UNIVERSITY OF CALIFORNIA
RIVERSIDE

Fully Distributed Active Joint Localization and Target Tracking Algorithms Design for Multi-Robot System

A Thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Electrical Engineering

by

Shaoshu Su

March 2022

Thesis Committee:

Wei Ren, Chairperson
Konstantinos Karydis
Samet Oymak

The Thesis of Shaoshu Su is approved:

Committee Chairperson

University of California, Riverside

Acknowledgments

First of all, I would like to express my sincere gratitude to my advisor Prof. Wei Ren for the continuous support of my study and research. His patience, motivation, enthusiasm, and immense guidance helped me a lot during my life at UCR. I could not have imagined having a better advisor and mentor for my master's study.

Besides my advisor, I would like to thank other professors I met at UCR. Their broad and rich inspired me much in my study at UCR.

My sincere thanks also go to my fellow labmates in COVEN Group: Pengxiang Zhu, Shan Sun, Yong Ding, Jie Xu, Shaocheng Wang, and the visiting scholars, Bo Wang, Bowen Xu. And I want to give my special thanks to Pengxiang Zhu, whose guidance helped me greatly in my research.

Last but not least, I would like to thank my parents, for giving birth to me in the first place and always supporting and encouraging me throughout my life.

To my advisor and parents and Pengxiang for all the support.

ABSTRACT OF THE THESIS

Fully Distributed Active Joint Localization and Target Tracking Algorithms Design for Multi-Robot System

by

Shaoshu Su

Master of Science, Graduate Program in Electrical Engineering
University of California, Riverside, March 2022
Wei Ren, Chairperson

In this thesis, we study the problem of multi-robot active joint localization and target tracking (AJLATT), where a team of robots mounted with sensors of limited field of view *actively* estimate their own and the target's states cooperatively. Each robot designs its motion strategy to gain better estimation performance while avoiding collisions by using only the information from itself and its one-hop communicating neighbors. By leveraging the framework of joint localization and target tracking (JLATT) presented in our previous work, we propose two fully distributed algorithms that help each robot design motion strategies to achieve better localization and target tracking performance. These two algorithms are designed from, respectively, the control and optimization perspectives. The control-based algorithm is designed by incorporating the estimated target's and robots' states and their uncertainties as well as collision avoidance in the control policy. The optimization-based algorithm minimizes an objective function involving both the target's and robots' estimation uncertainties and a potential function that helps each robot avoid collision and maintain communication connectivity when the robot is planning its motion. Monte Carlo simulations

demonstrate our algorithms' feasibility to solve the AJLATT problem, and performance comparison between these two algorithms is given.

Contents

List of Figures **ix**

List of Tables **x**

1 Introduction **1**

2 Preliminaries **6**
2.1 Motion and Measurement Models 6
2.2 Graphs 7
2.3 Information Fusion Strategy 8
2.4 Joint Localization and Target Tracking 9
2.4.1 Robot Propagation 9
2.4.2 Target Propagation 9
2.4.3 Robot State Update 10
2.4.4 Target State Update 11

3 Active Joint Localization and Target Tracking **13**
3.1 Control-based Approach 14
3.2 Optimization Based Approach 18

4 Simulation **25**
4.1 Simulation Setup 25

5 Conclusions **36**

Bibliography **38**

List of Figures

3.1 Potential function V_{ij} with $d_{s_i} = 6$ m, $d_{o_i} = 10$ m, and $R_i = 15$ m. 17
3.2 Potential function $J_{R_{ij}}$ with $\underline{d}_i = 2$ m, $\bar{d}_i = 15$ m, and $a_i = 2$ m. Notice that the right-hand side increases more moderately than the left-hand side, and is still defined when $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| > \bar{d}_i$. 21

4.1 Position and orientation estimate RMSE for the target on 6 Robots (tracking). 29
4.2 Position and orientation estimate RMSE for 6 Robots (localization). 31
4.3 Comparison of robot localization performance between AJLATT-O1 and AJLATT-O2. 32
4.4 Snapshots of the robots' estimates without the potential function term in the objective function (3.7). The red triangle denotes the true target, and the (overlapped) pink triangles denote the estimated targets on different robots. Triangles and sectors in other colors represent, respectively, different robots' self estimates and fields of view. The line of a robot's sector's is solid when the target is in the robot's field of view; otherwise, it is a dash line. 34
4.5 Robot position RMSE for the example illustrated by Fig. 4.4. Green solid line corresponds to the green robot (Robot 5 here), dash lines with different colors correspond to other robots in the same color in the snapshots in Fig. 4.4. 35

List of Tables

3.1 AJLATT by Robot i 23
3.2 State and Covariance Prediction (SACP) by Robot i at $k-1$ 24

Chapter 1

Introduction

Autonomous multi-robot systems equipped with sensors have attracted more and more attention in recent years due to their wide applications in search and rescue, region monitoring, area surveillance, etc. Multi-robot systems have many appealing properties. In this work, we focus on their usage for estimation. Compared with one single robot, a multi-robot system is able to obtain better self-localization and target tracking performance by utilizing abundant robot-to-robot and robot-to-target measurements as well as information exchanged across the team. That property plays an essential role in the applications of the multi-robot system, especially in the indoor scenario where it is hard to receive the GPS signal. Besides, compared with a static sensor network, a team of autonomous mobile sensors is more adaptive since it is much easier and more effective to be deployed in an unknown environment.

There are many results on estimating the states of robots or a target or both. Some works focus on multi-robot cooperative localization [1–9], where a team of robots can

improve their self-localization accuracy by utilizing the mutual relative measurements and interacting with their team members. Some other works focus on target tracking [10–15], where a static or moving sensor network seeks to estimate the target's state. Here it is usually implicitly assumed that the sensors' states are known (no self-localization needed). There also appear some later works that achieve the cooperative localization and target tracking simultaneously, both in a centralized [16–18] or fully distributed manner [19, 20]. [1]However, all the above works [1–20] have a limitation that the sensors or robots are either static or move without a properly designed motion strategy to actively localize themselves or track the target.

Some algorithms have been proposed to solve the problem of active target tracking from the control or optimization perspective. In the controls field, there are numerous results on distributed tracking or leader-follower tracking (see, e.g., [21]). However, in these results, both the robots' and target's states are usually assumed to be known, and the emphasis is on designing distributed controllers. In [22], the target's state is assumed to be an inaccurate variable, and a gradient-based decentralized motion control strategy is developed to drive a team of robots to estimate and actively track the target's state. Nevertheless, an all-to-all communication network is required in this work. In [23], an information-driven flocking algorithm is proposed to achieve the distributed target state estimation. However, the estimator in [23] relies on a restrictive assumption that the target is jointly observed by each robot and its neighbors. As for the optimization-based approaches, [24] presents an approach for a robot team to find the local optimal action to maximize the knowledge

[1]We use the term *fully distributed* to describe an algorithm that uses only a robot's own and one-hop neighbors' information without the requirement for global parameters, multi-hop information transmission, or multiple communication iterations on certain quantities for iterative consensus-type calculations per time instant.

about the targets by using a greedy search. An algorithm is proposed in [25] to analytically obtain the next global optimal sensing locations for the robots. A non-myopic search-based algorithm is proposed in [26] to drive one robot to actively track the target by minimizing the logarithm of the determinant of the target covariance obtained by the Kalman filter. An algorithm is proposed in [27] to maximize the mutual information between the robots' measurements and their current belief of the target position using an experimental time-of-flight range sensor model for measurement and a Bayesian filter for estimation. Nevertheless, all the above optimization-based works are implemented in a centralized manner. In [28], the approach in [26] is extended in a decentralized way, where each robot in the team uses the coordinate descent method to plan its motion sequentially. As a result, multi-hop information transmission would be required in this approach. Besides, in all of the works above, the robots' states are assumed to be accurately known, which is an unrealistic assumption in practice.

There are some works addressing the active joint localization and target tracking (AJLATT) problem, where motion strategies are designed to improve the target tracking and/or robot self-localization performance. Consider limited sensing capabilities, [29] develops an algorithm which coordinates robots' motions and switches the sensing topology to minimize the uncertainty of the target state estimate. Although this work estimates the robots' states together with the target's state, it does not make improving robot self-localization performance as an objective in its optimization problem. In [30], a gradient-based control policy is designed to minimize the uncertainty of both the robots' and the target's states that are estimated by the Kalman-Bucy filter. While these two works consider

jointly estimating both robots' states and target's state, their approaches are centralized. Although the centralized approaches have the advantage of the capability to obtain the optimal estimation and planning performance, they also cause a heavy burden in computation and communication. In [31], a distributed method is proposed by using a Bayesian filter for estimation and a gradient-based algorithm for control design. However, the method requires multi-hop information transmission for estimation and multi-hop information transmission or multiple communication iterations for iterative consensus calculations per time instant in its control design. As a result, the method is not fully distributed. To the best of our knowledge, there is no existing fully distributed algorithm solving the AJLATT problem.

Considering all the limitations of the previous works, we aim to propose fully distributed AJLATT algorithms. Our previous work [20] introduces a framework to solve the pure joint localization and target tracking (JLATT) problem without active motion strategy design in a fully distributed manner. Leveraging that framework, in this thesis, we move forward by considering how to design fully distributed motion strategies for each robot so that it can not only follow the target but also achieve better self-localization and target tracking performance.

The contributions of this thesis are summarized as follows:

- Based on our previous fully distributed JLATT framework, two motion strategies are proposed from control and optimization perspectives to actively improve each robot's self-localization and target tracking performance.

- In the controls field, compared with the existing centralized and distributed works, it is the first time to solve the AJLATT problem in a fully distributed manner which

requires only one-hop information transmission, no global parameter in the algorithm, and no multiple communication iterations for iterative consensus-type calculations per time instant.

- From the perspective of optimization-based approaches, to the best of our knowledge, our algorithm is the first fully distributed or even the first distributed approach to solve the AJLATT problem compared with the existing centralized works.

- Extensive Monte-Carlo simulations are used to validate and compare the proposed control and optimization-based algorithms.

Chapter 2

Preliminaries

2.1 Motion and Measurement Models

We consider the scenario where M robots track a target on a surface. We use the vectors $\mathbf{x}_i^k$ and $\mathbf{x}_T^k$ to represent, respectively, the true state of robot i and the target at time k. Their movements are driven by a nonlinear motion model as

$$\mathbf{x}_i^k = f_i(\mathbf{x}_i^{k-1}, \mathbf{u}_i^{k-1}, \mathbf{w}_i^{k-1}), \tag{2.1}$$

$$\mathbf{x}_T^k = g(\mathbf{x}_T^{k-1}, \mathbf{u}_T^{k-1}, \mathbf{w}_T^{k-1}), \tag{2.2}$$

where $\mathbf{u}_i^{k-1}$ and $\mathbf{u}_T^{k-1}$ are, respectively, the control inputs for robot i and the target. $\mathbf{w}_i^{k-1} \sim \mathcal{N}(0, \mathbf{Q}_i^{k-1})$ and $\mathbf{w}_T^{k-1} \sim \mathcal{N}(0, \mathbf{Q}_T^{k-1})$ are, respectively, the zero-mean white Gaussian process noises for robot i and the target.

