulation, we explicitly consider the increasing uncertainty about the predicted target-position estimates. To solve the multi-step-ahead optimization problem, we developed an efficient algorithm based on cyclic coordinate descent, whose computational complexity is significantly lower compared to that of a grid-based exhaustive search method (quadratic vs. exponential in the number of time steps considered).

- Computational complexity reduction of the EKF-based CL:
  In Chapter 5, we focused on improving the localization accuracy of multiple robots

by performing CL. One of the key challenges of CL is its high processing requirements (quartic in the number of robots), which can overwhelm the limited computational resources of a real-time system. In this work, we developed an algorithm that reduces the overall processing cost of CL by performing dimensionality reduction of the measurement equations through QR factorization on the measurement matrix $\mathbf{H}$. Furthermore, we developed an efficient algorithm, termed the Modified Householder QR, that fully exploits the sparsity of $\mathbf{H}$, and reduces the cost of the QR factorization of $\mathbf{H}$ by an order of magnitude as compared to traditional QR algorithms. As a result, we proved that the worst-case computational complexity of CL, when employing the Modified Householder QR, reduces from quartic to cubic.

It is our belief that the performance gains realized by the efficient active sensing algorithms we developed for the problems of leader-follower formation control and target tracking, will inspire and motivate further research on other other important active sensing problems in robotics, such as calibration, exploration, etc.

6.2 Future research directions

The one-step-ahead optimal motion strategy for a follower robot in leader-follower formations described in Chapter 2 significantly improves localization accuracy over existing methods. At the same time, several problems still remain open. Of particular interest is that of optimizing the follower's motion over multiple time steps, which has the potential to further reduce its positioning uncertainty. However, carrying out optimization over multiple time steps is quite challenging since it requires solving polynomial systems (which correspond to the KKT optimality conditions) of higher order in a large number of unknowns. To this end, recently-developed fast numerical polynomial solvers based on the action matrix [130] or the Macaulay resultant [130] offer promising directions for addressing such complex problems.

The work on active target tracking opens a number of avenues of future research. The most straightforward extension (of multi-step-ahead optimization in Chapter 4) is to investigate multi-step-ahead single-sensor active target tracking using (i) bearing-only

or (ii) distance-and-bearing observations. Furthermore, one unexplored (and interesting) question in our work is the active tracking of multiple targets. Multi-target tracking has many potential applications (e.g., in environmental monitoring) where the number of objects being tracked is often larger than the number of sampling sensors. Developing efficient algorithms for single (or multiple) sensor multi-target active tracking has the potential to significantly reduce the operational cost of such tasks. Additionally, the fundamental issue of assessing the computational complexity of multi-target active tracking deserves further investigation.

In Chapter 5, we focused on reducing the computational complexity of the EKF-based CL, while implicitly assuming that both the communication bandwidth and the sensors' communication range are sufficiently large. In reality, however, such perfect communication (in terms of links and bandwidth) may not be always feasible. For example, strong signal attenuation due to the environment may significantly reduce the communication bandwidth. Furthermore, it is very difficult to maintain a constant communication topology in a mobile sensor network due to the mobility of the robots and obstacles in their surroundings. Recently, several efficient estimation algorithms have been proposed that can handle communication bandwidth limitations. The key idea behind these approaches is to quantize and transmit sensor observations using only one or few bits (see [131, 132, 133]). On the other hand, Leung *et al.* [114] have recently proposed an approach to deal with asynchronous communication. The idea is to transmit all available data stored in each sensor to its neighbors, and postpone the (complete) measurement update, till all relevant information from all robots becomes available.[1] These novel methodologies for addressing the issues of limited communication bandwidth and fast-changing communication graphs, can also be used for performing multi-robot active sensing and distributed CL.

In our current work, we have assumed that the robots and targets move in 2-D. This assumption is valid when robots operate in man-made environments, such as indoors or on flat roads. Indeed, most research in robotics assumes that the robots move on planar surfaces. However, the world we live in is 3-dimensional, and in increasingly many

[1] Note that intermediate estimates for the system's state are also available in real time (causal estimator) based only on the measurements locally available to each sensor. However, complete, i.e., based on all measurements, estimates are only available with some delay that depends on the communication range and bandwidth of the sensors [134].

robotics applications, the planar motion assumption can no longer be used to describe the motion of robots and targets that operate underwater, fly in the air, or navigate in hilly roads or over tough terrain. In these cases, it is necessary to consider the full 3-D pose of the robots as well as estimate the 3-D location of the target. Therefore, it remains an interesting topic of future research to extend the proposed active formation control and active target tracking algorithms to 3-D.

References

[1] J. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active vision. *International Journal of Computer Vision*, 1(4):333–356, Jan. 1988.

[2] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, Aug. 1988.

[3] P. J. Burt. Smart sensing within a pyramid vision machine. *Proceedings of the IEEE*, 76(8):1006–1015, Aug. 1988.

[4] D. H. Ballard. Animate vision. *Artificial Intelligence*, 48(1):57–86, Feb. 1991.

[5] D. H. Ballard and C. M. Brown. Principles of animate vision. *CVGIP: Image Understanding*, 56(1):3–21, Jul. 1992.

[6] R. Bajcsy and M. Campos. Active and exploratory perception. *CVGIP: Image Understanding*, 56(1):31–40, Jul. 1992.

[7] D. T. Wilkes and J. K. Tsotsos. Active object recognition. In *Proceedings of the 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 136–141, Champaign, IL, Jun. 15–18 1992.

[8] F. G. Callari and F. P. Ferrie. Active recognition: using uncertainty to reduce ambiguity. In *Proceedings of the 13th International Conference on Pattern Recognition*, volume 1, pages 925–929, Vienna , Austria, Aug. 25–29 1996.

[9] S. J. Dickinson, H. I. Christensen, J. K. Tsotsos, and G. Olofsson. Active object recognition integrating attention and viewpoint control. *Computer Vision and Image Understanding*, 67(3):239–260, Sep. 1997.

[10] D. Paulus, C. Drexler, M. Reinhold, M. Zobel, and J. Denzler. Active computer vision system. In *Proceedings of the 5th IEEE International Workshop on Computer Architectures for Machine Perception*, pages 18–27, Padova, Italy, Sep. 11–13 2000.

[11] J. Denzler and C. M. Brown. Information theoretic sensor data selection for active object recognition and state estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):145–157, Feb. 2002.

[12] M. A. Sipe and D. Casasent. Feature space trajectory methods for active computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1634–1643, Dec. 2002.

[13] K. N. Kutulakos and C. R. Dyer. Recovering shape by purposive viewpoint adjustment. *International Journal of Computer Vision*, 12(2–3):113–136, Apr. 1994.

[14] E. Marchand and F. Chaumette. Active vision for complete scene reconstruction and exploration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(1):65–72, Jan. 1999.

[15] E. Marchand and F. Chaumette. An autonomous active vision system for complete and accurate 3D scene reconstruction. *International Journal of Computer Vision*, 32(3):171–194, Aug. 1999.

[16] J. Haupt, R. Nowak, and R. Castro. Adaptive sensing for sparse signal recovery. In *Proceedings of the 13th IEEE Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop*, pages 702–707, Marco Island, FL, Jan. 4–7 2009.

[17] D. L. Hecht, S. Uma, R. Matusiak, R. Kowalski, and E. Shrader. Optical beam position active sensing and control using acoustooptic satellite beams. In *Proceedings of the 2003 IEEE Ultrasonics Symposium*, volume 1, pages 507–512, Honolulu, HI, Oct. 5–8 2003.

[18] T. Nakamoto, H. Ishida, and T. Moriizumi. Active odor sensing system. In *Proceedings of the IEEE International Symposium on Industrial Electronics*, volume 1, pages SS128–SS133, Guimaraes, Portugal, Jul. 7–11 1997.

[19] Y. L. Hsieh, S. H. Song, and Q. T. Zhang. Active sensing for cognitive radio. In *Proceedings of the 2009 IEEE 70th Vehicular Technology Conference Fall*, pages 1–5, Anchorage, AK, Sep. 20–23 2009.

[20] S. Liu and L. E. Holloway. Active sensing policies for stochastic systems. *IEEE Transactions on Automatic Control*, 47(2):373–377, Feb. 2002.

[21] K. Patel, W. Macklem, S. Thrun, and M. Montemerlo. Active sensing for high-speed offroad driving. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3162–3168, Barcelona, Spain, Apr. 18–22 2005.

[22] R. Sukthankar, D. Pomerleau, and C. Thorpe. Panacea: an active sensor controller for the ALVINN autonomous driving system. Technical Report CMU-RI-TR-93-09, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Apr. 1993.

[23] T. Arici and Y. Altunbasak. Adaptive sensing for environment monitoring using wireless sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, volume 4, pages 2347–2352, Atlanta, GA, Mar. 21–25 2004.

[24] V. Singhvi, A. Krause, C. Guestrin, J. H. Garrett, and H. S. Matthews. Intelligent light control using sensor networks. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pages 218–229, San Diego, CA, Nov. 2–4 2005. Association for Computing Machinery.

[25] L. Mihaylova, T. Lefebvre, H. Bruyninckx, K. Gadeyne, and J. De Schutter. Active sensing for robotics–a survey. In *Proceedings of the 5th International Conference on Numerical Methods and Applications*, Borovets, Bulgaria, Aug. 20–24 2002.

[26] L. Mihaylova, H. Bruyninckx, and J. De Schutter. Active sensing of a nonholonomic wheeled mobile robot. In *Proceedings of the IEEE Benelux Signal Processing Symposium*, pages 125–128, Leuven, Belgium, Mar. 21–22 2002.

[27] L. Mihaylova, T. Lefebvre, H. Bruyninckx, K. Gadeyne, and J. De Schutter. A comparison of decision making criteria and optimization methods for active robotic sensing. In I. Dimov, I. Lirkov, S. Margenov, and Z. Zlatev, editors, *Numerical*

Methods and Applications, volume 2542 of *Lecture Notes in Computer Science*, pages 316–324. Springer, Berlin, Germany, 2003.

[28] V. Isler and R. Bajcsy. The sensor selection problem for bounded uncertainty sensing models. *IEEE Transactions on Automation Science and Engineering*, 3(4):372–381, Oct. 2006.

[29] C. Kreuchera, K. Kastellab, and A. O. Hero. Sensor management using an active sensing approach. *Signal Processing*, 85(3):607–624, Mar. 2005.

[30] A. I. Mourikis and S. I. Roumeliotis. Optimal sensor scheduling for resource-constrained localization of mobile robot formations. *IEEE Transactions on Robotics*, 22(5):917–931, Oct. 2006.

[31] Y. He and E. K. P. Chong. Sensor scheduling for target tracking: a Monte Carlo sampling approach. *Digital Signal Processing*, 16(5):533–545, Sep. 2006.

[32] C. Kreuchera, D. Blatt, A. O. Hero, and K. Kastellab. Adaptive multi-modality sensor scheduling for detection and tracking of smart targets. *Digital Signal Processing*, 16(5):546–567, Sep. 2006.

[33] C. M. Kreucher, A. O. Hero, K. D. Kastella, and M. R. Morel. An information based approach to sensor management in large dynamic networks. *Proceedings of the IEEE*, 95(5):978–999, May 2007.

