Signal and Systems
Mekelweg 4,
2628 CD Delft
The Netherlands
https://sps.ewi.tudelft.nl/

SPS-2025-08

M.Sc. Thesis

Distributed Gaussian Process with Multi-Agents Localization and Tracking

Yi Dai B.Sc.
6015093

Abstract

Accurate cooperative-localization of stationary agents and tracking of mobile targets are critical for multi-agent autonomy, particularly in global navigation satellite system (GNSS)-denied environments such as maritime search-and-rescue (SAR) missions. In such settings, agents often lack reliable global positioning and complete target observability, challenging distributed perception and coordination. State-of-the-art approaches such as joint Kalman filtering was widely applied.

To address this, we propose two approaches: a sequential optimization strategy and a unified integrated optimization framework. The sequential method decouples localization and tracking—first estimating agent positions via inter-agent ranging and then performing distributed Gaussian Process (GP) tracking using alternating direction method of multipliers (ADMM) -based fusion. Although efficient and modular, this approach may suffer from error propagation. To mitigate this, we introduce integrated optimization framework that couples both tasks via a weighted multi-objective cost. A convex relaxation of the localization subproblem yields closed-form updates, and the full problem is solved in a distributed manner using ADMM.

Simulations based on GNSS-denied maritime scenarios show that both methods enhance tracking and localization performance, with the integrated framework offering superior speed. These results underscore the value of cooperative self-localization and distributed tracking in complex multi-agent environments.

Distributed Gaussian Process with Multi- Agents Localization and Tracking

Thesis

submitted in partial fulfillment of the
requirements for the degree of

Master of Science

in

Electrical Engineering

by

Yi Dai B.Sc.
from Shanghai, China

This work was performed in:

Signal and Systems Group
Department of Microelectronics
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Delft University of Technology

Delft University of Technology
Department of
Microelectronics

The undersigned hereby certify that they have read and recommend to the Faculty of Electrical Engineering, Mathematics and Computer Science for acceptance a thesis entitled **"Distributed Gaussian Process with Multi- Agents Localization and Tracking"** by **Yi Dai B.Sc.** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: 21/08/2025

Chairman: _____

Dr. Raj Thilak Rajan

Advisor: _____

Ir. Ellen Riemens

Committee Members: _____

Dr. Francesco Fioranelli

_____

Abstract

Accurate cooperative-localization of stationary agents and tracking of mobile targets are critical for multi-agent autonomy, particularly in global navigation satellite system (GNSS)-denied environments such as maritime search-and-rescue (SAR) missions. In such settings, agents often lack reliable global positioning and complete target observability, challenging distributed perception and coordination. State-of-the-art approaches such as joint Kalman filtering was widely applied.

To address this, we propose two approaches: a sequential optimization strategy and a unified integrated optimization framework. The sequential method decouples localization and tracking—first estimating agent positions via inter-agent ranging and then performing distributed Gaussian Process (GP) tracking using alternating direction method of multipliers (ADMM) -based fusion. Although efficient and modular, this approach may suffer from error propagation. To mitigate this, we introduce integrated optimization framework that couples both tasks via a weighted multi-objective cost. A convex relaxation of the localization subproblem yields closed-form updates, and the full problem is solved in a distributed manner using ADMM.

Simulations based on GNSS-denied maritime scenarios show that both methods enhance tracking and localization performance, with the integrated framework offering superior speed. These results underscore the value of cooperative self-localization and distributed tracking in complex multi-agent environments.

Acknowledgments

First and foremost, I thank my supervisor, Professor Raj Rajan, for his meticulous guidance, profound insights, and unfailing support throughout my research. Whenever I faced difficulties, he offered encouragement and clear direction that made this thesis possible. His rigorous scholarship and illuminating discussions broadened my horizons and kindled my passion for research.

I am also deeply grateful to my daily supervisor, Ellen Riemens, for her patient guidance and steady support during my thesis. She provided detailed advice and walked me through each experiment. Without her help, this thesis would not have been possible. She also shared valuable perspectives on academic life and offered personal advice, listening to my concerns with great kindness.

I owe heartfelt thanks to my parents and family. I am deeply grateful to Ms. Xiuli Wang and Mr. Fei Dai for their unwavering support, which sustained me both materially and emotionally. I sincerely appreciate their profound, enduring, and selfless love.