We define $\bar{\mathbf{x}}_i^k$ and $\hat{\mathbf{x}}_i^k$ as, respectively, robot i's prior and posterior estimates of its true state $\mathbf{x}_i^k$, and $\bar{\mathbf{x}}_{T_i}^k$ and $\hat{\mathbf{x}}_{T_i}^k$ as, respectively, each robot i's prior and posterior estimates of the target's state $\mathbf{x}_{T_i}^k$. Their corresponding approximated prior and posterior covariances

are defined as $\bar{\mathbf{p}}_i^k$, $\hat{\mathbf{p}}_i^k$, $\bar{\mathbf{p}}_{T_i}^k$, and $\hat{\mathbf{p}}_{T_i}^k$, respectively.

At time k, if robot j or the target is within the sensing region of robot i, robot i can obtain a robot-to-robot measurement $\mathbf{z}_{R_{ij}}^k$ or a robot-to-target measurement $\mathbf{z}_{R_iT}^k$. The measurement models are defined as

$$\begin{aligned} \mathbf{z}_{R_{ij}}^k &= h_{ij}(\mathbf{x}_i^k, \mathbf{x}_j^k) + \mathbf{v}_{R_{ij}}^k, \\ \mathbf{z}_{R_iT}^k &= h_{iT}(\mathbf{x}_i^k, \mathbf{x}_T^k) + \mathbf{v}_{R_iT}^k, \end{aligned} \tag{2.3}$$

where $\mathbf{v}_{R_{ij}}^k \sim \mathcal{N}(0, \mathbf{R}_{ij}^k)$ and $\mathbf{v}_{R_iT}^k \sim \mathcal{N}(0, \mathbf{R}_{iT}^k)$ are the measurement noises assumed to be zero-mean white Gaussian. The measurement noises are assumed to be mutually uncorrelated across robots and uncorrelated with the process noises.

2.2 Graphs

In the team of M robots, a directed communication graph $G_c^k = (\mathcal{V}, \mathcal{E}_c^k)$ is defined, where $\mathcal{V} = \{R_1, \ldots, R_M\}$ is the robot set and $\mathcal{E}_c^k \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set representing the communication links between robots at time k. If robot i receives information from robot j at time k, then a directed edge (j, i) exists in $\mathcal{E}_c^k$. We assume that the self edge (i, i) exists in $\mathcal{E}_c^k$, $\forall i \in \mathcal{V}$, which means robot i can also use the information from itself. At time k, the communicating neighbor set of robot i is defined as $\mathcal{N}_{c,i}^k = \{l | (l, i) \in \mathcal{E}_c^k, \forall l \neq i, l \in \mathcal{V}\}$. Then, the inclusive communicating neighbor set of robot i is $\mathcal{I}_{c,i}^k = \mathcal{N}_{c,i}^k \cup \{i\}$. Similarly, we define a directed sensing graph $G_s^k = (\mathcal{V}, \mathcal{E}_s^k)$ to describe robot-to-robot measurements, where $\mathcal{E}_s^k \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set, which represents the detection links between robots at time k. If robot i can detect robot j at time k, a directed edge (j, i) exists in $\mathcal{E}_s^k$. At time

k, we define robot i's sensing neighbors (all robots detected by robot i) as $\mathcal{N}_{s,i}^k = \{l|(l,i) \in \mathcal{E}_s^k, \forall l \neq i, l \in \mathcal{V}\}$.

We assume that for each robot, the communication radius is larger than the sensing radii of all robots. Then when robot i detects robot j, robot i can receive the information from robot j.

2.3 Information Fusion Strategy

The consistency is a vital property for estimation. An estimate pair $(\hat{\mathbf{p}}^k,\ \hat{\mathbf{x}}^k)$ is *consistent* if the true error covariance is upper bounded by the estimated covariance as $\mathbb{E}\{(\mathbf{x}^k - \hat{\mathbf{x}}^k)(\mathbf{x}^k - \hat{\mathbf{x}}^k)^\mathsf{T}\} \leq \hat{\mathbf{p}}^k$ [32]. An inconsistent estimate that underestimates the actual errors might eventually diverge. At time k, given multiple consistent estimation pairs $(\hat{\mathbf{p}}_i^k, \hat{\mathbf{x}}_i^k)$, $i = 1, \ldots, n$, of $\mathbf{x}^k$, we seek to compute an improved consistent estimate $(\hat{\mathbf{p}}_c^k, \hat{\mathbf{x}}_c^k)$ by using the Covariance Intersection (CI) algorithm [33]

$$\begin{aligned} \hat{\mathbf{p}}_c^k &= \left(\sum_{i=1}^{n} \alpha_i^k (\hat{\mathbf{p}}_i^k)^{-1} \right)^{-1}, \\ \hat{\mathbf{x}}_c^k &= \hat{\mathbf{p}}_c^k \left(\sum_{i=1}^{n} \alpha_i^k (\hat{\mathbf{p}}_i^k)^{-1} \hat{\mathbf{x}}_i^k \right), \end{aligned} \tag{2.4}$$

where $\alpha_i^k \in [0,1]$ and $\sum_{i=1}^{n} \alpha_i^k = 1$. The parameters α_i^k are usually chosen to satisfy certain optimal criterion such as minimizing the trace of $\hat{\mathbf{p}}_c^k$. In order to reduce the computation burden, we adopt a fast and simplified approach in [34] to calculate α_i^k as

$$\alpha_i^k = \frac{1/\mathrm{tr}(\hat{\mathbf{p}}_i^k)}{\sum_{j=1}^{n} 1/\mathrm{tr}(\hat{\mathbf{p}}_j^k)}, \tag{2.5}$$

where $\mathrm{tr}(\cdot)$ denotes the trace of a matrix.

2.4 Joint Localization and Target Tracking

In this section, we briefly introduce our previous work about JLATT [20]. In this work, each robot can estimate the pose of itself (localization) and the state of a target (tracking) using only its own information and the information from its one-hop communicating neighbors while preserving estimation consistency.

2.4.1 Robot Propagation

The estimate of robot i's state and its corresponding covariance are propagated at time $k-1$ as

$$\begin{aligned} \bar{\mathbf{x}}_i^k &= f_i(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}, 0), \\ \bar{\mathbf{p}}_i^k &= \mathbf{\Phi}_i^{k-1}\hat{\mathbf{p}}_i^{k-1}(\mathbf{\Phi}_i^{k-1})^\mathsf{T} + \bar{\mathbf{Q}}_i^{k-1} \\ &= \rho_{Rp}(\hat{\mathbf{p}}_i^{k-1}, \hat{\mathbf{x}}_i^{k-1}, \mathbf{Q}_i^{k-1}), \end{aligned} \tag{2.6}$$

where $\mathbf{\Phi}_i^{k-1} = \frac{\partial f_i}{\partial \mathbf{x}_i}(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}, 0)$, $\mathbf{G}_i^{k-1} = \frac{\partial f_i}{\partial \mathbf{w}_i}(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}, 0)$, and $\bar{\mathbf{Q}}_i^{k-1} = \mathbf{G}_i^{k-1}\mathbf{Q}_i^{k-1}(\mathbf{G}_i^{k-1})^\mathsf{T}$.

2.4.2 Target Propagation

Robot i propagates its estimate of the target's state and its corresponding covariance at time $k-1$ as

$$\begin{aligned} \bar{\mathbf{x}}_{T_i}^k &= g(\hat{\mathbf{x}}_{T_i}^{k-1}, \mathbf{u}_T^{k-1}, 0), \\ \bar{\mathbf{p}}_{T_i}^k &= \mathbf{\Phi}_{T_i}^k\hat{\mathbf{p}}_{T_i}^{k-1}(\mathbf{\Phi}_{T_i}^{k-1})^\mathsf{T} + \bar{\mathbf{Q}}_{T_i}^{k-1} \\ &= \rho_{Tp}(\hat{\mathbf{p}}_{T_i}^{k-1}, \hat{\mathbf{x}}_{T_i}^{k-1}, \mathbf{Q}_{T_i}^{k-1}), \end{aligned} \tag{2.7}$$

where $\mathbf{\Phi}_{T_i}^{k-1} = \frac{\partial g}{\partial \mathbf{x}_T}(\hat{\mathbf{x}}_{T_i}^{k-1}, \mathbf{u}_T^{k-1}, 0)$, $\mathbf{G}_{T_i}^{k-1} = \frac{\partial g}{\partial \mathbf{w}_T}(\hat{\mathbf{x}}_{T_i}^{k-1}, \mathbf{u}_T^{k-1}, 0)$, and $\bar{\mathbf{Q}}_{T_i}^{k-1} = \mathbf{G}_{T_i}^{k-1}\mathbf{Q}_{T_i}^{k-1}(\mathbf{G}_{T_i}^{k-1})^\mathsf{T}$.

2.4.3 Robot State Update

To update the estimate of robot i's state, we first obtain the correction pairs $(\mathbf{s}^k_{R_{il}}, \mathbf{y}^k_{R_{il}})$ and $(\mathbf{s}^k_{R_iT}, \mathbf{y}^k_{R_iT})$ using robot i's robot-robot measurements $\mathbf{z}^k_{R_{il}}$, $l \in \mathcal{N}^k_{s,i}$, and robot-target measurement $\mathbf{z}^k_{R_iT}$, respectively. These correction pairs are calculated as

$$\mathbf{s}^k_{R_{il}} = (\mathbf{H}^k_{il})^\mathsf{T}(\bar{\mathbf{R}}^k_{il})^{-1}\mathbf{H}^k_{il}, \tag{2.8a}$$

$$\mathbf{y}^k_{R_{il}} = (\mathbf{H}^k_{il})^\mathsf{T}(\bar{\mathbf{R}}^k_{il})^{-1}(\bar{\mathbf{z}}^k_{R_{il}} + \mathbf{H}^k_{il}\bar{\mathbf{x}}^k_i), \tag{2.8b}$$

where $\mathbf{H}^k_{il} = \frac{\partial \mathbf{h}_{il}}{\partial \mathbf{x}^k_i}(\bar{\mathbf{x}}^k_i, \bar{\mathbf{x}}^k_l)$, $\bar{\mathbf{R}}^k_{il} = \mathbf{R}^k_{il} + \widetilde{\mathbf{H}}^k_{il}\bar{\mathbf{p}}^k_l(\widetilde{\mathbf{H}}^k_{il})^\mathsf{T}$, $\widetilde{\mathbf{H}}^k_{il} = \frac{\partial \mathbf{h}_{il}}{\partial \mathbf{x}^k_l}(\bar{\mathbf{x}}^k_i, \bar{\mathbf{x}}^k_l)$, and $\bar{\mathbf{z}}^k_{R_{il}} = \mathbf{z}^k_{R_{il}} - \mathbf{h}_{il}(\bar{\mathbf{x}}^k_i, \bar{\mathbf{x}}^k_l)$, and

$$\mathbf{s}^k_{R_iT} = (\mathbf{H}^k_{iT})^\mathsf{T}(\bar{\mathbf{R}}^k_{iT})^{-1}\mathbf{H}^k_{iT}, \tag{2.9a}$$

$$\mathbf{y}^k_{R_iT} = (\mathbf{H}^k_{iT})^\mathsf{T}(\bar{\mathbf{R}}^k_{iT})^{-1}(\bar{\mathbf{z}}^k_{R_iT} + \mathbf{H}^k_{iT}\bar{\mathbf{x}}^k_i), \tag{2.9b}$$

where $\mathbf{H}^k_{iT} = \frac{\partial \mathbf{h}_{iT}}{\partial \mathbf{x}^k_i}(\bar{\mathbf{x}}^k_i, \bar{\mathbf{x}}^k_{T_i})$, $\bar{\mathbf{R}}^k_{iT} = \mathbf{R}^k_{iT} + \widetilde{\mathbf{H}}^k_{iT}\bar{\mathbf{p}}^k_{T_i}(\widetilde{\mathbf{H}}^k_{iT})^\mathsf{T}$, $\widetilde{\mathbf{H}}^k_{iT} = \frac{\partial \mathbf{h}_{iT}}{\partial \mathbf{x}^k_{T_i}}(\bar{\mathbf{x}}^k_i, \bar{\mathbf{x}}^k_{T_i})$, and $\bar{\mathbf{z}}^k_{R_iT} = \mathbf{z}^k_{R_iT} - \mathbf{h}_{iT}(\bar{\mathbf{x}}^k_i, \bar{\mathbf{x}}^k_{T_i})$.