[34] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in Gaussian processes: theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, Feb. 2008.

[35] A. Singh, A. Krause, C. Guestrin, and W. Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34:707–755, 2009.

[36] A. Arsénio and M. I. Ribeiro. Active range sensing for mobile robot localization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1066–1071, Victoria, Canada, Oct. 13–17 1998.

[37] D. Fox, W. Burgard, and S. Thrun. Active Markov localization for mobile robots. *Robotics and Autonomous Systems*, 25(3–4):195–207, Nov. 1998.

[38] A. C. Murtra, J. M. M. Tur, and A. Sanfeliu. Efficient active global localization for mobile robots operating in large and cooperative environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2758–2763, Pasadena, CA, May 19–23 2008.

[39] A. C. Murtra, J. M. M. Tur, and A. Sanfeliu. Action evaluation for mobile robot global localization in cooperative environments. *Robotics and Autonomous Systems*, 56(10):807–818, Oct. 2008.

[40] R. Kurazume and S. Hirose. Study on cooperative positioning system: optimum moving strategies for CPS-III. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 2896–2903, Leuven, Belgium, May 16–20 1998.

[41] N. Trawny and T. Barfoot. Optimized motion strategies for cooperative localization of mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pages 1027–1032, New Orleans, LA, Apr. 26–May 1 2004.

[42] H. J. S. Feder, J. J. Leonard, and C. M. Smith. Adaptive mobile robot navigation and mapping. *The International Journal of Robotics Research*, 18(7):650–668, Jul. 1999.

[43] C. Leung, S. Huang, and G. Dissanayake. Active SLAM using model predictive control and attractor based exploration. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5031, Beijing, China, Oct. 9–15 2006.

[44] T. Kollar and N. Roy. Trajectory optimization using reinforcement learning for map exploration. *The International Journal of Robotics Research*, 27(2):175–196, Feb. 2008.

[45] V. N. Christopoulos and S. I. Roumeliotis. Adaptive sensing for instantaneous gas release parameter estimation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4450–4456, Barcelona, Spain, Apr. 18–22 2005.

[46] V. N. Christopoulos and S. I. Roumeliotis. Multi robot trajectory generation for single source explosion parameter estimation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2803–2809, Barcelona, Spain, Apr. 18–22 2005.

[47] N. E. Leonard, D. A. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. E. Davis. Collective motion, sensor networks, and ocean sampling. *Proceedings of the IEEE*, 95(1):48–74, Jan. 2007.

[48] J. P. Le Cadre. Optimization of the observer motion for bearings-only target motion analysis. In *Proceedings of the 36th IEEE Conference on Decision and Control*, volume 4, pages 3126–3131, San Diego, CA, Dec. 10–12 1997.

[49] J. P. Le Cadre and S. Laurent-Michel. Optimizing the receiver maneuvers for bearings-only tracking. *Automatica*, 35(4):591–606, Apr. 1999.

[50] J. M. Passerieux and D. Van Cappel. Optimal observer maneuver for bearings-only tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 34(3):777–788, Jul. 1998.

[51] S. Martínez and F. Bullo. Optimal sensor placement and motion coordination for target tracking. *Automatica*, 42(4):661–668, Apr. 2006.

[52] Y. Oshman and P. Davidson. Optimization of observer trajectories for bearings-only target localization. *IEEE Transactions on Aerospace and Electronic Systems*, 35(3):892–902, Jul. 1999.

[53] R. Olfati-Saber. Distributed tracking for mobile sensor networks with information-driven mobility. In *Proceedings of the American Control Conference*, pages 4606–4612, New York, NY, Jul. 11–13 2007.

[54] T. H. Chung, J. W. Burdick, and R. M. Murray. A decentralized motion coordination strategy for dynamic target tracking. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2416–2422, Orlando, FL, May 15–19 2006.

[55] P. Yang, R. A. Freeman, and K. M. Lynch. Distributed cooperative active sensing using consensus filters. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 405–410, Rome, Italy, Apr. 10–14 2007.

[56] E. Stump, V. Kumar, B. Grocholsky, and P. M. Shiroma. Control for localization of targets using range-only sensors. *The International Journal of Robotics Research*, 28(6):743–757, 2009.

[57] A. Logothetis, A. Isaksson, and R. J. Evans. An information theoretic approach to observer path design for bearings-only tracking. In *Proceedings of the 36th IEEE Conference on Decision and Control*, volume 4, pages 3132–3137, San Diego, CA, Dec. 10–12 1997.

[58] E. W. Frew. *Trajectory design for target motion estimation using monocular vision.* PhD thesis, Stanford University, Stanford, CA, Aug. 2003.

[59] A. W. Stroupe and T. Balch. Value-based action selection for observation with robot teams using probabilistic techniques. *Robotics and Autonomous Systems*, 50(2–3):85–97, Feb. 2005.

[60] G. L. Mariottini, G. Pappas, D. Prattichizzo, and K. Daniilidis. Vision-based localization of leader-follower formations. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 635–640, Seville, Spain, Dec. 12–15 2005.

[61] G. L. Mariottini, S. Martini, and M. B. Egerstedt. A switching active sensing strategy to maintain observability for vision-based formation control. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2637–2642, Kobe, Japan, May 12–17 2009.

[62] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation.* Wiley-Interscience, John Wiley & Sons Inc., New York, NY, 2001.

[63] S. S. Blackman and R. F. Popoli. *Design and Analysis of Modern Tracking Systems*. Artech House Radar Library. Artech House Publishers, Boston, MA, 1999.

[64] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 2nd edition, 1999.

[65] J. G. Bender. An overview of systems studies of automated highway systems. *IEEE Transactions on Vehicular Technology*, 40(1):82–99, Feb. 1991.

[66] J. R. Spletzer, A. K. Das, R. Fierro, C. J. Taylor, V. Kumar, and J. P. Ostrowski. Cooperative localization and control for multi-robot manipulation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 631–636, Maui, HI, Oct. 29– Nov. 3 2001.

[67] T. Balch and R. C. Arkin. Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, 14(6):926–939, Dec. 1998.

[68] K.-H. Tan and M. A. Lewis. Virtual structures for high-precision cooperative mobile robotic control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 132–139, Osaka, Japan, Nov. 4–8 1996.

[69] J. P. Desai, J. Ostrowski, and V. Kumar. Controlling formations of multiple mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 2864–2869, Leuven, Belgium, May 16–20 1998.

[70] I. M. Rekleitis, G. Dudek, and E. E. Milios. Multi-robot cooperative localization: a study of trade-offs between efficiency and accuracy. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2690–2695, Lausanne, Switzerland, Sep. 30– Oct. 4 2002.

[71] X. S. Zhou and S. I. Roumeliotis. Robot-to-robot relative pose estimation from range measurements. *IEEE Transactions on Robotics*, 24(6):1379–1393, Dec. 2008.

[72] X. S. Zhou and S. I. Roumeliotis. Determining the robot-to-robot relative pose using range-only measurements. Technical Report TR-2007-001, Department of

Computer Science and Engineering, University of Minnesota, Minneapolis, MN, May 2007.

[73] X. S. Zhou, K. X. Zhou, and S. I. Roumeliotis. Optimized motion strategies for localization in leader-follower formations. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 98–105, San Francisco, CA, Sep. 25–30 2011.

[74] F. Zhang, B. Grocholsky, and V. Kumar. Formations for localization of robot networks. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 3369–3374, New Orleans, LA, Apr. 26–May 1 2004.

[75] Y. S. Hidaka, A. I. Mourikis, and S. I. Roumeliotis. Optimal formations for cooperative localization of mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4126–4131, Barcelona, Spain, Apr. 18–22 2005.

[76] D. A. Cox, J. B. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Undergraduate Texts in Mathematics. Springer, New York, NY, 3rd edition, 2007.

[77] A. Edelman and H. Murakami. Polynomial roots from companion matrix eigenvalues. *Mathematics of Computation*, 64(210):763–776, Apr. 1995.

[78] J. R. Polastre. Design and implementation of wireless sensor networks for habitat monitoring. Master's thesis, University of California, Berkeley, Berkeley, CA, May 2003.

[79] L. E. Parker, B. Birch, and C. Reardon. Indoor target intercept using an acoustic sensor network and dual wavefront path planning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 278–283, Las Vegas, NV, Oct. 27–31 2003.

[80] R. Bodor, R. Morlok, and N. Papanikolopoulos. Dual-camera system for multi-level activity recognition. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 643–648, Sendai, Japan, Sep. 28–Oct. 2 2004.

[81] J. Pineau, M. Montemerlo, M. Pollack, N. Roy, and S. Thrun. Towards robotic assistants in nursing homes: challenges and results. *Robotics and Autonomous Systems*, 42(3–4):271–281, Mar. 2003.

[82] G. M. Siouris, G. Chen, and J. Wang. Tracking an incoming ballistic missile using an extended interval Kalman filter. *IEEE Transactions on Aerospace and Electronic Systems*, 33(1):232–240, Jan. 1997.

[83] B. Jung and G. S. Sukhatme. Tracking targets using multiple robots: the effect of environment occlusion. *Autonomous Robots*, 13(3):191–205, Nov. 2002.

[84] R. Vidal, O. Shakernia, H. J. Kim, D. H. Shim, and S. Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE Transactions on Robotics and Automation*, 18(5):662–669, Oct. 2002.

[85] K. Zhou and S. I. Roumeliotis. Multirobot active target tracking with combinations of relative observations. *IEEE Transactions on Robotics*, 27(4):678–695, Aug. 2011.

[86] K. Zhou and S. I. Roumeliotis. Optimal motion strategies for range-only constrained multisensor target tracking. *IEEE Transactions on Robotics*, 24(5):1168–1185, Oct. 2008.

[87] K. X. Zhou and S. I. Roumeliotis. Multi-robot active target tracking with distance and bearing observations. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2209–2216, St. Louis, MO, Oct. 10–15 2009.

[88] K. X. Zhou and S. I. Roumeliotis. Optimal motion strategies for range-only distributed target tracking. In *Proceedings of the American Control Conference*, pages 5195–5200, Minneapolis, MN, Jun. 14–16 2006.

[89] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, New York, NY, 1990.

[90] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Books in the Mathematical Sciences. W. H. Freeman, San Francisco, CA, 1979.

[91] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Belmont, MA, 1997.

[92] A. O. Hero, D. A. Castanón, D. Cochran, and K. Kastella, editors. *Foundations and Applications of Sensor Management*. Signals and Communication Technology. Springer, New York, NY, 2008.

[93] E. K. P. Chong, C. M. Kreucher, and A. O. Hero. Partially observable Markov decision process approximations for adaptive sensing. *Discrete Event Dynamic Systems*, 19(3):377–422, Sep. 2009.

[94] C. Kreucher, A. O. Hero, K. Kastella, and D. Chang. Efficient methods of non-myopic sensor management for multitarget tracking. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, volume 1, pages 722–727, Atlantis, Bahamas, Dec. 14–17 2004.

[95] Y. He and E. K. P. Chong. Sensor scheduling for target tracking in sensor networks. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, volume 1, pages 743–748, Atlantis, Bahamas, Dec. 14–17 2004.

[96] Y. Li, L. W. Krakow, E. K. P. Chong, and K. N. Groomn. Dynamic sensor management for multisensor multitarget tracking. In *2006 40th Annual Conference on Information Sciences and Systems*, pages 1397–1402, Princeton, NJ, Mar. 22–24 2006.