I would like to thank all my dear friends from TU Delft, SPS, KKDH, and everyone who has accompanied me along the way. Thank you for your help—your companionship has made my two years in the Netherlands rich and colorful, and your encouragement has enabled me to complete this journey.

Finally, I want to thank myself—the version of me who, two years ago, bravely decided to come to the Netherlands. I made it.

Yi Dai B.Sc.
Delft, The Netherlands
21/08/2025

"But were things different: had I not a friend left in the world; were there not a single house open to me in pity; had I to accept the wallet and ragged cloak of sheer penury: as long as I am free from all resentment, hardness and scorn, I would be able to face the life with much more calm and confidence than I would were my body in purple and fine linen, and the soul within me sick with hate."

—Oscar Wilde, *De Profundis*

Contents

List of Figures

List of Tables

1 Introduction

In multi-agent systems, the ability of agents to perceive their surrounding environment is essential for enabling autonomous behavior and collaborative tasks. As such systems are increasingly deployed in complex and dynamic scenarios—such as drone swarms, vehicular networks, and distributed sensor systems—there is a growing demand for perception frameworks that are accurate, robust, and capable of being implemented in a fully distributed manner. Against this backdrop, performing reliable *localization* and *target tracking* under distributed settings has become a key research topic in the field of multi-agent systems.

Localization focuses on enabling each agent to estimate its own position, which is a prerequisite for reliable operation in unknown or partially observable environments. Target tracking, on the other hand, involves the continuous estimation of external dynamic targets and forms the basis for monitoring, identification, pursuit, and response tasks. Although these two tasks are distinct in both objective and algorithmic design, they are often deployed side by side in the overall architecture of multi-agent systems, operating as complementary modules within a unified perception framework.

Motivated by this need, this work proposes an *integrated optimization framework* that structurally combines distributed target tracking with cooperative localization. While the two modules are designed to function independently—i.e., the localization process does not rely on feedback from the tracking results—the proposed framework enables shared information and functional coordination at the system level. This allows for more efficient and scalable autonomous perception in complex and uncertain environments.

1.1 Background

Localization and tracking are two foundational components of intelligent decision-making in multi-agent systems. In such systems, multiple autonomous agents—such as unmanned aerial vehicles (UAVs), autonomous ground robots, or marine buoys—cooperate to sense their environment, coordinate their motion, and perform tasks such as surveillance, environmental monitoring, or search-and-rescue operations. Accurate knowledge of each agent's own position and the states of dynamic targets in the environment is essential for effective collaboration and situational awareness.

Cooperative-localization enables each agent to determine its precise position within a global or shared reference frame, serving as a fundamental prerequisite for collaborative sensing, path planning, and task allocation. Especially when some agents have access to accurate position information, cooperative localization can become highly precise. It typically relies on sensor data—such as GPS, IMU, or relative distance measurements—and employs estimation algorithms to reconstruct the true position as

accurately as possible under locally limited information [1, 2].

Target tracking, on the other hand, involves continuously estimating the states of one or more dynamic objects in the environment—such as humans, vehicles, or UAVs—including their positions, velocities, and even motion trajectories. This task becomes particularly challenging when targets are non-communicative or exhibit unknown behaviors [3].

Although cooperative-localization and target tracking can be modeled and solved independently in theory and algorithm design, in practical distributed multi-agent systems, they are often interdependent and tightly coupled. In resource-constrained environments—such as those with limited communication bandwidth, restricted sensing range, and heterogeneous computational capabilities—each agent must rely on incomplete information from itself and its neighbors, share observations, and cooperate to jointly estimate both its own state and the states of external targets [4, 5].

The tasks of localization and tracking are central to a wide range of real-world applications. In disaster response scenarios, UAVs are deployed to locate victims or drifting objects (e.g., life rafts) in GNSS-denied environments. In environmental monitoring, autonomous agents must associate their observations—such as temperature, salinity, or gas concentration—with precise spatial coordinates to reconstruct field distributions [6]. Other use cases include mobile asset tracking in logistics, cooperative mapping in unknown environments, and surveillance of sensitive infrastructure or borders. In all these settings, location awareness and dynamic tracking are critical to mission success.