Then, we apply the CI algorithm (2.4) on these correction pairs to compute a consistent estimate $\breve{\mathbf{x}}_i$ and its corresponding covariance $\breve{\mathbf{p}}_i$ as

$$\breve{\mathbf{p}}^k_i = (\sum_{l \in \mathcal{N}^k_{s,i}} \eta^k_{il}\mathbf{s}^k_{R_{il}} + \eta^k_{iT}\mathbf{s}^k_{R_iT})^{-1}, \tag{2.10a}$$

$$\breve{\mathbf{x}}^k_i = \breve{\mathbf{p}}^k_i(\sum_{l \in \mathcal{N}^k_{s,i}} \eta^k_{il}\mathbf{y}^k_{R_{il}} + \eta^k_{iT}\mathbf{y}^k_{R_iT}), \tag{2.10b}$$

where $\eta^k_{il} \in [0, 1]$, and $\eta^k_{iT} = 0$ if robot i cannot detect the target and otherwise $\eta^k_{iT} \in [0, 1]$ subject to $\sum_{l \in \mathcal{N}^k_{s,i}} \eta^k_{il} + \eta^k_{iT} = 1$. The calculation of these η_{il} and η_{iT} follows the fast and simplified approach (2.5) with $(\mathbf{s}^k_{R_{il}})^{-1}$ and $(\mathbf{s}^k_{R_iT})^{-1}$ playing the role of the covariances.

After we obtain the estimation pair $(\breve{\mathbf{p}}_i^k, \breve{\mathbf{x}}_i^k)$ using relative measurements and also the prior estimation pair $(\bar{\mathbf{p}}_i^k, \bar{\mathbf{x}}_i^k)$ from the robot propagation step, we can use the CI algorithm (2.4) to fuse these two estimation pairs to obtain the posterior estimation pair $(\hat{\mathbf{p}}_i^k, \hat{\mathbf{x}}_i^k)$ as

$$\hat{\mathbf{p}}_i^k = \left(\zeta_{i1}^k (\breve{\mathbf{p}}_i^k)^{-1} + \zeta_{i2}^k (\bar{\mathbf{p}}_i^k)^{-1}\right)^{-1}, \tag{2.11a}$$

$$\hat{\mathbf{x}}_i^k = \hat{\mathbf{p}}_i^k \left(\zeta_{i1}^k (\breve{\mathbf{p}}_i^k)^{-1} \breve{\mathbf{x}}_i^k + \zeta_{i2}^k (\bar{\mathbf{p}}_i^k)^{-1} \bar{\mathbf{x}}_i^k\right), \tag{2.11b}$$

where ζ_{i1}^k and $\zeta_{i2}^k \in [0, 1]$, subject to $\zeta_{i1}^k + \zeta_{i2}^k = 1$, are calculated according to (2.5).

2.4.4 Target State Update

Similarly, for the target state estimation, each robot i first collects the target correction pairs $(\tilde{\mathbf{s}}_{R_jT}^k,\ \tilde{\mathbf{y}}_{R_jT}^k)$ from available robot-target measurements calculated by its inclusive communicating neighbors j, $j \in \mathcal{I}_{c,i}^k$,

$$\tilde{\mathbf{s}}_{R_jT}^k = (\widetilde{\mathbf{H}}_{jT}^k)^{\mathsf{T}} (\widetilde{\mathbf{R}}_{jT}^k)^{-1} \widetilde{\mathbf{H}}_{jT}^k, \tag{2.12a}$$

$$\tilde{\mathbf{y}}_{R_jT}^k = (\widetilde{\mathbf{H}}_{jT}^k)^{\mathsf{T}} (\widetilde{\mathbf{R}}_{jT}^k)^{-1} (\bar{\mathbf{z}}_{R_jT}^k + \widetilde{\mathbf{H}}_{jT}^k \bar{\mathbf{x}}_{T_j}^k), \tag{2.12b}$$

where $\widetilde{\mathbf{R}}_{jT}^k = \mathbf{R}_{jT}^k + \mathbf{H}_{jT}^k \bar{\mathbf{p}}_j^k (\mathbf{H}_{jT}^k)^{\mathsf{T}}$.

Then, at time k, by combining all available target correction pairs, we obtain an intermediate estimation pair as

$$\breve{\mathbf{p}}_{T_i}^k = \Big(\sum_{j \in \mathcal{I}_{c,i}^k} \tilde{\eta}_j^k \tilde{\mathbf{s}}_{R_jT}^k\Big)^{-1}, \tag{2.13a}$$

$$\breve{\mathbf{x}}_{T_i}^k = \breve{\mathbf{p}}_{T_i}^k \Big(\sum_{j \in \mathcal{I}_{c,i}^k} \tilde{\eta}_j^k \tilde{\mathbf{y}}_{R_jT}^k\Big), \tag{2.13b}$$

where $\tilde{\eta}_j^k = 0$ if robot j cannot directly detect the target, and otherwise $\tilde{\eta}_j^k \in [0,1]$ subject to $\sum_{j\in\mathcal{I}_{c,i}^k} \tilde{\eta}_j^k = 1$.

Then we fuse all available prior estimation pairs $(\bar{\mathbf{p}}_{T_j}^k, \bar{\mathbf{x}}_{T_j}^k)$, $\forall j \in \mathcal{I}_{c,i}^k$ with the CI algorithm (2.4) as

$$\check{\mathbf{p}}_{T_i}^k = \left(\sum_{j\in\mathcal{I}_{c,i}^k} \pi_j^k (\bar{\mathbf{p}}_{T_j}^k)^{-1} \right)^{-1}, \tag{2.14a}$$

$$\check{\mathbf{x}}_{T_i}^k = \check{\mathbf{p}}_{T_i}^k \left(\sum_{j\in\mathcal{I}_{c,i}^k} \pi_j^k (\bar{\mathbf{p}}_{T_j}^k)^{-1} \bar{\mathbf{x}}_{T_j}^k \right), \tag{2.14b}$$

where $\pi_j^k \in [0,1]$, subject to $\sum_{j\in\mathcal{I}_{c,i}^k} \pi_j^k = 1$, is calculated according to (2.5).

Eventually, after obtaining the pairs $(\check{\mathbf{p}}_{T_i}^k, \check{\mathbf{x}}_{T_i}^k)$ and $(\breve{\mathbf{p}}_{T_i}^k, \breve{\mathbf{x}}_{T_i}^k)$, we calculate the posterior estimation pair $(\hat{\mathbf{p}}_{T_i}^k, \hat{\mathbf{x}}_{T_i}^k)$ of the target as

$$\hat{\mathbf{p}}_{T_i}^k = \left(\zeta_{iT_1}^k (\check{\mathbf{p}}_{T_i}^k)^{-1} + \zeta_{iT_2}^k (\breve{\mathbf{p}}_{T_i}^k)^{-1} \right)^{-1}, \tag{2.15a}$$

$$\hat{\mathbf{x}}_{T_i}^k = \hat{\mathbf{p}}_{T_i}^k \left(\zeta_{iT_1}^k (\check{\mathbf{p}}_{T_i}^k)^{-1} \check{\mathbf{x}}_{T_i}^k + \zeta_{iT_2}^k (\breve{\mathbf{p}}_{T_i}^k)^{-1} \breve{\mathbf{x}}_{T_i}^k \right), \tag{2.15b}$$

where $\zeta_{iT_1}^k$ and $\zeta_{iT_2}^k \in [0,1]$, subject to $\zeta_{iT_1}^k + \zeta_{iT_2}^k = 1$, are calculated according to (2.5).

Chapter 3

Active Joint Localization and Target Tracking

While our previous work [20] achieves fully distributed JLATT, the robots' motions are not actively controlled or planned and they simply move around randomly. As a result, the full potential to improve the localization and target tracking performance is not exploited. In this thesis, we aim to actively control the motions of the robots to achieve better localization and target tracking performance than random movements. We use the term AJLATT to emphasise the fact that we try to not only estimate the states of the robots and the target but also design proper control actions to improve the estimation performance. Our goal is to design the control input $\mathbf{u}_i^k$ for each robot i modeled by (2.1) at each time k in a fully distributed manner by using only its own information and the information from its one-hop communicating neighbors. Next, we present two approaches, one control based and the other optimization based.

An overview of the proposed AJLATT algorithm is summarized in Table 3.1. In **Step 3**, we either implement our control-based AJLATT algorithm in **Step 3 (1)** using (3.2) or optimization-based AJLATT algorithm in **Step 3 (2)** using (3.7) to solve the AJLATT problem. Detailed explanations for these two algorithms are given in the following subsections 3.1 and 3.2.

3.1 Control-based Approach

In order to obtain a more accurate estimate of the target's state, one way is to approach the target. The first reason is that the detection range of a sensor is limited in general. Besides, the measurement noise usually has a positive relationship with the measurement distance with the detected object during a certain range interval. Second, when the robots move closer to the target, they are closer to other robots, which will also intuitively help improve self-localization. While the distributed tracking problem has been studied in the controls community extensively (see, e.g., [21]), most of the works assume the poses of the robots and even the target are known (no estimation needed), which is not realistic. Therefore, in this subsection, we propose a fully distributed algorithm to solve the AJLATT problem from the perspective of control.