[97] Y. Li, L. W. Krakow, E. K. P. Chong, and K. N. Groomn. Approximate stochastic dynamic programming for sensor scheduling to track multiple targets. *Digital Signal Processing*, 19(6):978–989, Dec. 2009.

[98] L. W. Krakow, E. K. P. Chong, K. N. Groomn, J. Harrington, Y. Li, and B. Rigdon. Control of perimeter surveillance wireless sensor networks via partially observable Marcov decision process. In *Proceedings 2006 40th Annual IEEE International Carnahan Conferences Security Technology*, pages 261–268, Lexington, KY, Oct. 16–19 2006.

[99] S. Ji, R. Parr, and L. Carin. Nonmyopic multiaspect sensing with partially observable Markov decision processes. *IEEE Transactions on Signal Processing*, 55(6):2720–2730, Jun. 2007.

[100] A. Logothetis, A. Isaksson, and R. J. Evans. Comparison of suboptimal strategies for optimal own-ship maneuvers in bearings-only tracking. In *Proceedings of the American Control Conference*, volume 6, pages 3334–3338, Philadelphia, PA, Jun. 24–26 1998.

[101] M. L. Hernandez. Optimal sensor trajectories in bearings-only. In *Proceedings of the Seventh International Conference on Information Fusion*, pages 893–900, Stockholm, Sweden, Jun. 28–Jul. 1 2004.

[102] A. S. Chhetri, D. Morrell, and A. Papandreou-Suppappola. Nonmyopic sensor scheduling and its efficient implementation for target tracking applications. *EURASIP Journal on Applied Signal Processing*, 2006(31520):1–18, Apr. 2006.

[103] A. H. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 64 of *Mathematics in Science and Engineering*. Academic Press, New York, NY, 1970.

[104] G. P. Huang, K. Zhou, N. Trawny, and S. I. Roumeliotis. A bank of maximum a posteriori estimation algorithm for single-sensor range-only target tracking. Technical Report TR-2009-010, Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, Oct. 2009.

[105] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in Mathematical Sciences. The Johns Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

[106] S. I. Roumeliotis and G. Bekey. Distributed multirobot localization. *IEEE Transactions on Robotics and Automation*, 18(5):781–795, Oct. 2002.

[107] A. I. Mourikis and S. I. Roumeliotis. Performance analysis of multirobot cooperative localization. *IEEE Transactions on Robotics*, 22(4):666–681, Aug. 2006.

[108] R. Kurazume, S. Nagata, and S. Hirose. Cooperative positioning with multiple robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1250–1257, San Diego, CA, May 8–13 1994.

[109] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. A probabilistic approach to collaborative multi-robot localization. *Autonomous Robots*, 8(3):325–344, Jun. 2000.

[110] S. I. Roumeliotis and I. Rekleitis. Propagation of uncertainty in cooperative multirobot localization: analysis and experimental results. *Autonomous Robots*, 17(1):41–54, Jul. 2004.

[111] N. Trawny, S. I. Roumeliotis, and G. B. Giannakis. Cooperative multi-robot localization under communication constraints. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4394–4400, Kobe, Japan, May 12–17 2009.

[112] A. Howard, M. J. Matarić, and G. S. Sukhatme. Localization for mobile robot teams using maximum likelihood estimation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 434–459, Lausanne, Switzerland, Sep. 30–Oct. 5 2002.

[113] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu. Decentralized localization for dynamic and sparse robot networks. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3135–3141, Kobe, Japan, May 12–17 2009.

[114] K. Y. K. Leung, T. D. Barfoot, and H. H. T. Liu. Decentralized localization of sparsely-communicating robot networks: a centralized-equivalent approach. *IEEE Transactions on Robotics*, 26(1):62–77, Feb. 2010.

[115] A. Howard, M. J. Matarić, and G. S. Sukhatme. Localization for mobile robot teams: a distributed MLE approach. In B. Siciliano and P. Dario, editors,

Experimental Robotics VIII: The 8th International Symposium on Experimental Robotics, volume 5 of *Springer Tracts in Advanced Robotics*, pages 146–155. Springer-Verlag, Berlin, German, 2003.

[116] E. D. Nerurkar, S. I. Roumeliotis, and A. Martinelli. Distributed maximum a posteriori estimation for multi-robot cooperative localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1402–1409, Kobe, Japan, May 12–17 2009.

[117] F. Dellaert, F. Alegre, and E. B. Martinson. Intrinsic localization and mapping with 2 applications: diffusion mapping and Macro Polo localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 2344–2349, Taipei, Taiwan, Sep. 14–19 2003.

[118] S. Panzieri, F. Pascucci, and R. Setola. Multirobot localisation using Interlaced extended Kalman filter. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2816–2821, Beijing, China, Oct. 9–15 2006.

[119] L. Glielmo, R. Setola, and F. Vasca. An interlaced extended Kalman filter. *IEEE Transactions on Automatic Control*, 44(8):1546–1549, Aug. 1999.

[120] N. Karam, F. Chausse, R. Aufrere, and R. Chapuis. Localization of a group of communicating vehicles by state exchange. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 519–524, Beijing, China, Oct. 9–15 2006.

[121] A. Martinelli. Improving the precision on multi robot localization by using a series of filters hierarchically distributed. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1053–1058, San Diego, CA, Oct. 29–Nov. 2 2007.

[122] G. P. Huang, N. Trawny, A. I. Mourikis, and S. I. Roumeliotis. Observability-based consistent EKF estimators for multi-robot cooperative localization. *Autonomous Robots*, 30(1):99–122, Jan. 2011.

[123] P. S. Maybeck. *Stochastic models, estimation and control*, volume 1. Academic Press, New York, NY, 1979.

[124] D. S. Bayard and P. B. Brugarolas. On-board vision-based spacecraft estimation algorithm for small body exploration. *IEEE Transactions on Aerospace and Electronic Systems*, 44(1):243–260, Jan. 2008.

[125] P. A. Businger and G. H. Golub. Linear least squares solutions by Householder transformations. *Numerische Mathematik*, 7(3):269–276, Jun. 1965.

[126] L. Kaufman. Application of dense Householder transformations to a sparse matrix. *ACM Transactions on Mathematical Software*, 5(4):442–450, Dec. 1979.

[127] K. X. Zhou and S. I. Roumeliotis. A sparsity-aware QR decomposition algorithm for efficient cooperative localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 799–806, St. Paul, MN, May 14–18 2012.

[128] T. A. Davis. Algorithm 915, SuiteSparseQR: multifrontal multithreaded rank-revealing sparse QR factorization. *ACM Transactions on Mathematical Software*, 38(1), Nov. 2011.

[129] T. A. Davis. *Direct Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2006.

[130] D. A. Cox, J. B. Little, and D. O'Shea. *Using Algebraic Geometry*. Graduate Texts in Mathematics. Springer, New York, NY, 2nd edition, 2005.

[131] A. Ribeiro, G. B. Giannakis, and S. I. Roumeliotis. SOI-KF: distributed Kalman filtering with low-cost communications using the sign of innovations. *IEEE Transactions on Signal Processing*, 54(12):4782–4795, Dec. 2006.

[132] E. J. Msechu, S. I. Roumeliotis, A. Ribeiro, and G. B. Giannakis. Decentralized quantized Kalman filtering with scalable communication cost. *IEEE Transactions on Signal Processing*, 56(8):3727–3741, Aug. 2008.

[133] E. D. Nerurkar, K. X. Zhou, and S. I. Roumeliotis. A hybrid estimation framework for cooperative localization under communication constraints. In *Proceedings of*

the IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 502–509, San Francisco, CA, Sep. 25–30 2011.

[134] E. D. Nerurkar and S. I. Roumeliotis. Asynchronous multi-centralized cooperative localization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4352–4359, Taipei, Taiwan, Oct. 18–22 2010.

Appendix A

Appendices for Chapter 3

A.1 Rational functions $f_0(\mathbf{s}^*)$ and $\nabla f_0(\mathbf{s}^*)$

We hereafter show that $f_0(\mathbf{s}^*)$ [see (3.49)] is a rational function in $\mathbf{s}^* = [x\ y]^{\mathrm{T}}$, i.e.,

$$f_0(\mathbf{s}^*) = \frac{h(\mathbf{s}^*)}{g(\mathbf{s}^*)}, \tag{A.1}$$

where $h(\mathbf{s}^*)$ and $g(\mathbf{s}^*)$ are polynomials in x and y, and thus

$$\nabla f_0(\mathbf{s}^*) = \frac{g(\mathbf{s}^*)\nabla h(\mathbf{s}^*) - h(\mathbf{s}^*)\nabla g(\mathbf{s}^*)}{g^2(\mathbf{s}^*)} \tag{A.2}$$

is also a rational function in x and y.

To proceed, note that $f_0(\mathbf{s}^*) = \mathrm{tr}(\mathbf{M}^{-1})$ where

$$\mathbf{M} = \mathbf{\Lambda} + \frac{\kappa_d}{\sigma_d^2}\frac{(\mathbf{s}^*)\,(\mathbf{s}^*)^{\mathrm{T}}}{(\mathbf{s}^*)^{\mathrm{T}}\,(\mathbf{s}^*)} + \frac{\kappa_\theta}{\sigma_\theta^2}\frac{\mathbf{J}\,(\mathbf{s}^*)\,(\mathbf{s}^*)^{\mathrm{T}}\,\mathbf{J}^{\mathrm{T}}}{\left((\mathbf{s}^*)^{\mathrm{T}}\,(\mathbf{s}^*)\right)^2}. \tag{A.3}$$

Moreover, $\mathrm{tr}(\mathbf{M}^{-1}) = \frac{\mathrm{tr}(\mathbf{M})\cdot\varepsilon}{\det(\mathbf{M})\cdot\varepsilon}$ holds true for any nonzero scalar ε and any invertible 2×2 matrix $\mathbf{M}$. By defining $\varepsilon = \lambda_1\lambda_2(x^2+y^2)^2 > 0$, we obtain [see (A.1) and (A.3)]:

$$\begin{aligned}
\mathrm{tr}\,(\mathbf{M}) &= \lambda_1^{-1} + \lambda_2^{-1} + \kappa_d\sigma_d^{-2} + \kappa_\theta\sigma_\theta^{-2}(x^2+y^2)^{-1},\\
\det(\mathbf{M}) &= \lambda_1^{-1}\lambda_2^{-1} + \kappa_d\sigma_d^{-2}\kappa_\theta\sigma_\theta^{-2}(x^2+y^2)^{-1}\\
&\quad + \lambda_1^{-1}\left[\kappa_d\sigma_d^{-2}(x^2+y^2)^{-1}y^2 + \kappa_\theta\sigma_\theta^{-2}(x^2+y^2)^{-2}x^2\right]\\
&\quad + \lambda_2^{-1}\left[\kappa_d\sigma_d^{-2}(x^2+y^2)^{-1}x^2 + \kappa_\theta\sigma_\theta^{-2}(x^2+y^2)^{-2}y^2\right],\\
h(\mathbf{s}^*) &= \mathrm{tr}\,(\mathbf{M})\cdot\varepsilon = a_2(x^2+y^2)^2 + a_1(x^2+y^2),
\end{aligned}$$

and

$$\begin{aligned} g(\mathbf{s}^*) &= \det(\mathbf{M}) \cdot \varepsilon \\ &= b_6(x^2+y^2)^2 + b_5(x^2+y^2)x^2 + b_4(x^2+y^2)y^2 + b_3(x^2+y^2) + b_2x^2 + b_1y^2, \end{aligned}$$

where the coefficients a_i $(i = 1, 2)$ and b_j $(j = 1, \ldots, 6)$ are expressed in terms of $\lambda_1, \lambda_2, \kappa_d\sigma_d^{-2}$, and $\kappa_\theta\sigma_\theta^{-2}$:

$$\begin{aligned} a_2 = \lambda_1 + \lambda_2 + \lambda_1\lambda_2\kappa_d\sigma_d^{-2}, \quad a_1 = \lambda_1\lambda_2\kappa_\theta\sigma_\theta^{-2}, \\ b_6 = 1, \quad b_5 = \lambda_1\kappa_d\sigma_d^{-2}, \quad b_4 = \lambda_2\kappa_d\sigma_d^{-2}, \\ b_3 = \lambda_1\lambda_2\kappa_d\sigma_d^{-2}\kappa_\theta\sigma_\theta^{-2}, \quad b_2 = \lambda_2\kappa_\theta\sigma_\theta^{-2}, \quad b_1 = \lambda_1\kappa_\theta\sigma_\theta^{-2}, \end{aligned} \tag{A.4}$$

and thus $f_0(\mathbf{s}^*) = \frac{h(\mathbf{s}^*)}{g(\mathbf{s}^*)}$ is a rational function in $\mathbf{s}^*$.