1.2 Application Case

Cooperative-localization and target tracking serve as the backbone of numerous real-world applications that demand autonomous perception, coordination, and decision-making in dynamic and uncertain environments. In disaster response missions, for instance, unmanned aerial vehicles (UAVs) and ground robots rely on precise localization to navigate collapsed structures and collaboratively search for survivors or hazardous objects in GPS-denied environments [7]. Simultaneous tracking of mobile entities—such as victims, rescuers, or moving debris—is essential for adaptive task planning and risk mitigation.

In the context of environmental monitoring, autonomous marine or aerial agents equipped with chemical, thermal, or acoustic sensors must associate their observations with accurate spatial positions to reconstruct dynamic environmental fields—such as pollution plumes, temperature gradients, or algal blooms [8]. Here, real-time tracking of drifting targets like oil spills or marine life enhances the system's ability to perform persistent coverage and long-term prediction.

Smart transportation and logistics also benefit significantly from joint localization and tracking. Multi-robot systems deployed in warehouses or urban areas require robust cooperative-localization for safe navigation and must simultaneously track assets—such as delivery vehicles, parcels, or inventory—to ensure traceability and efficient coordination [9]. In border surveillance and infrastructure security, heterogeneous agents collaboratively track intruding objects while maintaining situational awareness through continuous self-localization updates [10].

In all these applications, the interplay between localization and tracking is critical: localization errors directly impact tracking fidelity, and sustained observations of known targets can, in turn, refine agent positions. Therefore, integrating both functionalities into a unified framework not only improves system robustness and autonomy, but also enables intelligent behaviors such as target-aware navigation, adaptive communication, and resilient coverage.

1.3 Challenges

Despite the ubiquity of these tasks, accurate and scalable localization and tracking remain challenging. Several key issues include:

- **GNSS limitations:** Agents may operate in environments where GPS signals are unavailable, such as underwater, indoors, or under dense foliage.
- **Limited sensing and communication:** Each agent can typically sense only a local subset of the environment and can communicate only with nearby agents within a limited range.
- **Dynamic targets and partial observations:** The target may be moving unpredictably and is often observed only indirectly or intermittently by a subset of agents.
- **Decentralization constraints:** Centralized solutions are often infeasible due to communication bottlenecks, latency, or robustness concerns.

These challenges require distributed algorithms that are robust to noise, scalable to large networks, and capable of operating under partial information.

1.4 State-of-the-Art Solutions

Recent advances have increasingly focused on cooperative localization and decentralized target tracking as two complementary tasks in multi-agent systems. In cooperative localization, each agent infers its own position by leveraging inter-agent measurements—such as relative distances, bearings, or time-of-flight—and fuses this information through distributed optimization or probabilistic filtering. Techniques like consensus-based EKF, distributed invariant Kalman filters defined on Lie groups, and information consensus filters have demonstrated improved consistency and accuracy in 3-D multi-robot settings [11, 12, 13].

For decentralized tracking, filtering methods—such as decentralized Kalman filters, particle filters, and Gaussian Process (GP) regression—enable each agent to locally estimate the dynamic target's state while exchanging information with neighbors. For instance, Rao–Blackwellized particle filters and Gaussian Mixture approaches have been successfully applied to multi-target tracking under partial observability and unknown target numbers [14, 15].

Furthermore, there is growing interest in joint localization and tracking(JLT), where both problems are solved within a unified estimation framework. This integration is driven by the mutual dependency: localization error degrades target tracking, while persistent observations of the target can help refine agent position estimates. Advanced joint estimators—such as graph-based SLAM variants and Rao–Blackwellized particle smoothing—have been extended to handle dual-state estimation, simultaneously tracking agents and targets in distributed scenarios [16, 12].

These JLT frameworks are particularly beneficial in GPS-denied or cluttered environments, as they enable uncertainty coupling and exploit interdependence between agent state and target state for enhanced situational awareness and collaborative performance.

1.5 Motivation and Contribution

This thesis is motivated by a real-world challenge in maritime search and rescue: tracking a drifting life raft entrained in a mesoscale anticyclonic eddy, where GNSS signals are often unavailable or unreliable. In such scenarios, a team of UAVs must both localize themselves—without relying on GPS—and track the motion of the raft using only noisy, relative observations and limited inter-agent communication. The inherent coupling between agent localization accuracy and target tracking performance raises the need for a unified, robust estimation framework.