Without loss of generality, we first assume robot i's model as

$$\mathbf{r}_i^k = \mathbf{r}_i^{k-1} + \mathbf{u}_i^{k-1}\delta t, \tag{3.1}$$

where $\mathbf{r}_i^{k-1} = [x_i^{k-1}, y_i^{k-1}]^T$ is the position in 2D at time $k-1$, $\mathbf{u}_i^{k-1} = [u_{xi}^{k-1}, u_{yi}^{k-1}]^T$ is the control input in 2D at time $k-1$, and δt is the sampling interval. We will later adapt

the model and design to address more realistic robot models (e.g., nonholonomic differential drive robots). Let $\mathbf{r}_T^{k-1}$ and $\mathbf{v}_T^{k-1}$ denote, respectively, the target's position and velocity at time $k-1$. Also let $\hat{\mathbf{r}}_i^{k-1}$, $\hat{\mathbf{r}}_{T_i}^{k-1}$, and $\hat{\mathbf{v}}_{T_i}^{k-1}$ denote, respectively, robot i's estimate of its own position, the target's position, and the target's velocity. Then we design the control $\mathbf{u}_i^{k-1}$ for (3.1) as

$$\begin{aligned}\mathbf{u}_i^{k-1} = &\sum_{j\in\mathcal{I}_{c,i}^{k-1}} \eta_j^{k-1}\hat{\mathbf{v}}_{T_j}^{k-1} - \alpha_i\Big(\sum_{m\in\mathcal{N}_{c,i}^{k-1}} \frac{\partial V_{im}}{\partial \hat{\mathbf{r}}_i^{k-1}} \\ &+ \sum_{j\in\mathcal{I}_{c,i}^{k-1}} \frac{\partial V_{iT_j}}{\partial \hat{\mathbf{r}}_i^{k-1}}\Big) - \gamma_i \sum_{j\in\mathcal{I}_{c,i}^{k-1}} \eta_j^{k-1}(\hat{\mathbf{r}}_i^{k-1} - \hat{\mathbf{r}}_{T_j}^{k-1}),\end{aligned} \tag{3.2}$$

where $\hat{\mathbf{v}}_{T_j}^{k-1}$ is obtained using the known target input $\mathbf{u}_T^{k-1}$ and estimated target's state $\hat{\mathbf{x}}_{T_j}^{k-1}$ as we will detail later in Section 4, α_i and γ_i are two positive constants which are used to adjust, respectively, the influence of the collision avoidance term and active target tracking term,

$$\eta_j^{k-1} = \frac{1/\mathrm{tr}(\hat{\mathbf{p}}_{T_j}^{k-1})}{\sum_{l\in\mathcal{I}_{c,i}^{k-1}} 1/\mathrm{tr}(\hat{\mathbf{p}}_{T_l}^{k-1})} \tag{3.3}$$

is the weight denoting the certainty of neighbor j's estimate of the target's state from the inclusive communicating neighbor set of robot i, and the differentiable, nonnegative functions V_{im} and V_{iT_j} are, respectively, the potential function defined on $\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_m\|$ and $\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_{T_j}\|$ for collision avoidance as detailed later.

The motivation behind (3.2) is to push each robot toward the weighted average of its own and communicating neighbors' estimates of the target's position while avoiding collisions with the estimated positions of its communicating neighbors as well as the estimated positions of the target on itself and its communicating neighbors. The weight η_j is selected according to the certainty of the estimates of the target's position. If the certainty of a neighbor is higher, then a larger weight is assigned.

The potential function V_{ij} has the following properties:

1. V_{ij} achieves its unique minimum when $\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\|$ is equal to its desired value d_{o_i} .

2. $V_{ij} \to \infty$ if $\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| \to d_{s_i}$, where d_{s_i} $(0 < d_{s_i} < d_{o_i})$ is the safe distance for collision avoidance.

3. $\partial V_{ij}/\partial\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| = \mathbf{0}$, if $\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| \geqslant R_i$, where $R_i > d_{o_i}$ denotes the communication radius of robot i.

Motivated by [21], we choose V_{ij} such that

$$\frac{\partial V_{ij}}{\partial \hat{\mathbf{r}}_i} = \begin{cases} -\infty \mathrm{sgn}(\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j) & \|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| < d_{s_i} \\ 20\frac{(\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j)}{\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\|}\frac{\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| - d_{o_i}}{\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| - d_{s_i}}, & d_{s_i} \leqslant \|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| < d_{o_i} \\ 0.5\frac{(\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j)\sin[\frac{\pi}{R_i - d_{o_i}}(\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| - d_{o_i})]}{\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\|}, & d_{o_i} \leqslant \|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| < R_i \\ \mathbf{0}, & \text{otherwise,} \end{cases} \tag{3.4}$$

where $\mathrm{sgn}(\cdot)$ denotes the sign function defined component-wise. The potential function V_{iT_j} is defined analogically. An example of V_{ij} is shown is shown in Figure 3.1. Note that unlike [21], we do not have the actual positions of the robots and the target, $\mathbf{r}_i$ and $\mathbf{r}_T$, and hence the actual distances between robots and between robot and target. Therefore, the potential functions are defined on the estimated relative distance between robots $\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_m\|$ and that between robot and target $\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_{T_j}\|$ instead.

The control input (3.2) is defined according to a simplified model (3.1). However, the model might not be applicable to more realistic models. Next, we use the commonly used unicycle model as an example to show how we extend the calculated control input

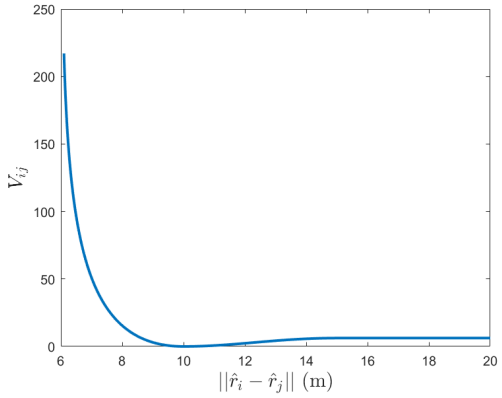


Figure 3.1: Potential function V_{ij} with $d_{s_i} = 6$ m, $d_{o_i} = 10$ m, and $R_i = 15$ m.

$\mathbf{u}_i^{k-1} = [u_{xi}^{k-1}, u_{yi}^{k-1}]^T$ in (3.2) to design the linear and angular velocity inputs in the unicycle model.

Consider a unicycle model described as

$$\begin{aligned} x_i^k &= x_i^{k-1} + v_i^{k-1}\delta t\cos(\theta_i^{k-1}), \\ y_i^k &= y_i^{k-1} + v_i^{k-1}\delta t\sin(\theta_i^{k-1}), \\ \theta_i^k &= \theta_i^{k-1} + \omega_i^{k-1}\delta t, \end{aligned} \tag{3.5}$$

where (x_i^{k-1}, y_i^{k-1}) is the position, θ_i^{k-1} is the orientation, v_i^{k-1} is the linear velocity, and ω_i^{k-1} is the angular velocity associated with robot i at time $k-1$. With $\mathbf{u}_i^{k-1} = [u_{xi}^{k-1}, u_{yi}^{k-1}]^T$ given by (3.2), we can calculate the desired linear velocity as

$$v_{i_d}^{k-1} = \sqrt{(u_{xi}^{k-1})^2 + (u_{yi}^{k-1})^2},$$

and the desired orientation as

$$\theta_{i_d}^{k-1} = \text{atan2}(u_{yi}^{k-1}, u_{xi}^{k-1}).$$

Then we can design the linear and angular velocity control input as

$$\begin{aligned} v_i^{k-1} &= {v_i}_d^{k-1}, \\ \omega_i^{k-1} &= -\lambda_i(\theta_i^{k-1} - {\theta_i}_d^{k-1}), \end{aligned} \tag{3.6}$$

where λ_i is a positive constant.

As we can see here, the control-based approach is computationally simple and time efficient in real-world implementations. It is worth noticing that the control policy here is not designed to explicitly optimize a certain criterion on the localization and target tracking performance. Instead, the hope is to bring the robots closer so as to improve the localization and target tracking performance. Another thing is that here we use a relatively simple unicycle model. Therefore, the original calculated control input for the simplified model can be transformed into the actual control command for the robot. However, as we can imagine, as the models of the robots become more complicated, it will be much harder to directly design controllers or transform controller design for simplified models to control commands applicable to real robots. Therefore, in the next subsection, we will introduce an optimization-based AJLATT algorithm, which is more performance-improvement-orientated and applicable to more general robot models.

3.2 Optimization Based Approach

In this subsection, we propose an optimization-based approach to solve the AJLATT problem. Our goal is to find an optimal control input $\mathbf{u}_i^{k-1}$ for each robot i modeled by (2.1), to optimize certain functions.

There exist some previous works solving a related problem from the optimization

perspective. However, they either ignore the robot localization [24–28] or are implemented in a centralized manner [29,30]. In contrast, here the robot localization is explicitly considered and the problem is solved in a fully distributed manner by using each robot's own and its one-hop neighbors' information. There is no center node required for computation, global parameter shared among the entire team, or information transmitted via multiple hops.

Our optimization-based AJLATT algorithm can be summarized as

$$\begin{aligned} \mathbf{u}_i^{k-1} = \quad & \underset{\mathbf{u}_i^{k-1}\in\mathcal{U}_i}{\arg\min}\ \alpha_{R_i}\mathrm{tr}(\bar{\mathbf{p}}_i^{k+t_2}) + \alpha_{T_i}\mathrm{tr}(\bar{\mathbf{p}}_{T_i}^{k+t_2}) \\ & +\beta_i \textstyle\sum_{j\in\mathcal{N}_{c,i}^{k-1}\cup\{T_i\}} J_{R_{ij}}^{k+t_1}, i = 1,\ldots,M \\ \text{s.t.} \quad & (\bar{\mathbf{x}}_j^{k+t_1}, \bar{\mathbf{x}}_{T_j}^{k+t_1}, \bar{\mathbf{p}}_i^{k+t_2}, \bar{\mathbf{p}}_{T_i}^{k+t_2}) = \\ & SACP(\hat{\mathbf{x}}_j^{k-1}, \hat{\mathbf{x}}_{T_j}^{k-1}, \hat{\mathbf{p}}_j^{k-1}, \hat{\mathbf{p}}_{T_j}^{k-1}, \mathbf{u}_j^{k-2}, \mathbf{u}_T^{k-1}),\ j \in \mathcal{I}_{c,i}^{k-1} \end{aligned} \tag{3.7}$$

where $\mathcal{U}_i$ is the control space of robot i. α_{R_i}, α_{T_i}, and β_i are three positive constant parameters that are used to adjust the influence of each term on the objective function. $\bar{\mathbf{p}}_i^{k+t_2}$ and $\bar{\mathbf{p}}_{T_i}^{k+t_2}$ are, respectively, the predicted robot and target covariances at time $k+t_2$ representing, respectively, robot i's predicted self-localization and target tracking uncertainty. $J_{R_{ij}}^{k+t_1}$, $j \in \mathcal{N}_{c,i}^{k-1}\cup\{T_i\}$, is the potential function term at time $k+t_1$, which is used to maintain communication connectivity, avoid collisions, and keep sight of the target. t_1 and t_2 are, respectively, the planning horizons for the potential function term and covariance terms. In general, we have $t_1 < t_2$, since the difference of uncertainty between different control inputs needs more time to show up, but the potential function term that helps robots to quickly avoid collision and frequently keep communication connectivity needs to be computed in a shorter time period. SACP is a function that is used to predict the estimate of the states

$\bar{\mathbf{x}}_j^{k+t_1}, \bar{\mathbf{x}}_{T_j}^{k+t_1}$ and covariances $\bar{\mathbf{p}}_i^{k+t_2}, \bar{\mathbf{p}}_{T_i}^{k+t_2}$ at time $k+t_1$ and $k+t_2$, respectively, and will be shown in detail later.