Since both $h(\mathbf{s}^*)$ and $g(\mathbf{s}^*)$ are bivariate polynomials, their derivatives $\nabla h(\mathbf{s}^*)$ and $\nabla g(\mathbf{s}^*)$ are 2×1 vectors with each component a bivariate polynomial. Specifically,

$$\nabla h(\mathbf{s}^*) = 2h_0(\mathbf{s}^*) \begin{bmatrix} x \\ y \end{bmatrix}, \tag{A.5}$$

$$\nabla g(\mathbf{s}^*) = 2g_0(\mathbf{s}^*) \begin{bmatrix} x \\ y \end{bmatrix} + 2 \begin{bmatrix} xg_x(\mathbf{s}^*) \\ yg_y(\mathbf{s}^*) \end{bmatrix}, \tag{A.6}$$

where $h_0(\mathbf{s}^*), g_0(\mathbf{s}^*), g_x(\mathbf{s}^*)$, and $g_y(\mathbf{s}^*)$ are polynomials in x and y, defined as:

$$h_0(\mathbf{s}^*) = 2a_2(x^2+y^2) + a_1, \tag{A.7}$$

$$g_0(\mathbf{s}^*) = 2b_6(x^2+y^2) + b_5x^2 + b_4y^2 + b_3,$$

$$g_x(\mathbf{s}^*) = b_5(x^2+y^2) + b_2, \tag{A.8}$$

$$g_y(\mathbf{s}^*) = b_4(x^2+y^2) + b_1. \tag{A.9}$$

A.2 Transforming (3.58) into (3.59)

By substituting the expressions of $\nabla f_0(\mathbf{s}^*)$, $\nabla h(\mathbf{s}^*)$, and $\nabla g(\mathbf{s}^*)$ [see (A.2)-(A.6)] into (3.58), and setting the numerator equal to zero, we obtain the following polynomial equation with respect to x and y:

$$\begin{aligned} 0 = \; & x(y-c_2)\big[g(\mathbf{s}^*)h_0(\mathbf{s}^*) - h(\mathbf{s}^*)g_0(\mathbf{s}^*) - h(\mathbf{s}^*)g_x(\mathbf{s}^*)\big] \\ & - y(x-c_1)\big[g(\mathbf{s}^*)h_0(\mathbf{s}^*) - h(\mathbf{s}^*)g_0(\mathbf{s}^*) - h(\mathbf{s}^*)g_y(\mathbf{s}^*)\big]. \end{aligned} \tag{A.10}$$

Rearranging terms in (A.10), yields:

$$\begin{aligned} 0 = \; & xyh(\mathbf{s}^*)\big[g_y(\mathbf{s}^*) - g_x(\mathbf{s}^*)\big] + h(\mathbf{s}^*)\big[c_2 x g_x(\mathbf{s}^*) - c_1 y g_y(\mathbf{s}^*)\big] \\ & + (c_1 y - c_2 x)\big[g(\mathbf{s}^*)h_0(\mathbf{s}^*) - h(\mathbf{s}^*)g_0(\mathbf{s}^*)\big]. \end{aligned} \tag{A.11}$$

Substituting the expressions of $h(\mathbf{s}^*)$, $g(\mathbf{s}^*)$, $h_0(\mathbf{s}^*)$, $g_0(\mathbf{s}^*)$, $g_x(\mathbf{s}^*)$, $g_y(\mathbf{s}^*)$, and rearranging terms in (A.11), we obtain the polynomial equation (3.59):

$$\begin{aligned} 0 = f_1(x,y) = \; & \beta_3 xy\Delta^3 + (\alpha_8 x + \alpha_7 y + \beta_2)xy\Delta^2 \\ & + (\alpha_6 x^3 + \alpha_5 x^2 y + \alpha_4 xy^2 + \alpha_3 y^3 + \beta_1 xy)\Delta + (\alpha_2 x + \alpha_1 y)xy, \end{aligned}$$

where $\Delta := x^2 + y^2$, and

$$\begin{aligned} \beta_3 &= a_2(b_4 - b_5), \\ \beta_2 &= a_1(b_4 - b_5) + a_2(b_1 - b_2), \\ \beta_1 &= a_1(b_1 - b_2), \\ \alpha_8 &= c_1 a_2(b_5 - b_4), \\ \alpha_7 &= c_2 a_2(b_5 - b_4), \\ \alpha_6 &= c_2\left[a_1(b_5 + b_6) - a_2(b_2 + b_3)\right], \\ \alpha_5 &= c_1\left[a_2(b_3 + 2b_2 - b_1) - a_1(b_6 + b_4)\right], \\ \alpha_4 &= c_2\left[a_2(b_2 - 2b_1 - b_3) + a_1(b_6 + b_5)\right], \\ \alpha_3 &= c_1\left[a_2(b_3 + b_1) - a_1(b_6 + b_4)\right], \\ \alpha_2 &= c_1 a_1(b_2 - b_1), \\ \alpha_1 &= c_2 a_1(b_2 - b_1). \end{aligned} \tag{A.12}$$

Since $a_1, a_2, b_1, \ldots, b_6$ [see (A.4)] are coefficients expressed in terms of $\lambda_1, \lambda_2, \kappa_d\sigma_d^{-2}$, and $\kappa_\theta\sigma_\theta^{-2}$, from (A.12) we conclude that $\beta_i,\ i = 1, 2, 3$, and $\alpha_j,\ j = 1, \ldots, 8$, are also functions of $\lambda_1, \lambda_2, c_1, c_2, \kappa_d\sigma_d^{-2}$, and $\kappa_\theta\sigma_\theta^{-2}$.

A.3 Coefficients of (3.59) for the case $\kappa_d = \kappa_\theta = 1, \lambda_1 > \lambda_2$

By substituting $\kappa_d = \kappa_\theta = 1$ into the expressions of $a_i, i = 1, 2$, and $b_j, j = 1, \ldots, 6$, [see (A.4)], we have:

$$\begin{aligned} a_2 = \lambda_1 + \lambda_2 + \lambda_1\lambda_2\sigma_d^{-2}, \quad a_1 = \lambda_1\lambda_2\sigma_\theta^{-2}, \\ b_6 = 1, \quad b_5 = \lambda_1\sigma_d^{-2}, \quad b_4 = \lambda_2\sigma_d^{-2}, \\ b_3 = \lambda_1\lambda_2\sigma_d^{-2}\sigma_\theta^{-2}, \quad b_2 = \lambda_2\sigma_\theta^{-2}, \quad b_1 = \lambda_1\sigma_\theta^{-2}. \end{aligned}$$

Thus, from (A.12), we conclude that in general $\beta_i \neq 0, i = 1, 2, 3$, and $\alpha_j \neq 0, j = 1, \ldots, 8$. Additionally, $\beta_3 = a_2(b_4 - g_5) = (\lambda_2 - \lambda_1)\sigma_d^{-2}a_2 < 0$, since $a_2 > 0$ and $\lambda_1 > \lambda_2$. Hence, f_1 [see (3.59)] is an 8th-order bivariate polynomial.

Moreover, by comparing the coefficients of (3.59) and (3.60), it is evident that χ_i, $i = 0, \ldots, 7$, are polynomials with respect to x, whose coefficients are functions of $\lambda_1, \lambda_2, c_1, c_2, \sigma_d^{-2}$, and σ_θ^{-2} [see (A.4) and (A.12)], i.e.,

$$\begin{aligned} \chi_7 &= \beta_3 x, \\ \chi_6 &= \alpha_7 x, \\ \chi_5 &= 3\beta_3 x^3 + \alpha_8 x^2 + \beta_2 x + \alpha_3, \\ \chi_4 &= 2\alpha_7 x^3 + \alpha_4 x, \\ \chi_3 &= 3\beta_3 x^5 + 2\alpha_8 x^4 + 2\beta_2 x^3 + (\alpha_5 + \alpha_3)x^2 + \beta_1 x, \\ \chi_2 &= \alpha_7 x^5 + (\alpha_6 + \alpha_4)x^3 + \alpha_1 x, \\ \chi_1 &= \beta_3 x^7 + \alpha_8 x^6 + \beta_2 x^5 + \alpha_5 x^4 + \beta_1 x^3 + \alpha_2 x^2, \\ \chi_0 &= \alpha_6 x^5. \end{aligned}$$

A.4 Coefficients of (3.59) for the case $\kappa_d = 0, \kappa_\theta = 1, \lambda_1 > \lambda_2$

Substituting $\kappa_d = 0, \kappa_\theta = 1$ into the expressions of $a_i, i = 1, 2$, and $b_j, j = 1, \ldots, 6$, [see (A.4)], we obtain:

$$\begin{aligned} a_2 = \lambda_1 + \lambda_2, \quad a_1 = \lambda_1\lambda_2\sigma_\theta^{-2}, \\ b_6 = 1, \quad b_5 = b_4 = b_3 = 0, \quad b_2 = \lambda_2\sigma_\theta^{-2}, \quad b_1 = \lambda_1\sigma_\theta^{-2}. \end{aligned}$$

From (A.12), it is easy to verify that $\beta_3 = \alpha_8 = \alpha_7 = 0$, and $\beta_2 = a_2(b_1 - b_2) = (\lambda_1 - \lambda_2)(\lambda_1 + \lambda_2)\sigma_\theta^{-2} > 0$. Thus, f_1 [see (3.59)] can be simplified into the following 6th-order bivariate polynomial [see (3.64)]:

$$0 = f_1(x, y) = \beta_2 xy\Delta^2 + (\alpha_6 x^3 + \alpha_5 x^2 y + \alpha_4 xy^2 + \alpha_3 y^3 + \beta_1 xy)\Delta + (\alpha_2 x + \alpha_1 y)xy.$$