However, most existing approaches treat cooperative localization and target tracking as separate problems. The mainstream solution—Joint Kalman Filtering—fails to deliver reliable estimates in highly nonlinear systems due to its reliance on linearization and global observability. To address this, we explore alternative strategies that integrate both tasks into a distributed estimation process suitable for resource-constrained environments.

The main contributions of this thesis are summarized as follows:

- We identify the failure modes of the Joint Kalman Filter in nonlinear target-agent systems through theoretical analysis and simulation.
- We develop a sequential optimization method that decouples localization and tracking into two stages, enabling stable estimation when localization is accurate or the target moves slowly.
- We propose an integrated optimization framework that simultaneously estimates agent positions and target states by coupling the two objectives in a single optimization problem. This formulation remains robust in nonlinear systems.
- We design a distributed tracking module based on Gaussian Process regression, allowing each agent to learn and predict the target's trajectory from partial and noisy local observations.
- We compare the sequential and integrated approaches in various settings, and provide insights into their respective advantages, limitations, and applicability across dynamic and uncertain environments.

1.6 Research Questions

This work addresses the following research questions, aimed at building a robust, decentralized estimation framework that integrates localization and tracking under nonlinear dynamics and limited sensing:

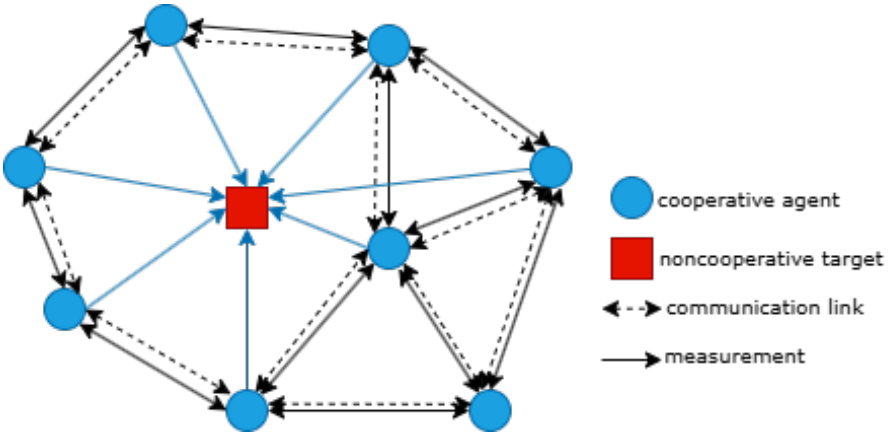


Figure 1.1: Overview of the proposed integrated localization and tracking framework in a decentralized multi-agent UAV system.

- **RQ1:** How can agents be cooperatively localized by using only local, noisy range measurements and partial knowledge of neighbor positions, when only few agents known their position, without relying on any global reference?
- **RQ2:** How can the trajectory of a non-cooperative and dynamically evolving target be accurately predicted in a distributed manner using Gaussian Process models and noisy local observations?
- **RQ3:** Given the coupling between agent localization and target tracking, how can both tasks be integrated into a unified estimation framework, and how does this integrated approach compare to a sequential solution in terms of robustness, accuracy, and applicability?

1.7 Outline

This thesis is organized as follows:

- **Chapter 1 – Introduction**: Introduces the background, motivating applications core challenges, related work, research questions, and the key contributions of this thesis.
- **Chapter 2 – Problem Formulation**: Defines the three estimation problems addressed in this work, including (i) cooperative-localization with few known-position agents, (ii) distributed tracking of a non-cooperative target, and (iii) their integration as a coupled estimation problem.

- **Chapter 3 – Methodology**: Presents the modeling and algorithmic foundations. It introduces Gaussian Process regression for target dynamics learning, a neighbor-based localization method using distance measurements, and validates each module individually through simulations.
- **Chapter 4 – Distributed Gaussian Process with Multi-agent Localization and Tracking**: Develops the full ICLDT framework. This chapter compares a Joint-KF baseline with two proposed solutions: a sequential optimization strategy and a fully integrated ADMM-based method. It also analyzes their performance under nonlinear dynamics.
- **Chapter 5 – Conclusion**: Summarizes the major findings, highlights the advantages of the integrated approach, and discusses possible future directions in decentralized integrated perception and estimation.

Problem formulation

2

In this section, we are going to elaborate on the details of the research topics in this thesis work. First, we define some functions as preliminary knowledge.