Let $\mathbf{r}_i = [x_i, y_i]^T$ be the position part of the state $\hat{\mathbf{x}}_i$ in (2.1). For each robot i, the potential function $J_{R_{ij}}$ is defined as

$$J_{R_{ij}} = \begin{cases} \infty\ , & \|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \leqslant \underline{d}_i \\ -10\log(\frac{\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| - \underline{d}_i}{a_i}), & \underline{d}_i < \|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \\ & \qquad \leqslant \underline{d}_i + a_i \\ 0, & \underline{d}_i + a_i < \|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \\ & \qquad \leqslant \bar{d}_i - a_i \\ 10(\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| - (\bar{d}_i - a_i))^2, & \|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| > \bar{d}_i - a_i \end{cases} \tag{3.8}$$

where $\bar{\mathbf{r}}_i = [\bar{x}_i, \bar{y}_i]^T$ is the prior estimate of $\mathbf{r}_i$, $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\|$, $j \in \mathcal{N}_{c,i} \cup \{T_i\}$, is the estimated robot-robot or robot-target distance, $\underline{d}_i$ and $\bar{d}_i$ are, respectively, the minimum and maximum acceptable distances that are used to avoid collisions and maintain communication connectivity, and a_i is the length of the non-zero interval. Note that the definition of $J_{R_{ij}}$ accommodates both robot-robot potential and robot-target potential. An example of the potential function is shown in Figure 3.2.

According to the definition of the potential function $J_{R_{ij}}$, there are several properties that we would like to point out:

1. $J_{R_{ij}} = 0$, when $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \in [\underline{d}_i + a_i, \bar{d}_i - a_i]$, which means that only when other robots or the target moves too close to the minimum range or almost moves out of the maximum acceptable range, this potential function will play a role. Otherwise, only the covariance terms take effect in the objective function in (3.7).

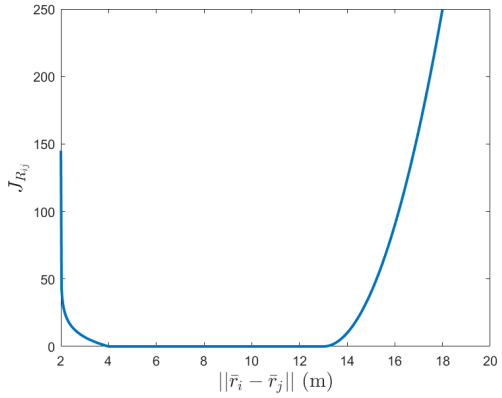


Figure 3.2: Potential function $J_{R_{ij}}$ with $\underline{d}_i = 2$ m, $\bar{d}_i = 15$ m, and $a_i = 2$ m. Notice that the right-hand side increases more moderately than the left-hand side, and is still defined when $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| > \bar{d}_i$.

2. $J_{R_{ij}} \to \infty$, if $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \to \underline{d}_i$ or $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \to \infty$. As shown in Figure 3.2, the potential function will immediately give a large penalty (close to infinity) when the robot comes too close to its neighbors or the target to avoid collisions. In contrast, it gives a relatively soft and weak penalty when the robot moves far away from its neighbors or the target so as to maintain communication connectivity or not lose sight of the target, respectively.

-

The State and Covariance Prediction (SACP) process is first used to predict robot i's estimates of its own and the target's states, and robot i's estimates of each neighbor j's state at time $k + t_1$ so as to calculate the potential function term $J_{R_{ij}}^{k+t_1}$. The process is consecutively used to predict robot i's covariance $\bar{\mathbf{p}}_i^{k+t_2}$ and the target covariance $\bar{\mathbf{p}}_{T_i}^{k+t_2}$ at time $k + t_2$ which constitute the uncertainty minimization terms in the objective function

in (3.7). The SACP process is shown in Table 3.2.

Remark 1. *Since* $\mathbf{u}_j^{k-1}$ *is the optimization variable for each robot* j*, robot* i *propagates its neighbor* j*'s state estimate with the neighbor's latest available control input* $\mathbf{u}_j^{k-2}$ *at time* $k-2$*. Robot* i*'s own state is propagated with* $\mathbf{u}_i^{k-1}$*. The target's state is propagated with* $\mathbf{u}_T^{k-1}$*, which is assumed to be known for each robot at time* $k-1$*.*

Remark 2. *In* ***Step 2*** *in Table 3.2,* ${}^*\mathcal{N}_{s,i}^k$ *and* ${}^*\mathcal{I}_{c,i}^k$ *are two predicted neighbor sets at time* k *in the prediction process. Since the real movement does not happen, robot* i *can only utilize the information from its communicating neighbors at time* $k-1$*. Thus* ${}^*\mathcal{N}_{s,i}^k$ *and* ${}^*\mathcal{I}_{c,i}^k$ *are the subsets of the known set* $\mathcal{I}_{c,i}^{k-1}$*, where* ${}^*\mathcal{N}_{s,i}^k = \{i|(l,i) \in ({}^*\mathcal{E}_s^k \cap \mathcal{E}_c^{k-1}), \forall l \neq i, l \in \mathcal{V}\}$ *and* ${}^*\mathcal{I}_{c,i}^k = \{i|(l,i) \in ({}^*\mathcal{E}_c^k \cap \mathcal{E}_c^{k-1}), l \in \mathcal{V}\}$*. Here* ${}^*\mathcal{E}_s^k$ *and* ${}^*\mathcal{E}_c^k$ *are, respectively, the predicted sensing and communication edge set at time* k *which are determined by the predicted estimates of the robots' states* $\bar{\mathbf{x}}_j^k$ *together with, respectively, the robots' sensing fields of view and communication radii. In addition,* η_{iT}^k *in* (2.10a) *is determined by* $\bar{\mathbf{x}}_i^k$ *and the estimate of the target's state* $\bar{\mathbf{x}}_{T_i}^k$ *together with the robots' sensing fields of view.*

Remark 3. *In* ***Step 4*** *in Table 3.2, each robot* i *propagates its own state estimate and its estimate of the target's state up to time* $k+t_1$*. For each of robot* i*'s communicating neighbors at time* $k-1$ *(*$j \in \mathcal{N}_{c,i}^{k-1}$*), robot* i *also propagates neighbor* j*'s state estimate up to time* $k+t_1$*. The potential function term in* (3.7) *would result in a larger penalty when robot* i *loses communication connectivity with its neighbor* j*,* $j \in \mathcal{N}_{c,i}^{k-1}$*, at time* $k+t_1$*. Instead, if the potential function term were given by* $\beta_i \sum_{j \in {}^*\mathcal{N}_{c,i}^{k+t_1} \cup \{T_i\}} J_{R_{ij}}^{k+t_1}$ *in (22), then a smaller value would be obtained due to the removal of robot* j *in* ${}^*\mathcal{N}_{c,i}^{k+t_1}$*.*

Table 3.1: AJLATT by Robot i

Initialization:

1 Initialize $\hat{\mathbf{x}}_i^0$,$\hat{\mathbf{p}}_i^0$, $\hat{\mathbf{x}}_{T_i}^0$, $\hat{\mathbf{p}}_{T_i}^0$, and set $u_i^{-1} = 0$.

At time $k-1$

Information Exchange for AJLATT:

2.1 Send $\hat{\mathbf{x}}_i^{k-1}$, $\hat{\mathbf{p}}_i^{k-1}$, $\mathbf{u}_i^{k-2}$, $f_i(\cdot)$, $\hat{\mathbf{x}}_{T_i}^{k-1}$, $\hat{\mathbf{p}}_{T_i}^{k-1}$ to robot j, $i \in \mathcal{N}_{c,j}^{k-1}$.

2.2 Receive $\hat{\mathbf{x}}_j^{k-1}$, $\hat{\mathbf{p}}_j^{k-1}$, $\mathbf{u}_j^{k-2}$, $f_j(\cdot)$, $\hat{\mathbf{x}}_{T_j}^{k-1}$, $\hat{\mathbf{p}}_{T_j}^{k-1}$ from robot j, $j \in \mathcal{N}_{c,i}^{k-1}$.

AJLATT Motion Planning:

3 (1) Calculate $\mathbf{u}_i^{k-1}$ using the control-based AJLATT algorithm (3.2).

3 (2) Calculate $\mathbf{u}_i^{k-1}$ using the optimization-based AJLATT algorithm (3.7).

Propagation:

4 Propagate robot i's estimate of its own state $\hat{\mathbf{x}}_i^{k-1}$ and covariance $\hat{\mathbf{p}}_i^{k-1}$ with the obtained $\mathbf{u}_i^{k-1}$ using (2.6) and the target's state $\hat{\mathbf{x}}_{T_i}^{k-1}$ and covariance $\hat{\mathbf{p}}_{T_i}^{k-1}$ using (2.7) to obtain $\bar{\mathbf{x}}_i^k$, $\bar{\mathbf{p}}_i^k$, $\bar{\mathbf{x}}_{T_i}^k$ and $\bar{\mathbf{p}}_{T_i}^k$.

At time k

Update:

5.1 Obtain the robot-robot measurements $\mathbf{z}_{R_{il}}^k$, $l \in \mathcal{N}_{s,i}^k$, and robot-target measurement $\mathbf{z}_{R_iT}^k$ (if the target is detected by robot i) and generate the corresponding correction pairs $(\mathbf{s}_{R_{il}}^k, \mathbf{y}_{R_{il}}^k)$, $(\mathbf{s}_{R_iT}^k, \mathbf{y}_{R_iT}^k)$, $(\tilde{\mathbf{s}}_{R_iT}^k, \tilde{\mathbf{y}}_{R_iT}^k)$ using (2.8), (2.9), (2.12).

5.2 Send $(\tilde{\mathbf{s}}_{R_iT}^k, \tilde{\mathbf{y}}_{R_iT}^k)$, $(\bar{\mathbf{p}}_{T_i}^k, \bar{\mathbf{x}}_{T_i}^k)$ to robot j, $i \in \mathcal{N}_{c,j}^k$.

5.3 Receive $(\tilde{\mathbf{s}}_{R_jT}^k, \tilde{\mathbf{y}}_{R_jT}^k)$, $(\bar{\mathbf{p}}_{T_j}^k, \bar{\mathbf{x}}_{T_j}^k)$ from robot j, $j \in \mathcal{N}_{c,i}^k$.

5.4 Calculate the posterior robot estimate pair $(\hat{\mathbf{p}}_i^k, \hat{\mathbf{x}}_i^k)$ using (2.10), (2.11), and the posterior target estimate pair $(\hat{\mathbf{p}}_{T_i}^k, \hat{\mathbf{x}}_{T_i}^k)$ using (2.13), (2.14), (2.15).