By setting the coefficients of (3.64) and (3.65) equal, we have:

$$\begin{aligned}
\zeta_5 &= \beta_2 x + \alpha_3, \\
\zeta_4 &= \alpha_4 x, \\
\zeta_3 &= 2\beta_2 x^3 + (\alpha_5 + \alpha_3)x^2 + \beta_1 x, \\
\zeta_2 &= (\alpha_6 + \alpha_4)x^3 + \alpha_1 x, \\
\zeta_1 &= \beta_2 x^5 + \alpha_5 x^4 + \beta_1 x^3 + \alpha_2 x^2, \\
\zeta_0 &= \alpha_6 x^5.
\end{aligned}$$

A.5 Coefficients of (3.59) for the case $\kappa_d = 1, \kappa_\theta = 0, \lambda_1 > \lambda_2$

Based on the expressions of β_3, α_8 and α_7 in (A.12), it is straightforward to verify that $\alpha_8 = -c_1\beta_3$ and $\alpha_7 = -c_2\beta_3$. By substituting $\kappa_d = 1, \kappa_\theta = 0$ in (A.4), we have:

$$\begin{gathered}
a_2 = \lambda_1 + \lambda_2 + \lambda_1\lambda_2\sigma_d^{-2}, \quad a_1 = 0, \\
b_6 = 1, \quad b_5 = \lambda_1\sigma_d^{-2}, \quad b_4 = \lambda_2\sigma_d^{-2}, \quad b_3 = b_2 = b_1 = 0.
\end{gathered}$$

Substituting the above a_i, $i = 1, 2$, and b_j, $j = 1, \ldots, 6$, into (A.12), and recalling that $\lambda_1 > \lambda_2$, yields:

$$\begin{aligned}
&\beta_3 = a_2(b_4 - b_5) = (\lambda_2 - \lambda_1)\sigma_d^{-2}a_2 < 0, \\
&\alpha_8 = -c_1\beta_3, \quad \alpha_7 = c_2\beta_3, \\
&\beta_2 = \beta_1 = \alpha_6 = \alpha_5 = \alpha_4 = \alpha_3 = \alpha_2 = \alpha_1 = 0,
\end{aligned}$$

and we obtain (3.67):

$$0 = f_1(x, y) = \beta_3\Delta^2 xy(x^2 + y^2 - c_1 x - c_2 y).$$

A.6 Cartesian coordinates for tangent points A & B

We hereafter determine the elements of the set $\bar{\Xi}_{1l} = \{(x, y)|\xi_1(x, y) = f_2(x, y) = 0\}$.

Clearly, $f_2 = 0$ [see (3.57)] describes a circle in the plane, denoted as $\mathsf{O1}$, with radius r and center C, whose Cartesian coordinates are $[x_C \ y_C]^{\mathrm{T}} = \mathbf{c} = [c_1 \ c_2]^{\mathrm{T}}$ [see Figure 3.6(a)]. On the other hand, by rewriting (3.68) as:

$$0 = \xi_1(x, y) = \left(x - \frac{c_1}{2}\right)^2 + \left(y - \frac{c_2}{2}\right)^2 - \frac{\|\mathbf{c}\|^2}{4},$$

it is straightforward to see that $\xi_1 = 0$ also corresponds to a circle $\mathsf{O2}$ in the plane with radius $\frac{1}{2}\|\mathbf{c}\|$, whose center C' [see Figure 3.6(a)] is the midpoint between the origin O and C, i.e., $[x_{C'} \ y_{C'}]^{\mathrm{T}} = \frac{1}{2}\mathbf{c} = \frac{1}{2}[c_1 \ c_2]^{\mathrm{T}}$. By assumption $\mathbf{c} \neq \mathbf{0}_{2\times 1}$,[1] hence $\mathsf{O2}$ and $\mathsf{O1}$ are not concentric, which in turn implies $\bar{\Xi}_{1l}$, corresponding to the intersection of $\mathsf{O1}$ and $\mathsf{O2}$, has at most two elements.

Note that both O and C satisfy (3.68) and thus belong to $\mathsf{O2}$, and since O, C', and C are on the same line, we conclude that the line segment[2] $\overline{OC}$ is the diameter of the circle $\mathsf{O2}$. Moreover, since $\overline{OA}$ and $\overline{OB}$ are two tangent lines to the circle $\mathsf{O1}$, intersecting $\mathsf{O1}$ at A and B respectively, both triangles OAC and OBC are right triangles and share the common hypotenuse $\overline{OC}$ [see Figure 3.6(a)]. Now let us focus on the right triangle OAC. Recalling that C' is the midpoint of the hypotenuse $\overline{OC}$, based on the median theorem, we conclude that $\|C'A\|$ is exactly half of $\|OC\|$, i.e., $\|C'A\| = \frac{1}{2}\|\mathbf{c}\|$. In other words, A is located on the circle whose center is C' and radius is $\frac{1}{2}\|\mathbf{c}\|$, which is precisely $\mathsf{O2}$. Therefore, $A \in \bar{\Xi}_{1l}$. The same argument also applies to B. Since it has been established that $\bar{\Xi}_{1l}$ has at most two elements, we conclude that $\bar{\Xi}_{1l}$ contains exactly two real elements, which are the two tangent points A and B.

To acquire the Cartesian coordinates of A and B [see Figure 3.6(a)], we first apply the Pythagorean theorem to the right triangles OAC and OBC to obtain $\|OA\| = \|OB\| = \tau = \sqrt{\|\mathbf{c}\|^2 - r^2}$. Note that the angles $\widehat{AOC} = \widehat{COB} = \omega = \arcsin\left(\frac{r}{\|\mathbf{c}\|}\right)$,

[1] Note that when $\mathbf{c} = \mathbf{0}_{2\times 1}$, the sensor's current location coincides with the one-step-ahead target's estimated position, i.e., $\mathbf{p}_S(k) = \hat{\mathbf{p}}_T(k+1|k)$, yielding $r = 0$. If the sensor opts to stay at its current location, i.e., $\mathbf{p}_S(k+1) = \mathbf{p}_S(k) = \hat{\mathbf{p}}_T(k+1|k)$, collision between the sensor and the target will likely occur at the next time-step $k+1$. Thus, the sensor should move in order to avoid collision.

[2] $\overline{AB}$ represents the line segment with two end points A and B, while $\|AB\|$ denotes the Euclidean norm of $\overline{AB}$.

$\widehat{XOC} = \varphi_C = \arctan\left(\frac{c_2}{c_1}\right)$, and

$$\begin{aligned} \widehat{XOA} = \widehat{XOC} - \widehat{AOC} &\implies \varphi_A = \varphi_C - \omega, \\ \widehat{XOB} = \widehat{XOC} + \widehat{COB} &\implies \varphi_B = \varphi_C + \omega. \end{aligned}$$

Therefore

$$\begin{aligned} \mathbf{s}_A = \begin{bmatrix} x_A \\ y_A \end{bmatrix} = \|OA\| \begin{bmatrix} \cos\varphi_A \\ \sin\varphi_A \end{bmatrix} = \tau \begin{bmatrix} \cos(\varphi_C - \omega) \\ \sin(\varphi_C - \omega) \end{bmatrix}, \\ \mathbf{s}_B = \begin{bmatrix} x_B \\ y_B \end{bmatrix} = \|OB\| \begin{bmatrix} \cos\varphi_B \\ \sin\varphi_B \end{bmatrix} = \tau \begin{bmatrix} \cos(\varphi_C + \omega) \\ \sin(\varphi_C + \omega) \end{bmatrix}, \end{aligned}$$

which is precisely (3.70).

A.7 Coefficients of (3.59) for the case $\lambda_1 = \lambda_2$

Substituting $\lambda_1 = \lambda_2 = \lambda$ into the expressions of $a_i, i = 1, 2$, and $b_j, j = 1, \ldots, 6$, [see (A.4)], yields:

$$\begin{aligned} a_2 = 2\lambda + \lambda^2\kappa_d\sigma_d^{-2}, \quad a_1 = \lambda^2\kappa_\theta\sigma_\theta^{-2}, \\ b_6 = 1, \quad b_5 = b_4 = \lambda\kappa_d\sigma_d^{-2}, \\ b_3 = \lambda^2\kappa_d\sigma_d^{-2}\kappa_\theta\sigma_\theta^{-2}, \quad b_2 = b_1 = \lambda\kappa_\theta\sigma_\theta^{-2}. \end{aligned}$$

From (A.12), it is easy to verify that

$$\begin{aligned} \beta_3 = \beta_2 = \beta_1 = \alpha_8 = \alpha_7 = \alpha_2 = \alpha_1 = 0, \\ \alpha_6 = \alpha_4 = c_2\epsilon, \quad \alpha_5 = \alpha_3 = -c_1\epsilon, \end{aligned}$$

where

$$\epsilon = a_1(b_5 + b_6) - a_2(b_2 + b_3) = -\lambda^2\kappa_\theta\sigma_\theta^{-2}(1 + \lambda\kappa_d\sigma_d^{-2})^2.$$

Thus $f_1(x, y)$ [see (3.59)] can be transformed into:

$$0 = f_1(x, y) = \epsilon(c_2x^3 - c_1x^2y + c_2xy^2 - c_1y^3)\Delta = \epsilon\Delta^2(c_2x - c_1y).$$

For the single-sensor target tracking with either bearing-only or distance-and-bearing observations, $\kappa_\theta = 1$, thus $\epsilon < 0$. Additionally, $\Delta = x^2 + y^2 > 0$, hence,

$$f_1(x, y) = 0 \iff \xi_3(x, y) = c_2 x - c_1 y = 0.$$

It is straightforward to verify that ξ_3 depicts a straight line passing through the origin O and the center C. Therefore, the critical points satisfying $f_1 = f_2 = 0$, or equivalently $\xi_3 = f_2 = 0$, must be the intersections between the line defined by $\xi_3 = 0$ and the circle described by $f_2 = 0$ [see (3.57)]. Hence, by referring to Figure 3.6(a), the two critical points are readily obtained as D and D', with $\mathbf{s}_D = \frac{\mathbf{c}}{\|\mathbf{c}\|}\big(\|\mathbf{c}\| - r\big)$ and $\mathbf{s}_{D'} = \frac{\mathbf{c}}{\|\mathbf{c}\|}\big(\|\mathbf{c}\| + r\big)$.

On the other hand, for the distance-only measurement model, $\kappa_\theta = 0$, thus $\epsilon = 0$, which yields $f_1(x, y) = 0$ regardless of x and y. Furthermore, by substituting in (3.49) $\kappa_d = 1, \kappa_\theta = 0, \lambda_1 = \lambda_2 = \lambda$, it can be shown that

$$f_0(\mathbf{s}) = \mathrm{tr}\left(\lambda^{-1}\mathbf{I}_2 + \frac{1}{\sigma_d^2}\frac{\mathbf{s}\mathbf{s}^{\mathrm{T}}}{\mathbf{s}^{\mathrm{T}}\mathbf{s}}\right)^{-1} = \lambda + \frac{\lambda\sigma_d^2}{\lambda + \sigma_d^2} = \text{const.}$$

In other words, $f_0(\mathbf{s})$ is independent of $\mathbf{s}$, and thus $\nabla f_0(\mathbf{s}) = \mathbf{0}_{2\times 1},\ \forall \mathbf{s}$. Therefore, the sensor's location will not affect the trace of the target position posterior covariance for the distance-only measurement model, if $\lambda_1 = \lambda_2 = \lambda$.

A.8 Transforming (3.74) into (3.75)

By comparing equations (3.58) and (3.74), it is obvious that (3.74) is a special case of (3.58), obtained by choosing $\mathbf{c} = [c_1\ c_2]^{\mathrm{T}} = [0\ 0]^{\mathrm{T}}$ in (3.58). Thus, from (A.12), we conclude $\alpha_j = 0,\ j = 1, \ldots, 8$. Moreover, by substituting $\alpha_j = 0,\ j = 1, \ldots, 8$, in (3.59), or equivalently, setting $c_1 = c_2 = 0$ in (A.11), we obtain:

$$0 = xyh(\mathbf{s}^*)\big[g_y(\mathbf{s}^*) - g_x(\mathbf{s}^*)\big], \tag{A.13}$$

where the polynomials $h(\mathbf{s}^*),\ g_x(\mathbf{s}^*),$ and $g_y(\mathbf{s}^*)$ are defined in (A.7)-(A.9).

To acquire $g_y(\mathbf{s}^*) - g_x(\mathbf{s}^*)$, we substitute b_1, b_2, b_4, b_5 [see (A.4)] into (A.8)-(A.9) and obtain:

$$g_y(\mathbf{s}^*) - g_x(\mathbf{s}^*) = (\lambda_2 - \lambda_1)\big[\kappa_d\sigma_d^{-2}(x^2 + y^2) - \kappa_\theta\sigma_\theta^{-2}\big] = (\lambda_2 - \lambda_1)(\kappa_d\sigma_d^{-2}\rho^2 - \kappa_\theta\sigma_\theta^{-2}), \tag{A.14}$$

where we have employed the equality $x^2 + y^2 = \rho^2$ [see (3.73)]. Note that $h(\mathbf{s}^*) > 0$, and if we assume $\lambda_1 > \lambda_2$, and $\kappa_d \sigma_d^{-2} \rho^2 \neq \kappa_\theta \sigma_\theta^{-2}$ (which is automatically satisfied for distance-only and bearing-only measurement models, and also holds true if $\rho \neq \frac{\sigma_d}{\sigma_\theta}$ for the distance-and-bearing observation model), then from (A.14), $g_y(\mathbf{s}^*) - g_x(\mathbf{s}^*) \neq 0$. Hence, (A.13) can be further simplified into (3.75), i.e., $f_4(x, y) = xy = 0$.

A.9 Feasibility of the critical points

In what follows, we show that $\acute{\mathbf{s}} = [-\text{sign}(c_1)\rho \;\; 0]^{\mathrm{T}}$ and $\grave{\mathbf{s}} = [0 \;\; -\text{sign}(c_2)\rho]^{\mathrm{T}}$ violate the maximum-speed constraint (3.50). To proceed, we evaluate $\|\mathbf{s} - \mathbf{c}\|$ at $\acute{\mathbf{s}}$ and $\grave{\mathbf{s}}$:

$$\|\acute{\mathbf{s}} - \mathbf{c}\|^2 = \big(-\text{sign}(c_1)\rho - c_1 \big)^2 + c_2^2 = \big(\rho + |c_1|\big)^2 + c_2^2 > c_1^2 + c_2^2 = \|\mathbf{c}\|^2 \geq r^2,$$
$$\|\grave{\mathbf{s}} - \mathbf{c}\|^2 = c_1^2 + \big(-\text{sign}(c_2)\rho - c_2 \big)^2 = c_1^2 + \big(\rho + |c_2|\big)^2 > c_1^2 + c_2^2 = \|\mathbf{c}\|^2 \geq r^2.$$

Therefore $\|\acute{\mathbf{s}} - \mathbf{c}\| > r$ and $\|\grave{\mathbf{s}} - \mathbf{c}\| > r$, or equivalently, $\acute{\mathbf{s}}$ and $\grave{\mathbf{s}}$ do not satisfy the maximum-speed constraint (3.50).

Next, let us consider $-\acute{\mathbf{s}} = [\text{sign}(c_1)\rho \;\; 0]^{\mathrm{T}}$. Clearly $-\acute{\mathbf{s}}$ automatically satisfies $\|-\acute{\mathbf{s}}\| = \rho$, hence,

$$-\acute{\mathbf{s}} \in \bar{\Omega} \iff r^2 \geq \| -\acute{\mathbf{s}} - \mathbf{c}\|^2 = \big(\rho - |c_1|\big)^2 + c_2^2,$$

and by subtracting c_2^2 on both sides, we obtain (3.76).

Applying the same argument to $-\grave{\mathbf{s}} = [0 \;\; \text{sign}(c_2)\rho]^{\mathrm{T}}$,

$$-\grave{\mathbf{s}} \in \bar{\Omega} \iff r^2 \geq \| -\grave{\mathbf{s}} - \mathbf{c}\|^2 = c_1^2 + \big(\rho - |c_2|\big)^2$$

yields (3.77).

A.10 Cartesian coordinates for intersections E & F

The Cartesian coordinates of E and F can be derived in a similar way as A and B in Appendix A.6. Referring to Figure 3.6(b), since the two boundary points E and F are also the intersection points of the two circles: $\|\mathbf{s} - \mathbf{c}\| = r$ and $\|\mathbf{s}\| = \rho$, we have $\|OE\| = \|OF\| = \rho$ and $\|CE\| = \|CF\| = r$. Furthermore, $\|OC\| = \|\mathbf{c}\|$. Applying

the law of cosines to the triangles OEC and OFC, we obtain $\widehat{EOC} = \widehat{COF} = \varpi = \arccos\left(\frac{\rho^2+\|\mathbf{c}\|^2-r^2}{2\rho\|\mathbf{c}\|}\right)$. Moreover, $\widehat{XOC} = \varphi_C = \arctan\left(\frac{c_2}{c_1}\right)$, and

$$\begin{aligned}\widehat{XOE} = \widehat{XOC} - \widehat{EOC} \quad &\Longrightarrow \quad \varphi_E = \varphi_C - \varpi,\\ \widehat{XOF} = \widehat{XOC} + \widehat{COF} \quad &\Longrightarrow \quad \varphi_F = \varphi_C + \varpi.\end{aligned}$$

Thus,

$$\begin{aligned}\mathbf{s}_E = \begin{bmatrix} x_E \\ y_E \end{bmatrix} = \|OE\| \begin{bmatrix} \cos\varphi_E \\ \sin\varphi_E \end{bmatrix} = \rho \begin{bmatrix} \cos(\varphi_C - \varpi) \\ \sin(\varphi_C - \varpi) \end{bmatrix},\\ \mathbf{s}_F = \begin{bmatrix} x_F \\ y_F \end{bmatrix} = \|OF\| \begin{bmatrix} \cos\varphi_F \\ \sin\varphi_F \end{bmatrix} = \rho \begin{bmatrix} \cos(\varphi_C + \varpi) \\ \sin(\varphi_C + \varpi) \end{bmatrix},\end{aligned}$$

which is precisely (3.78).

A.11 Special cases of (A.13)

Note that (A.13) remains 0 regardless of x and y, if and only if $g_x(\mathbf{s}^*) = g_y(\mathbf{s}^*)$, which [see (A.14)] is equivalent to either $\lambda_1 = \lambda_2$, or $\rho = \frac{\sigma_d}{\sigma_\theta}$ for $\kappa_d = \kappa_\theta = 1$.

Let us first examine $\lambda_1 = \lambda_2 = \lambda$. To proceed, we first parameterize the circle [see (3.73)] through its polar coordinates, i.e., $\mathbf{s} = [x\ y]^{\mathrm{T}} = \rho[\cos\varphi\ \sin\varphi]^{\mathrm{T}}$, and evaluate (3.49) by substituting $\mathbf{s} = \rho[\cos\varphi\ \sin\varphi]^{\mathrm{T}}$. After algebraic manipulation, we obtain, for any $\mathbf{s}$ such that $\mathbf{s}^{\mathrm{T}}\mathbf{s} = \rho^2$,

$$f_0(\mathbf{s}) = \mathrm{tr}\left(\lambda^{-1}\mathbf{I}_2 + \frac{\kappa_d}{\sigma_d^2}\frac{\mathbf{s}\mathbf{s}^{\mathrm{T}}}{\mathbf{s}^{\mathrm{T}}\mathbf{s}} + \frac{\kappa_\theta}{\sigma_\theta^2}\frac{\mathbf{J}\mathbf{s}\mathbf{s}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{(\mathbf{s}^{\mathrm{T}}\mathbf{s})^2}\right)^{-1} = \frac{\lambda\sigma_d^2}{\kappa_d\lambda + \sigma_d^2} + \frac{\lambda\sigma_\theta^2\rho^2}{\kappa_\theta\lambda + \sigma_\theta^2\rho^2} = \text{const.}$$

In other words, $f_0(\mathbf{s})$ remains constant along the curve EGF [see Figure 3.6(b)] if $\lambda_1 = \lambda_2$. Moreover, the value of this constant depends on κ_d and κ_θ.

Next, we focus on the other condition: $\kappa_d = \kappa_\theta = 1$, and $\rho = \frac{\sigma_d}{\sigma_\theta}$. To proceed, we first realize that the following identity: $\mathbf{s}\mathbf{s}^{\mathrm{T}} + \mathbf{J}\mathbf{s}\mathbf{s}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}} = \left(\mathbf{s}^{\mathrm{T}}\mathbf{s}\right)^2\mathbf{I}_2$, holds true for any 2D vector $\mathbf{s}$. By substituting $\kappa_d = \kappa_\theta = 1$, $\mathbf{s}^{\mathrm{T}}\mathbf{s} = \rho^2 = \frac{\sigma_d^2}{\sigma_\theta^2}$ into (3.49), and employing the above identity, we obtain:

$$\mathbf{\Lambda} + \frac{1}{\sigma_d^2}\frac{\mathbf{s}\mathbf{s}^{\mathrm{T}}}{\mathbf{s}^{\mathrm{T}}\mathbf{s}} + \frac{1}{\sigma_\theta^2}\frac{\mathbf{J}\mathbf{s}\mathbf{s}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{(\mathbf{s}^{\mathrm{T}}\mathbf{s})^2} = \mathbf{\Lambda} + \frac{1}{\sigma_d^2}\frac{\mathbf{s}\mathbf{s}^{\mathrm{T}}}{\mathbf{s}^{\mathrm{T}}\mathbf{s}} + \frac{1}{\sigma_d^2}\frac{\mathbf{J}\mathbf{s}\mathbf{s}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}}{\mathbf{s}^{\mathrm{T}}\mathbf{s}} = \mathbf{\Lambda} + \sigma_d^{-2}\mathbf{I}_2,$$

and

$$f_0(\mathbf{s}) = \mathrm{tr}\left(\mathbf{\Lambda} + \sigma_d^{-2}\mathbf{I}_2\right)^{-1} = \frac{\lambda_1\sigma_d^2}{\lambda_1 + \sigma_d^2} + \frac{\lambda_2\sigma_d^2}{\lambda_2 + \sigma_d^2} = \text{const.}$$

Hence, $f_0(\mathbf{s})$ remains constant along the curve EGF [see Figure 3.6(b)], if $\kappa_d = \kappa_\theta = 1$, and $\rho = \frac{\sigma_d}{\sigma_\theta}$.