Table 3.2: State and Covariance Prediction (SACP) by Robot i at $k-1$

One-step Propagation and Predicted Update:

1 Propagate each inclusive communicating neighbor j's, $j \in \mathcal{I}_{c,i}^{k-1}$, estimates of its own and the target's states and covariances in one step:

$$\bar{\mathbf{x}}_i^k = f_i(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}, 0), \quad \bar{\mathbf{p}}_i^k = \rho_{Rp}(\hat{\mathbf{p}}_i^{k-1}, \hat{\mathbf{x}}_i^{k-1}, \mathbf{Q}_i^{k-1}),$$

$$\bar{\mathbf{x}}_j^k = f_j(\hat{\mathbf{x}}_j^{k-1}, \mathbf{u}_j^{k-2}, 0), \quad \bar{\mathbf{p}}_j^k = \rho_{Rp}(\hat{\mathbf{p}}_j^{k-1}, \hat{\mathbf{x}}_j^{k-1}, \mathbf{Q}_j^{k-1}), \quad j \in \mathcal{N}_{c,i}^{k-1}$$

$$\bar{\mathbf{x}}_{T_j}^k = g(\hat{\mathbf{x}}_{T_j}^{k-1}, \mathbf{u}_T^{k-1}, 0), \quad \bar{\mathbf{p}}_{T_j}^k = \rho_{Tp}(\hat{\mathbf{p}}_{T_j}^{k-1}, \hat{\mathbf{x}}_{T_j}^{k-1}, \mathbf{Q}_{T_j}^{k-1}), \quad j \in \mathcal{I}_{c,i}^{k-1}$$

2 Generate one-step predicted correction terms $\mathbf{s}_{R_{il}}^k$, $l \in {}^*\mathcal{N}_{s,i}^k$, $\mathbf{s}_{R_iT}^k$, $\tilde{\mathbf{s}}_{R_jT}^k$, $j \in {}^*\mathcal{I}_{c,i}^k$, using (2.8a), (2.9a), (2.12a).

3 Calculate one-step predicted posterior robot covariance $\hat{\mathbf{p}}_i^k$ using (2.10a), (2.11a), and one-step predicted posterior target covariance $\hat{\mathbf{p}}_{T_i}^k$ using (2.13a), (2.14a), (2.15a).

Propagation in t_1 and t_2 Planning Horizons:

4 Propagate robot i's estimate of the target's state and corresponding covariance and each inclusive communicating neighbor j's, $j \in \mathcal{I}_{c,i}^{k-1}$, estimates of its own state and corresponding covariance for the **potential function term** in t_1 planning horizon:

$t = 0, \ldots, t_1 - 1,$

$$\bar{\mathbf{x}}_{T_i}^{k+t+1} = g(\bar{\mathbf{x}}_{T_i}^{k+t}, \mathbf{u}_T^{k-1}, 0), \ \bar{\mathbf{x}}_i^{k+t+1} = f_i(\bar{\mathbf{x}}_i^{k+t}, \mathbf{u}_i^{k-1}, 0), \ \bar{\mathbf{x}}_j^{k+t+1} = f_j(\bar{\mathbf{x}}_j^{k+t}, \mathbf{u}_j^{k-2}, 0), j \in \mathcal{N}_{c,i}^{k-1}$$

$$\begin{cases} \bar{\mathbf{p}}_{T_i}^{k+t+1} = \rho_{Tp}(\hat{\mathbf{p}}_{T_i}^{k+t}, \bar{\mathbf{x}}_{T_i}^{k+t}, \mathbf{Q}_{T_i}^{k+t}), & \text{if } t = 0 \\ \bar{\mathbf{p}}_{T_i}^{k+t+1} = \rho_{Tp}(\bar{\mathbf{p}}_{T_i}^{k+t}, \bar{\mathbf{x}}_{T_i}^{k+t}, \mathbf{Q}_{T_i}^{k+t}), & \text{if } t \geqslant 1 \end{cases} \quad \begin{cases} \bar{\mathbf{p}}_i^{k+t+1} = \rho_{Rp}(\hat{\mathbf{p}}_i^{k+t}, \bar{\mathbf{x}}_i^{k+t}, \mathbf{Q}_i^{k+t}), \text{ if } t = 0 \\ \bar{\mathbf{p}}_i^{k+t+1} = \rho_{Rp}(\bar{\mathbf{p}}_i^{k+t}, \bar{\mathbf{x}}_i^{k+t}, \mathbf{Q}_i^{k+t}), \text{ if } t \geqslant 1 \end{cases}$$

Compute $J_{R_{ij}}^{k+t_1}$ using $\|\bar{\mathbf{r}}_i^{k+t_1} - \bar{\mathbf{r}}_j^{k+t_1}\|$, $j \in \mathcal{N}_{c,i}^{k-1} \cup \{T_i\}$.

5 Propagate robot i's estimates of its own and the target's states and corresponding covariances to obtain the **covariance terms** in t_2 planning horizon

$t = t_1, \ldots, t_2 - 1,$

$$\bar{\mathbf{x}}_i^{k+t+1} = f_i(\bar{\mathbf{x}}_i^{k+t}, \mathbf{u}_i^{k-1}, 0), \qquad \bar{\mathbf{x}}_{T_i}^{k+t+1} = g(\bar{\mathbf{x}}_{T_i}^{k+t}, \mathbf{u}_T^{k-1}, 0),$$

$$\bar{\mathbf{p}}_i^{k+t+1} = \rho_{Rp}(\bar{\mathbf{p}}_i^{k+t}, \bar{\mathbf{x}}_i^{k+t}, \mathbf{Q}_i^{k+t}), \quad \bar{\mathbf{p}}_{T_i}^{k+t+1} = \rho_{Tp}(\bar{\mathbf{p}}_{T_i}^{k+t}, \bar{\mathbf{x}}_{T_i}^{k+t}, \mathbf{Q}_{T_i}^{k+t}).$$

Chapter 4

Simulation

In this section, we will use Monte-Carlo simulations to demonstrate the performance of our algorithms.

4.1 Simulation Setup

Consider the scenario where $M = 6$ robots and a target move on a surface. Here we adopt the widely used unicycle model for both robots and the target in the simulation. The robot pose $\mathbf{x}_i^k$ consists of the position (x_i^k, y_i^k) and the orientation θ_i^k in the global frame. The motion models (2.1) and (2.2) can be expressed as

$$\begin{aligned} x_i^k &= x_i^{k-1} + (v_i^{k-1} + w_{v_i}^{k-1})\delta t \cos(\theta_i^{k-1}), \\ y_i^k &= y_i^{k-1} + (v_i^{k-1} + w_{v_i}^{k-1})\delta t \sin(\theta_i^{k-1}), \\ \theta_i^k &= \theta_i^{k-1} + (\omega_i^{k-1} + w_{\omega_i}^{k-1})\delta t, \end{aligned} \tag{4.1}$$

where $i \in \{1, \ldots, M\} \cup \{T\}$, $\delta t = 1$ s is the sampling interval, $\mathbf{u}_i^{k-1} = [v_i^{k-1},\ \omega_i^{k-1}]^T$ represents the linear and angular velocities as the input for robot i, and $\mathbf{w}_i = [w_{v_i}^{k-1}, w_{\omega_i}^{k-1}]^T$

represents process noises for the linear and angular velocities. For robot i, the input $\mathbf{u}_i^{k-1}$ is calculated by our AJLATT algorithms. The target's input is assumed to be constant with $v_T = 0.25$ m/s and $\omega_T = 0$. The target's input $\mathbf{u}_T = [v_T,\, \omega_T]^T$ is known by every robot i. The corresponding process noise $\mathbf{w}_i$, $i \in \{1, \ldots, M\} \cup \{T\}$, is assumed to be white Gaussian, with the standard deviations for $w_{v_i}^{k-1}$ and $w_{\omega_i}^{k-1}$ as, respectively, $\sigma_{v_i}^{k-1} = \frac{\sqrt{2}}{2}\sigma_i^{k-1}$ and $\sigma_{\omega_i}^{k-1} = 2\sqrt{2}\sigma_i^{k-1}$, where σ_i^{k-1} is proportional to the linear velocity as $\sigma_i^{k-1} = 1\% v_i^{k-1}$ for each robot i, $i \in \{1, \ldots, M\}$, and $\sigma_T^{k-1} = 3\% v_T^{k-1}$ for the target. Hence $\mathbf{Q}_i^{k-1}$, $i \in \{1, \ldots, M\} \cup \{T\}$ defined after (2.1) and (2.2) is given as

$$\mathbf{Q}_i^{k-1} = \begin{bmatrix} (\sigma_{v_i}^{k-1})^2 & 0 \\ 0 & (\sigma_{\omega_i}^{k-1})^2 \end{bmatrix}. \tag{4.2}$$

Given the model (4.1), it follows from (2.6) that

$$\mathbf{\Phi}_i^{k-1} = \begin{bmatrix} 1 & 0 & -v_i^{k-1}\delta t \sin(\theta_i^{k-1}) \\ 0 & 1 & v_i^{k-1}\delta t \cos(\theta_i^{k-1}) \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.3}$$

According to (4.2) and (4.3), when robot i stops moving (i.e. $v_i = 0$), $\mathbf{Q}_i^{k-1}$ and $\mathbf{\Phi}_i^{k-1}$ will become, respectively, the zero matrix $\mathbf{0}_{2\times 2}$ and the identity matrix I_3. As a result, it follows from (2.6) that $\bar{\mathbf{p}}_i^k = \hat{\mathbf{p}}_i^{k-1}$, which means that the robot covariance will not increase during propagation.

We assume that each robot has a limited communication range with a radius of $R_i = 30$ m and a limited field of view with $R_{min} = 2$ m and $R_{max} = 15$ m for the range and $\phi = 60^\circ$ for the angle of view. As for the measurement model, we consider an indoor application scenario in this work and assume that these robots do not have access to the ab-

solute position measurement. The relative distance-bearing measurement model is adopted for each robot in our simulation. If robot i detects robot j at time instant k, then the relative measurement can be expressed as

$$\mathbf{z}_{R_{ij}}^k = \begin{bmatrix} \sqrt{(x_j^k - x_i^k)^2 + (y_j^k - y_i^k)^2} \\ \text{atan2}((y_j^k - y_i^k), (x_j^k - x_i^k)) - \theta_i^k \end{bmatrix} + \mathbf{v}_{R_{ij}}^k,$$

where $\mathbf{v}_{R_{ij}}$ is a zero-mean white Gaussian noise. The standard deviation of the distance noise is set to be 3% of the actual distance, and the standard deviation of the bearing noise equals to 1°. The same measurement model is used for the robot-to-target measurement $\mathbf{z}_{R_iT}$.

Since the absolute measurement is not available, we assume that each robot initializes its estimated pose $\hat{\mathbf{x}}_i^0$ with its true pose $\mathbf{x}_i^0$, and the initial pose covariance $\hat{\mathbf{p}}_i^0$ is set to $\hat{\mathbf{p}}_i^0 = 10^{-3}\mathbf{I}_3$. The initial estimate of the target's state obtained by each robot i, $\hat{\mathbf{x}}_{T_i}^0$, does not necessarily equal to the true initial state of the target $\mathbf{x}_T^0$. In our simulation, we set $\hat{\mathbf{x}}_T^0 = [10, 10, 0]$ while the true initial target state is $\mathbf{x}_T^0 = [10, 5, 0]$. Since we assume that an accurate initial target state is not available, we initialize the target covariance with relatively large uncertainty as $\hat{\mathbf{p}}_{T_i}^0 = \mathbf{I}_3$.

We compare the performance of the following three algorithms.

- *Random Motion (RM)*: In this case, each robot moves with a constant linear velocity of $v_i = 0.5$ m/s. Its angular velocity ω_i is uniformly chosen from an interval of $[-\frac{\pi}{5}, \frac{\pi}{5}]$ rad/s. These robots behave as in our previous work [20] except that there is no moving field boundary for them. By adopting the RM strategy, robots do not tend to pursue the target or maintain communication connectivity with other robots, which eventually

results in worse localization and tracking performance than other AJLATT algorithms as shown later.