Appendix B

Appendix for Chapter 5

B.1 Computational complexity of alternative QR decomposition algorithms

In this appendix, we analyze the computational complexity of QR decomposition on $\mathbf{\Lambda}$ [see (5.31)] by employing (i) the Cholesky decomposition, (ii) the modified Gram-Schmidt process, and (iii) the Givens rotations. Similar to the discussion in Section 5.4.3, we assume $\mathbf{\Pi}_\ell = \mathbf{I}_{n_{\mathbf{\Lambda}}}, \ell = 1, \ldots, n_{\mathbf{\Lambda}}$, for simplicity, where $m_{\mathbf{\Lambda}}$ and $n_{\mathbf{\Lambda}}$ are the number of rows and columns of $\mathbf{\Lambda}$. Furthermore, the thin QR decomposition of $\mathbf{\Lambda}$ takes the form $\mathbf{\Lambda} = \mathbf{Q}_{\mathbf{\Lambda}}\mathbf{R}_{\mathbf{\Lambda}}$, where $\mathbf{Q}_{\mathbf{\Lambda}}$ is orthonormal and $\mathbf{R}_{\mathbf{\Lambda}}$ is upper triangular.

B.1.1 Cholesky decomposition

Since the columns of $\mathbf{Q}_{\mathbf{\Lambda}}$ are orthonormal, we have $\mathbf{\Lambda}^{\mathrm{T}}\mathbf{\Lambda} = \mathbf{R}_{\mathbf{\Lambda}}^{\mathrm{T}}\mathbf{R}_{\mathbf{\Lambda}}$. Therefore, we can apply the Cholesky decomposition (see [105, Section 4.2.3]) on $\mathbf{\Lambda}^{\mathrm{T}}\mathbf{\Lambda}$ to compute the upper triangular matrix $\mathbf{R}_{\mathbf{\Lambda}}$.

Analogous to the analysis shown in Section 5.3.3, the cost of evaluating $\mathbf{\Lambda}^{\mathrm{T}}\mathbf{\Lambda}$ is only quadratic in N due to the sparse structure of $\mathbf{\Lambda}$. In addition, since the Cholesky decomposition of a square matrix of dimension n requires $\mathcal{O}(n^3)$ operations [105, Section 4.2.3], we conclude that the time complexity of QR factorization on $\mathbf{\Lambda}$ using the Cholesky decomposition is $\mathcal{O}(N^3)$ (note that $\mathbf{Q}_{\mathbf{\Lambda}}$ is not computed when using the Cholesky decomposition). On the other hand, since $\mathbf{nnz}(\mathbf{\Lambda}) \sim \mathcal{O}(N^2)$ and both matrices $\mathbf{\Lambda}^{\mathrm{T}}\mathbf{\Lambda}$ and

$\mathbf{R}_{\mathbf{\Lambda}}$ are of dimensions $N \times N$, the space complexity is $\mathcal{O}(N^2)$.

Although QR factorization on $\mathbf{\Lambda}$ using the Cholesky decomposition has low computational complexity, it has two drawbacks. First, the matrix product $\mathbf{\Lambda}^{\mathrm{T}}\mathbf{\Lambda}$ squares the condition number of $\mathbf{\Lambda}$, i.e., $\kappa(\mathbf{\Lambda}^{\mathrm{T}}\mathbf{\Lambda}) = \kappa^2(\mathbf{\Lambda})$, which may introduce numerical instability. Second, in order to apply the Cholesky decomposition, the matrix product (i.e., $\mathbf{\Lambda}^{\mathrm{T}}\mathbf{\Lambda}$) must be strictly positive definite. Unfortunately, $\mathbf{H}$ is rank-deficient (see Remark 18), and $\mathbf{H}^{\mathrm{T}}\mathbf{H}$ is positive semi-definite. Hence, the Cholesky decomposition is not applicable for the QR factorization of $\mathbf{H}$ addressed in Chapter 5.

B.1.2 Modified Gram-Schmidt

For clarity, we use $\boldsymbol{\lambda}_i$ and $\mathbf{q}_i$ to denote the ith columns of $\mathbf{\Lambda}$ and $\mathbf{Q}_{\mathbf{\Lambda}}$, respectively. Furthermore, we use $r_{i,j}$ to denote the (i,j)th element of $\mathbf{R}_{\mathbf{\Lambda}}$. For the purpose of complexity analysis, we outline the algorithmic flow chart of the Modified Gram-Schmidt process in Algorithm 7. More details are provided in [105, Section 5.2.8].

Algorithm 7 Modified Gram-Schmidt [105, Algorithm 5.2.5]

Require: $\mathbf{\Lambda}$ of dimensions $m_{\mathbf{\Lambda}} \times n_{\mathbf{\Lambda}}$ ($m_{\mathbf{\Lambda}} \geq n_{\mathbf{\Lambda}}$).

Ensure: $\mathbf{R}_{\mathbf{\Lambda}}$ of dimensions $n_{\mathbf{\Lambda}} \times n_{\mathbf{\Lambda}}$.

1: **for** $\ell = 1$ **to** $n_{\mathbf{\Lambda}}$, **do**
2: Set $\bar{\mathbf{q}} = \boldsymbol{\lambda}_\ell$.
3: **for** $j = 1$ **to** $\ell - 1$, **do**
4: Compute $r_{j,\ell} = \mathbf{q}_j^{\mathrm{T}}\bar{\mathbf{q}}$.
5: Update $\bar{\mathbf{q}} \Leftarrow \bar{\mathbf{q}} - r_{j,\ell}\mathbf{q}_j$.
6: **end for**
7: Calculate $r_{\ell,\ell} = \|\bar{\mathbf{q}}\|_2$.
8: **if** $r_{\ell,\ell} = 0$, **then**
9: STOP.
10: **else**
11: $\mathbf{q}_\ell = \bar{\mathbf{q}}/r_{\ell,\ell}$.
12: **end if**
13: **end for**
14: **return** $\mathbf{R}_{\mathbf{\Lambda}}$.

From Algorithm 7, it is clear that the vector $\mathbf{q}_\ell$ is a linear combination of $\boldsymbol{\lambda}_\ell$ and $\{\mathbf{q}_j, j = 1, \ldots, \ell - 1\}$. Since every $\mathbf{q}_j, j = 2, \ldots, \ell - 1$, can be expressed as a linear combination of $\boldsymbol{\lambda}_j$ and $\{\mathbf{q}_i, i = 1, \ldots, j - 1\}$, and $\mathbf{q}_1$ is a scalar multiplication of $\boldsymbol{\lambda}_1$, we conclude by recursion that the vector $\mathbf{q}_\ell$ is a linear combination of $\{\boldsymbol{\lambda}_j, j = 1, \ldots, \ell\}$. By examining the indices of the nonzero elements in each vector $\{\boldsymbol{\lambda}_j, j = 1, \ldots, \ell\}$, it can be shown that the first $\sum_{j=1}^{\ell}(N-j)$ elements of $\mathbf{q}_\ell$ are non-zeros, while its rest elements are zeros. Therefore, we have $\mathbf{nnz}(\mathbf{q}_\ell) = \sum_{j=1}^{\ell}(N-j)$ for $\ell = 1, \ldots, N$. Hence, updating $\bar{\mathbf{q}}$ at the jth iteration inside the inner "for-loop" (see Algorithm 7, Line 5), which can be shown is the most dominant computational process, requires approximately $\sum_{i=1}^{j}(N-i)$ steps. Therefore, the overall time complexity of the QR decomposition of $\boldsymbol{\Lambda}$ using the Modified Gram-Schmidt QR is in the order of

$$\sum_{\ell=1}^{N}\left\{\sum_{j=1}^{\ell-1}\left[\sum_{i=1}^{j}(N-i)\right]\right\} \sim \mathcal{O}(N^4).$$

Furthermore, for space complexity, we focus on the storage of $\{\mathbf{q}_\ell, \ell = 1, \ldots, n_{\boldsymbol{\Lambda}}\}$, which can be shown is the most demanding source in terms of memory usage. Since $\mathbf{nnz}(\mathbf{q}_\ell) = \sum_{j=1}^{\ell}(N-j)$, we conclude that the space complexity is in the order of

$$\sum_{\ell=1}^{N}\left[\sum_{j=1}^{\ell}(N-j)\right] \sim \mathcal{O}(N^3).$$

An alternative method to the Modified Gram-Schmidt QR algorithm is to apply the Classical Gram-Schmidt process [105, Section 5.2.7]. However, the Classical Gram-Schmidt process only differs from the Modified Gram-Schmidt in the computation of $r_{j,\ell}$, while the update of $\bar{\mathbf{q}}$ is exactly the same for both methods. Hence, we conclude that the computational complexity of the QR decomposition of $\boldsymbol{\Lambda}$ using the Classical Gram-Schmidt is the same as that of the Modified Gram-Schmidt. Furthermore, the Classical Gram-Schmidt suffers from numerical instability and is rarely applied in practice [105, Section 5.2.8].

B.1.3 **Givens** QR

The Givens QR algorithm is one of the most widely adopted numerical techniques for implementing QR factorization [105, Section 5.2.3]. To understand the Givens QR

algorithm, we first introduce the Givens rotation $\mathbf{G}^{\phi} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$, a 2D rotational matrix. Given a 2×1 vector $\mathbf{x} = [x_1 \ x_2]^{\mathrm{T}}$, by selecting $\phi = \arctan(-x_2/x_1)$, it is straightforward to verify $\mathbf{G}^{\phi}\mathbf{x} = [\|\mathbf{x}\|_2 \ 0]^{\mathrm{T}}$ [105, Section 5.1.8]. In other words, left-multiplying $\mathbf{x}$ by $\mathbf{G}^{\phi}$ eliminates x_2. Essentially, the Givens QR successively invokes the Givens rotations to eliminate non-zeros below the diagonal of a matrix and transforms it into upper triangular form [105, Algorithm 5.2.2].

The time complexity of the Givens QR process for decomposing an arbitrary matrix $\mathbf{\Lambda}$ is in the order of $m_{\mathbf{\Lambda}} n_{\mathbf{\Lambda}}^2$ [105, Section 5.2.3]. For $\mathbf{\Lambda}$ addressed here [see (5.31)], the resulting time complexity is $\mathcal{O}(N^4)$. However, this does not take into account of the structure of $\mathbf{\Lambda}$, and we first conjectured that the overall time complexity using the Givens QR can be reduced to $\mathcal{O}(N^3)$ by exploiting the sparsity of $\mathbf{\Lambda}$. Unfortunately, this conjecture is false. In what follows, we examine both the time complexity and space complexity of applying the Givens QR algorithm to decompose $\mathbf{\Lambda}$ in (5.31). For clarity, we use $\lambda_{i,j}$ to denote the (i,j)th element of $\mathbf{\Lambda}$. Furthermore, we partition $\mathbf{\Lambda}$ by rows into $\mathbf{\Lambda} = [\mathbf{\Lambda}_1^{\mathrm{T}} \ \ldots \ \mathbf{\Lambda}_{N-1}^{\mathrm{T}}]^{\mathrm{T}}$, where the rows of the block-matrix $\mathbf{\Lambda}_i$ correspond to the $(J_{i-1}+1)$th through J_ith rows of $\mathbf{\Lambda}$, with $J_0 = 0$ and $J_i = J_{i-1} + (N-i) = \sum_{j=1}^{i}(N-j)$ for $i = 1, \ldots, N-1$. To facilitate the discussion, we provide the pseudo-code of the Givens QR in Algorithm 8. It is worth noting that each Givens rotation affects only two rows of $\mathbf{\Lambda}$ (see Algorithm 8, Lines 5–9), whereas the processing cost of Line 7 is constant (4 multiplications and 2 additions).

During the first iteration (i.e., $\ell = 1$) of the outermost for-loop (see Algorithm 8), a sequence of $N-2$ Givens rotations is applied to $\mathbf{\Lambda}$ (more specifically, to $\mathbf{\Lambda}_1$) so that its first column has zeros below its first component. In particular, the first Givens rotation $\mathbf{G}^{\phi_1}$ with $\phi_1 = \arctan\left(-\frac{\psi^{1,N}}{\psi^{1,N-1}}\right)$ eliminates $\psi^{1,N}$, while introducing 2 nonzero fill-ins $\lambda_{N-1,N-1}, \lambda_{N-2,N}$ to $\mathbf{\Lambda}_1$. Similarly in spirit, after the second Givens rotation $\mathbf{G}^{\phi_2}$ [$\phi_2 = \arctan\left(-\frac{\lambda_{N-2,1}}{\psi^{1,N-2}}\right)$], $\lambda_{N-2,1}$ vanishes and 3 extra non-zeros $\lambda_{N-2,N-2}, \lambda_{N-3,N-1}, \lambda_{N-3,N}$ are added to $\mathbf{\Lambda}_1$. In particular, $\lambda_{N-3,N}$ is introduced to $\mathbf{\Lambda}_1$ due to $\lambda_{N-2,N}$, which is generated previously by $\mathbf{G}^{\phi_1}$. The above process is repeated until all non-zeros below the first component in the first column of $\mathbf{\Lambda}$ are eliminated, and the resulting matrix

Algorithm 8 Givens QR [105, Algorithm 5.2.2]

Require: $\mathbf{\Lambda}$ of dimensions $m_{\mathbf{\Lambda}} \times n_{\mathbf{\Lambda}}$ ($m_{\mathbf{\Lambda}} \geq n_{\mathbf{\Lambda}}$).

Ensure: $\mathbf{R}_{\mathbf{\Lambda}}$ of dimensions $n_{\mathbf{\Lambda}} \times n_{\mathbf{\Lambda}}$.

1: **for** $\ell = 1$ **to** $n_{\mathbf{\Lambda}}$, **do**
2: Set $i_0 = \ell$, and seek all indices $\{i_1, \ldots, i_{F_\ell}\}$ such that $i_0 < i_1 < \cdots < i_{F_\ell} \leq m_{\mathbf{\Lambda}}$ and $\lambda_{i_\eta,\ell} \neq 0, \forall \eta = 1, \ldots, F_\ell$.
3: **for** $\eta = F_\ell$ **to** 1, **do**
4: Compute $\phi = \arctan\left(-\frac{\lambda_{i_\eta,\ell}}{\lambda_{i_{\eta-1},\ell}}\right)$, update $\lambda_{i_{\eta-1},\ell} \Leftarrow (\lambda^2_{i_{\eta-1},\ell} + \lambda^2_{i_\eta,\ell})^{1/2}$, $\lambda_{i_\eta,\ell} \Leftarrow 0$.
5: **for** $j = \ell + 1$ **to** $n_{\mathbf{\Lambda}}$, **do**
6: **if** $[\lambda_{i_{\eta-1},j}\ \lambda_{i_\eta,j}]^{\mathrm{T}} \neq \mathbf{0}_{2\times 1}$, **then**
7: $\begin{bmatrix} \lambda_{i_{\eta-1},j} \\ \lambda_{i_\eta,j} \end{bmatrix} \Leftarrow \mathbf{G}^{\phi} \begin{bmatrix} \lambda_{i_{\eta-1},j} \\ \lambda_{i_\eta,j} \end{bmatrix}$
8: **end if**
9: **end for**
10: **end for**
11: **end for**
12: **return** $\mathbf{R}_{\mathbf{\Lambda}}$, the first $n_{\mathbf{\Lambda}}$ rows of $\mathbf{\Lambda}$.

$\mathbf{\Lambda}$, at the end of this process, takes the form

$$\mathbf{\Lambda} = \begin{bmatrix} \times & \times & \boxtimes & \boxtimes & \cdots & \boxtimes \\ & \boxtimes & \times & \boxtimes & \cdots & \boxtimes \\ & & \ddots & \ddots & \ddots & \vdots \\ & & & & & \times \\ & \psi^{2,3} & \upsilon^{2,3} & & & \\ & \psi^{2,4} & & \upsilon^{2,4} & & \\ & \vdots & & & \ddots & \\ & \psi^{2,N} & & & & \upsilon^{2,N} \\ & & \vdots & \cdots & & \vdots \\ & & & & \psi^{N-1,N} & \upsilon^{N-1,N} \end{bmatrix}. \tag{B.1}$$

In particular, $\mathbf{\Lambda}_1$ is transformed into an upper triangular form, while $\mathbf{\Lambda}_j, j = 2, \ldots, N-1$, remains intact. The overall number of basic steps during the first iteration is in the order of $\sum_{i=2}^{N-1} i$. Furthermore, from (B.1), the total number of non-zeros of $\mathbf{\Lambda}_1$ is $\frac{(N-1)N}{2} - 1$, and $\mathbf{nnz}(\mathbf{\Lambda}) = \frac{(N-1)N}{2} + (N-2)(N-1) - 1$.

Similarly, at the second iteration (i.e., $\ell = 2$), a sequence of $N-3$ Givens rotations is applied first to transform $\mathbf{\Lambda}_2$ into upper triangular with upper bandwidth $q = 1$. This process requires the number of arithmetic operations proportional to $\sum_{i=2}^{N-2} i$, and the resulting $\mathbf{\Lambda}$ has $\mathbf{nnz}(\mathbf{\Lambda}) = \frac{1}{2}\sum_{i=1}^{2}(N-i)(N-i+1) + (N-3)(N-2) - 2$. However, note that after this process $\lambda_{J_1+1,2} \neq 0$. Therefore, an extra Givens rotation is required to eliminate $\lambda_{J_1+1,2}$, which affects the 2nd and (J_1+1)th row of $\mathbf{\Lambda}$. Since the (J_1+1)th row of $\mathbf{\Lambda}$ has $N-1$ nonzero elements, this extra Givens rotation process requires $\mathcal{O}(N-1)$ operations. Thus the total time complexity during the second iteration is in the order of $(N-1) + \sum_{i=2}^{N-2} i$.

To seek the pattern of the Givens QR elimination process, as well as the evolution of the matrix $\mathbf{\Lambda}$, let us further examine the third iteration (i.e., $\ell = 3$). Similarly, we first apply a sequence of $N-4$ Givens rotations so that the resulting $\mathbf{\Lambda}_3$ has upper bandwidth $q = 2$, and at the end of this process $\mathbf{nnz}(\mathbf{\Lambda}) = \frac{1}{2}\sum_{i=1}^{3}(N-i)(N-i+1) - 1 + (N-4)(N-3) - 3$, where the subtraction of 1 is due to the elimination of $\lambda_{J_1+1,2}$. After this process, we observe that $\lambda_{J_1+1,3}, \lambda_{J_1+2,3}$, and $\lambda_{J_2+1,3}$ are non-zeros, which requires $2+1=3$ extra Givens rotations to eliminate them, with every Givens

rotation invoking a cost proportional to $N-2$ [due to the fact that the (J_i+j)th row of $\mathbf{\Lambda}$ $(i=1,2; j=1,\ldots,3-i)$ has $N-2$ non-zeros], thus the total time complexity during the third iteration is in the order of $(N-2)\sum_{i=1}^{2} i + \sum_{i=2}^{N-3} i$. After these extra Givens rotations have been applied, the total number of non-zeros in $\mathbf{\Lambda}$ is reduced to $\mathbf{nnz}(\mathbf{\Lambda}) = \frac{1}{2}\sum_{i=1}^{3}(N-i)(N-i+1) - \sum_{j=1}^{2}\sum_{i=1}^{j} i + (N-4)(N-3) - 3$.

Now we are ready to infer the time complexity and space complexity at the ℓth iteration by induction. In particular, a sequence of $N-\ell-1$ Givens rotations is first applied and the resulting $\mathbf{\Lambda}_\ell$ has upper bandwidth $q=\ell-1$, and after this process $\mathbf{nnz}(\mathbf{\Lambda}) = \frac{1}{2}\sum_{i=1}^{\ell}(N-i)(N-i+1) - \sum_{j=1}^{\ell-2}\sum_{i=1}^{j} i + (N-\ell-1)(N-\ell) - \ell$. Second, a total number of $\sum_{i=1}^{\ell-1} i$ Givens rotations are applied to eliminate the non-zeros $\{\lambda_{J_i+j,\ell} \mid i=1,\ldots,\ell-1; j=1,\ldots,\ell-i\}$, with each Givens rotation process introducing a cost proportional to $N-\ell+1$, since there are $N-\ell+1$ nonzero elements in the (J_i+j)th row of $\mathbf{\Lambda}$, $i=1,\ldots,\ell-1; j=1,\ldots,\ell-i$. Hence, the total time complexity during the ℓth iteration is in the order of $(N-\ell+1)\sum_{i=1}^{\ell-1} i + \sum_{i=2}^{N-\ell} i$. After the elimination of $\sum_{i=1}^{\ell-1} i$ non-zeros $\{\lambda_{J_i+j,\ell} \mid i=1,\ldots,\ell-1; j=1,\ldots,\ell-i\}$, the overall number of non-zeros in $\mathbf{\Lambda}$ is reduced to $\mathbf{nnz}(\mathbf{\Lambda}) = \frac{1}{2}\sum_{i=1}^{\ell}(N-i)(N-i+1) - \sum_{j=1}^{\ell-1}\sum_{i=1}^{j} i + (N-\ell-1)(N-\ell) - \ell$.

In summary, the overall time complexity of the QR decomposition of $\mathbf{\Lambda}$ using the Givens QR is proportional to

$$\sum_{\ell=1}^{N}\left[(N-\ell+1)\sum_{i=1}^{\ell-1} i + \sum_{i=2}^{N-\ell} i\right] \sim \mathcal{O}(N^4).$$

On the other hand, since the most dominant source in terms of memory usage is the storage of $\mathbf{\Lambda}$, the space complexity of the QR decomposition of $\mathbf{\Lambda}$ using the Givens QR is

$$\max_{1\le\ell\le N}\big[\mathbf{nnz}(\mathbf{\Lambda})\big] \sim \mathcal{O}(N^3),$$

where $\mathbf{nnz}(\mathbf{\Lambda}) = \frac{1}{2}\sum_{i=1}^{\ell}(N-i)(N-i+1) - \sum_{j=1}^{\ell-2}\sum_{i=1}^{j} i + (N-\ell-1)(N-\ell) - \ell$.