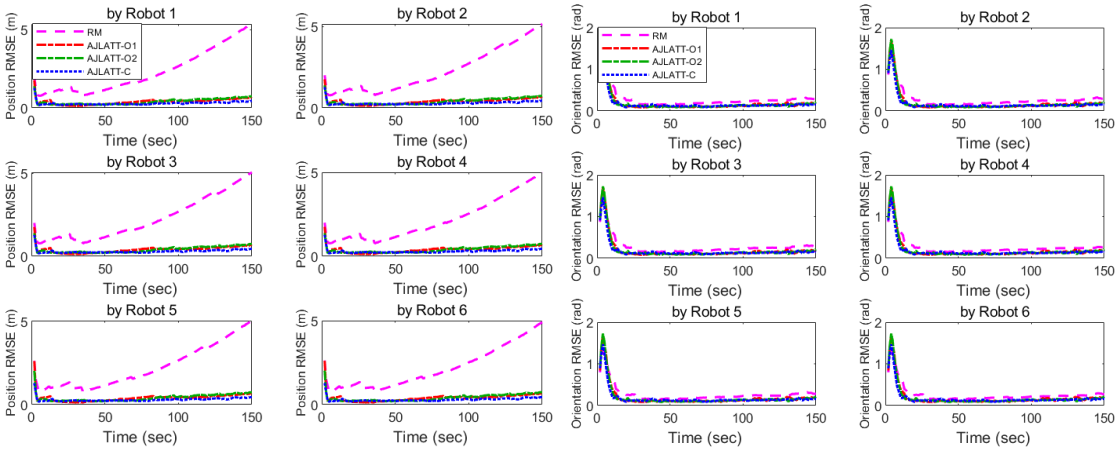
- *Control-based AJLATT (AJLATT-C)*: The control based AJLATT algorithm uses the control policy in subsection 3.1. The parameters as set as $\alpha_i = 0.015$, $\gamma_i = 1$ in (3.2), $d_{s_i} = 6$ m, $d_{o_i} = 10$ m in (3.4), and $\lambda_i = 1$ in (3.6). The estimated target velocity $\hat{\mathbf{v}}_{T_j}$ in (3.2) is calculated by $\hat{\mathbf{v}}_{T_j} = v_T \begin{bmatrix} \cos(\hat{\theta}_{T_j}) & \sin(\hat{\theta}_{T_j}) \end{bmatrix}^T$, where v_T is the known target's velocity, and $\hat{\theta}_{T_j}$ is the target's orientation estimated by robot j. This algorithm drives all of the robots to track the target while avoiding collisions. As a result, the robots tend to maintain communication connectivity with others and observe the target and other robots more often than RM. As shown later, it has better performance than RM.

- *Optimization-based AJLATT (AJLATT-O)*: In this setting, the robots' linear and angular velocities are calculated by the optimization-based AJLATT algorithm in Section 3.2. We use two sets of the AJLATT-O parameters in their objective functions to make a comparison. The first one with $\alpha_{R_i} = 0$, $\alpha_{T_i} = 1$, and $\beta_i = 1$ is labeled as AJLATT-O1, and the second one with $\alpha_{R_i} = 0.2$, $\alpha_{T_i} = 1$, and $\beta_i = 1$ is labeled as AJLATT-O2. AJLATT-O1 uses only the trace of the target covariance and the potential function as its optimization objective. Meanwhile, AJLATT-O2 adopts not only the trace of the target covariance and the potential function but also the trace of the robot covariance to optimize the localization performance. In the later simulation result, we will show the benefits by adding the trace of the robot covariance. As for the parameters on robot i, we set the length of the non-zero interval as $a_i = 2$ m and $\underline{d}_i = 2$ m in (3.8) for both the robot-robot and robot-target potential functions. We also set $\bar{d}_i = 30$ m

in (3.8) for the robot-robot potential function to make the robots maintain the communication with its neighbors, and $\bar{d}_i = 20$ m for the robot-target potential function to help keep sight of the target. The planning horizons are $t_1 = 4$ and $t_2 = 11$ for the SACP process.

For the optimization-based AJLATT algorithm, the control space of each robot i is set as $\mathcal{U} = \{(v_i, \omega_i) | v_i \in [0, 0.5]$ m/s, $\omega_i \in [-\frac{\pi}{5}, \frac{\pi}{5}]$ rad/s$\}$. The same control limitation is applied to the control-based AJLATT algorithm.

We run 50 Monte Carlo simulations and use the root mean square error (RMSE) as the metric for accuracy to test the performance of these algorithms. Figure 4.1 shows each robot's average target position and orientation estimate RMSE for target tracking. Figure 4.2 shows each robot's own average position and orientation estimate RMSE for localization.



(a) Position estimate RMSE (b) Orientation estimate RMSE

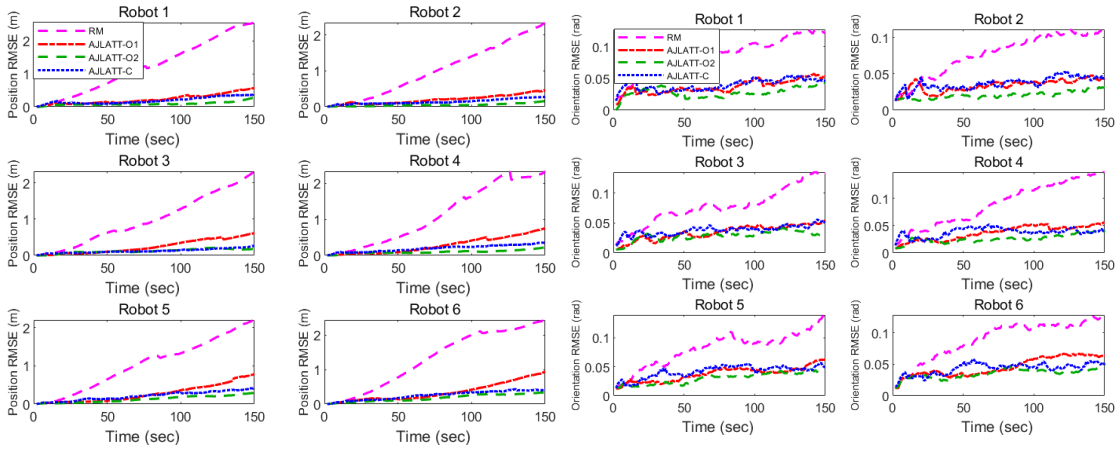
Figure 4.1: Position and orientation estimate RMSE for the target on 6 Robots (tracking).

As shown in Figures 4.1 and 4.2, the robots using AJLATT algorithms achieve

significant better performance than RM. The result confirms our expectation. The RM approach does not actively drive a one robot to observe the target and other robots or maintain the communication connectivity with its neighbors. As a result, the robot obtains fewer measurements and has fewer neighbors to exchange information, thus obtaining less accurate estimates of its own and the target's states compared with the AJLATT algorithms. Hence the RM approach has the worst performance. We can also notice that the performances of AJLATT-C and AJLATT-O are comparable, while each of these two algorithms has its own benefit. The optimization-based approach is not limited to specific models while the control-based one is computationally simple.

Also, as shown in Figure 4.1, there is no significant difference between AJLATT-O1 and AJLATT-O2 in target tracking performance. However, as we can see in a highlighted robot localization performance comparison between AJLATT-O1 and AJLATT-O2 shown in Figure 4.3, AJLATT-O2 has notable better robot localization performance than AJLATT-O1. That is because the objective function in AJLATT-O2 explicitly considers the robot localization performance. Under the same condition, by using AJLATT-O2 a robot will try to not only observe the target to obtain a more accurate target estimate and hence reduce the cost of the target covariance term in the objective function (3.7), but also observe other robots to improve self-localization performance and hence reduce the cost of the robot covariance term.

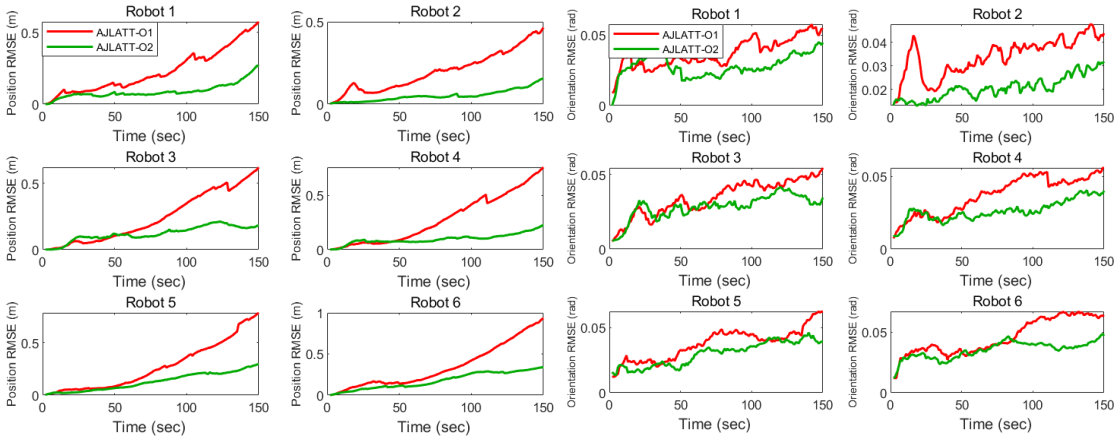
One thing worth noticing is the necessity of the potential function term in the objective function in (3.7). On the one hand, as stated in Section 3.2, the potential function term is used to avoid collisions. On the other hand, it also helps each robot to maintain



(a) Position estimate RMSE comparison (b) Orientation estimate RMSE Comparison

Figure 4.2: Position and orientation estimate RMSE for 6 Robots (localization).

a certain distance between itself and its teammates and between itself and the target. Although purely minimizing the target covariance term to some extent pushes these robots to chase the target, it will not always force all of the robots to chase the target at the same time. There are two reasons. First, a robot with a larger trace of the self-localization covariance may move its field of view away from observing the target and let its neighbors observe the target instead. A robot might have a larger trace of the self-localization covariance than its neighbors due to its movement. The previous movement of the robot such as continuously chasing the target would result in a large and consecutive non-zero v_i and would hence cause large $\mathbf{Q}_i^{k-1}$ and $\mathbf{\Phi}_i^{k-1}\hat{\mathbf{p}}_i^{k-1}(\mathbf{\Phi}_i^{k-1})^\mathsf{T}$ according to (4.2) and (4.3). From (2.6), these two terms will together induce the increase of the robot covariance during propagation. If the robot keeps sight of the target, its large trace of the self-localization covariance might make the trace of the fused target covariance $\hat{\mathbf{p}}_{T_i}^k$ calculated by (2.15a) (see **Step 3** in Table 3.2) larger than the case that it does not observe the target, which will eventually cause the increment



(a) Position estimate RMSE comparison (b) Orientation estimate RMSE comparison

Figure 4.3: Comparison of robot localization performance between AJLATT-O1 and AJLATT-O2.

of the cost of the target covariance term in the objective function (3.7). Therefore, in that case, the robot with larger self-localization covariance will move its field of view away from observing the target. Second, if we add the robot covariance term in addition to the target covariance term without using the potential function term, some robots might slow down or even stop moving to slow down the increase of the self-localization uncertainty caused by its movement. These two factors will gradually make only few robots keep sight of the target, and others might stop moving (could only spin) to observe other robots that also stop moving to improve their self-localization performance, and purely receive the target's state estimation information through the communication network. That brings benefits in the short term for the localization performance of the robots that stop moving. However, in the long term, these robots that stop moving could lose communication connectivity with the robots that keep sight of the target. As a result, these robots that stop moving might not be

able to observe the target and eventually lose their target state update. Besides, for the few robots that keep sight of the target, due to the reduction of the robot-robot measurements obtained among the entire team, they will gradually calculate inaccurate estimates of their own states at first and then the target's state, which might even eventually result in totally losing sight of the target. By adding the potential function term in the objective function (3.7), some robots might still move their fields of view away from the target, and only a few robots will keep sight of the target. However, since the robots are close enough to each other, a robot that has a smaller self-localization covariance but does not observe the target is able to quickly replace the role of the robots that are currently keeping sight of the target but having large self-localization covariances to observe the target.

The phenomenon mentioned above can be shown in Figure 4.4 in snapshots, where we use $\alpha_{R_i} = 0.2$, $\alpha_{T_i} = 1$, $\beta_i = 0$ (i.e., no potential function term in the objective function in (3.7)). As we can see, in the beginning ($k = 25$), many robots keep sight of the target. However, as time goes by, fewer and fewer robots will keep sight of the target ($k = 75$). After a certain time ($k = 125$), only the green robot is keeping sight of the target while the other robots stop moving. As shown in Figure 4.5, the lack of robot-robot measurements by the green robot (Robot 5) gradually induces a dramatic increase of the self-localization error compared with other robots that stop moving and also induces a large target estimation error as shown in Figure 4.4 at $k = 250$. Here the (overlapped) pink triangles represent the estimated target on different robots, and the red triangle represents the true target. Besides, there also comes communication disconnection between robots. For example, the green robot loses direct communication connectivity with the orange one at $k = 300$. As

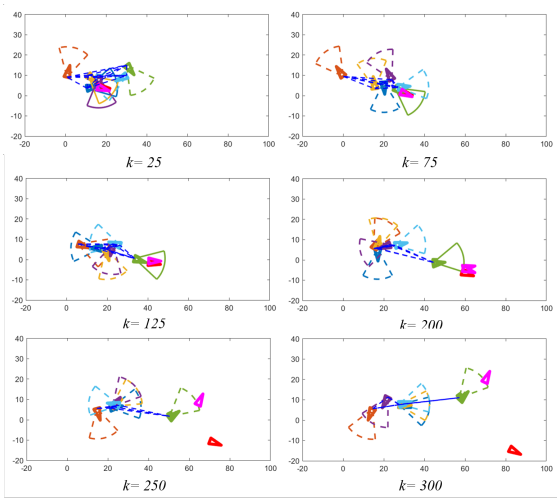


Figure 4.4: Snapshots of the robots' estimates without the potential function term in the objective function (3.7). The red triangle denotes the true target, and the (overlapped) pink triangles denote the estimated targets on different robots. Triangles and sectors in other colors represent, respectively, different robots' self estimates and fields of view. The line of a robot's sector's is solid when the target is in the robot's field of view; otherwise, it is a dash line.

we can imagine, the loss of communication connectivity will gradually make the robots that stop moving lose the latest update of the target. It is also worth noting that collisions also occur between robots. In a word, if there lacks the potential function term, the AJLATT-O is not guaranteed to work.

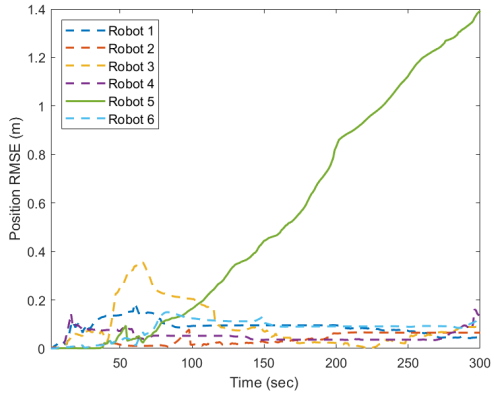


Figure 4.5: Robot position RMSE for the example illustrated by Fig. 4.4. Green solid line corresponds to the green robot (Robot 5 here), dash lines with different colors correspond to other robots in the same color in the snapshots in Fig. 4.4.

Chapter 5

Conclusions

In this thesis, we proposed two algorithms to solve the AJLATT problem in a fully distributed (communication, estimation, planning) manner to drive a team of robots to actively track a target and localize itself so as to achieve better self-localization and target tracking performance.

The first control-based algorithm explicitly incorporates the estimates of their own states and the target's state and collision avoidance in algorithm design. The other algorithm based on the optimization framework tries to find the optimal motion so that optimal robot localization and target tracking performance can be achieved while collision avoidance and communication maintenance are considered at the same time. Monte-Carlo simulations are performed to illustrate the effectiveness of our approaches. Factors that influence the performance of the optimization-based approach are discussed. The simulation result shows that both approaches work well, and their performance is comparable. Each of these two approaches has its benefits. The control-based approach is computationally simple and time

efficient, while the optimization-based approach can be applied to a wide range of realistic models.

Bibliography

[1] S. I. Roumeliotis and G. A. Bekey, "Distributed multirobot localization," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 781–795, 2002.

[2] S. S. Kia, S. F. Rounds, and S. Martinez, "A centralized-equivalent decentralized implementation of extended kalman filters for cooperative localization," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 3761–3766.

[3] T. Bailey, M. Bryson, H. Mu, J. Vial, L. McCalman, and H. Durrant-Whyte, "Decentralised cooperative localisation for heterogeneous teams of mobile robots," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2011, pp. 2859–2865.

[4] T. R. Wanasinghe, G. K. Mann, and R. G. Gosine, "Distributed leader-assistive localization method for a heterogeneous multirobotic system," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 795–809, 2015.

[5] 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*, 2007, pp. 1053–1058.

[6] 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*, 2006, pp. 2816–2821.

[7] L. Luft, T. Schubert, S. I. Roumeliotis, and W. Burgard, "Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication," *The International Journal of Robotics Research*, vol. 37, no. 10, pp. 1152–1167, 2018.

[8] L. C. Carrillo-Arce, E. D. Nerurkar, J. L. Gordillo, and S. I. Roumeliotis, "Decentralized multi-robot cooperative localization using covariance intersection," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1412–1417.

[9] A. Bahr, M. R. Walter, and J. J. Leonard, "Consistent cooperative localization," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2009, pp. 3415–3422.

[10] A. T. Kamal, J. A. Farrell, A. K. Roy-Chowdhury *et al.*, "Information weighted consensus filters and their application in distributed camera networks." *IEEE Transactions on Automatic Control*, vol. 58, no. 12, pp. 3112–3125, 2013.

[11] G. Battistelli and L. Chisci, "Kullback–leibler average, consensus on probability densities, and distributed state estimation with guaranteed stability," *Automatica*, vol. 50, no. 3, pp. 707–718, 2014.

[12] R. Olfati-Saber, "Kalman-consensus filter: Optimality, stability, and performance," in *Proceedings of the IEEE Conference on Decision and Control*, 2009, pp. 7036–7042.

[13] S. Wang and W. Ren, "On the convergence conditions of distributed dynamic state estimation using sensor networks: A unified framework," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 4, pp. 1300–1316, 2018.

[14] F. S. Cattivelli and A. H. Sayed, "Diffusion strategies for distributed kalman filtering and smoothing," *IEEE Transactions on Automatic Control*, vol. 55, no. 9, pp. 2069–2084, 2010.

[15] J. Hu, L. Xie, and C. Zhang, "Diffusion kalman filtering based on covariance intersection," *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 891–902, 2012.

[16] G. Huang, M. Kaess, and J. J. Leonard, "Consistent unscented incremental smoothing for multi-robot cooperative target tracking," *Robotics and Autonomous Systems*, vol. 69, pp. 52–67, 2015.

[17] A. Ahmad, G. D. Tipaldi, P. Lima, and W. Burgard, "Cooperative robot localization and target tracking based on least squares minimization," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2013, pp. 5696–5701.

[18] F. M. Mirzaei, A. I. Mourikis, and S. I. Roumeliotis, "On the performance of multi-robot target tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007, pp. 3482–3489.

[19] N. Atanasov, R. Tron, V. M. Preciado, and G. J. Pappas, "Joint estimation and localization in sensor networks," in *Proceedings of the IEEE Conference on Decision and Control*, 2014, pp. 6875–6882.

[20] P. Zhu and W. Ren, "Fully distributed joint localization and target tracking with mobile robot networks," *IEEE Transactions on Control Systems Technology*, pp. 1–14, 2020. [Online]. Available: https://doi.org/10.1109%2Ftcst.2020.2991126

[21] Y. Cao and W. Ren, "Distributed coordinated tracking with reduced interaction via a variable structure approach," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 33–48, 2011.

[22] T. Chung, J. Burdick, and R. Murray, "A decentralized motion coordination strategy for dynamic target tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006, pp. 2416–2422.

[23] R. Olfati-Saber and P. Jalalkamali, "Coupled distributed estimation and control for mobile sensor networks," *IEEE Transactions on Automatic Control*, vol. 57, no. 10, pp. 2609–2614, 2012.

[24] A. W. Stroupe and T. Balch, "Value-based action selection for observation with robot teams using probabilistic techniques," *Robotics and Autonomous Systems*, vol. 50, no. 2-3, pp. 85–97, 2005.

[25] K. Zhou and S. I. Roumeliotis, "Multirobot active target tracking with combinations of relative observations," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 678–695, 2011.

[26] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Information acquisition with sensing robots: Algorithms and error bounds," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6447–6454.

[27] B. Charrow, N. Michael, and V. Kumar, "Cooperative multi-robot estimation and control for radio source localization," *The International Journal of Robotics Research*, vol. 33, no. 4, pp. 569–580, 2014.

[28] B. Schlotfeldt, D. Thakur, N. Atanasov, V. Kumar, and G. J. Pappas, "Anytime planning for decentralized multirobot active information gathering," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1025–1032, 2018.

[29] K. Hausman, J. Müller, A. Hariharan, N. Ayanian, and G. S. Sukhatme, "Cooperative multi-robot control for target tracking with onboard sensing," *The International Journal of Robotics Research*, vol. 34, no. 13, pp. 1660–1677, 2015.

[30] F. Morbidi and G. L. Mariottini, "Active target tracking and cooperative localization for teams of aerial vehicles," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 5, pp. 1694–1707, 2013.

[31] F. Meyer, H. Wymeersch, M. Fröhle, and F. Hlawatsch, "Distributed estimation with information-seeking control in agent networks," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2439–2456, 2015.

[32] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the American Control Conference*, vol. 4, 1997, pp. 2369–2373.

[33] S. Julier and J. K. Uhlmann, "General decentralized data fusion with covariance intersection," in *Handbook of multisensor data fusion*. CRC Press, 2017, pp. 339–364.

[34] W. Niehsen, "Information fusion based on fast covariance intersection filtering," in *Proceedings of the IEEE International Conference on Information Fusion*, vol. 2, 2002, pp. 901–904.