2.1 Problem 1: Cooperative-localization with few known-position agents

We consider a static network of N agents and a set of agents with known positions. Each agent estimates its own position based on noisy distance measurements to its immediate neighbors and any available known-position agent. The whole network is defined as $\mathcal{F}$, let $\mathcal{A}$ denote the set of agent nodes with known positions $\mathbf{p}_a, a \in \mathcal{A}$, and let $\mathbf{p}_i$ denote the position of the i-th agent.

Measurement Model Definition

For any pair of neighboring agents $(k, j) \in \mathcal{N}_i$, the measured distance d_{kj}^t at time t represents the relative distance between agent k and agent j. Ideally, the measurement satisfies:

$$d_{kj}^t \approx \|\mathbf{p}_k^t - \mathbf{p}_j^t\|$$

Additive Gaussian Noise Model

To account for measurement uncertainty, we assume a Gaussian noise model for the pairwise distance measurements:

$$d_{kj}^t = \|\mathbf{p}_k^t - \mathbf{p}_j^t\| + \varepsilon_{kj}, \quad \varepsilon_{kj} \sim \mathcal{N}(0, \sigma_n^2)$$

Construction of the Localization Cost Function

Based on the squared error between the measured and estimated distances, the localization cost function is formulated as:

$$J_1(\mathbf{p}) = \sum_{(k,j)\in\mathcal{N}_s} \ln p(\mathbf{p}_k^t, \mathbf{p}_j^t \mid d_{kj}^t) \tag{2.1}$$

Only non-anchor agents update their positions iteratively via a distributed optimization algorithm, exchanging current estimates with neighbors. Anchors provide fixed reference points and do not participate in the update process, thereby establishing a global coordinate frame. The optimization problem is convex and hence can be readily solved.

This formulation allows all agents to localize themselves consistently in a common global frame while maintaining fully decentralized operation.

2.2 Problem 2: Distributed Target Tracking

Each agent $i \in \{1, \ldots, N\}$ obtains a relative measurement of the target position with respect to its own position. Specifically, the measurement model is given by:

$$\mathbf{z}_i^t = \mathbf{H}(\mathbf{x}_i^t - \mathbf{p}_i^t) + \mathbf{v}_i, \quad \mathbf{v}_i \sim \mathcal{N}(0, \Sigma)$$

where $\mathbf{x}_t \in \mathbb{R}^d$ is the unknown target position at time t, $\mathbf{p}_i \in \mathbb{R}^d$ is the known position of agent i, $\mathbf{H}$is the measurement function and $\mathbf{z}_i \in \mathbb{R}^d$ is the relative measurement subject to additive Gaussian noise.

Each agent constructs a local loss function $l^{(i)}(\boldsymbol{\theta_i})$ based on its measurement, where θ_i denotes the local estimate of the target position maintained by agent i. The distributed tracking cost function is then formulated as:

$$J_2 = \sum_{i=1}^{N} l^{(i)}(\boldsymbol{\theta}_i) \tag{2.2}$$

$$l^{(i)}(\boldsymbol{\theta}) = (\mathbf{z}_i + \mathbf{p}_i)^\top \boldsymbol{\Sigma}(\theta)^{-\mathbf{1}} (\mathbf{z}_i + \mathbf{p}_i) + log|\boldsymbol{\Sigma}(\theta)|$$

To enforce consensus among agents and estimate a single consistent target position, one can impose the constraint $\boldsymbol{\theta_i} = \boldsymbol{\theta_j}$ for all $(i, j) \in \mathcal{F}$, and solve the resulting optimization problem in a distributed manner using ADMM or consensus-based methods.

2.3 Problem 3: Cooperative localization and distributed target tracking (CLDT)

In real-world decentralized tracking systems, cooperative localization (CL) and distributed target tracking (DT) are inherently coupled. The accuracy of target tracking relies heavily on the precision of agent localization, while the observation of moving targets may in turn assist agents in improving their own localization. This mutual dependency motivates the design of a joint estimation framework.

We define two coupled cost functions: J_1 for localization and J_2 for target tracking. Specifically, the joint objective is formulated as a weighted combination:

$$J(\{\mathbf{p}, \boldsymbol{\theta}\}) = \alpha J_1(\{\mathbf{p}\}) + (1 - \alpha) J_2(\{\boldsymbol{\theta}, \mathbf{p}\}), \tag{2.3}$$

where $\alpha \in [0, 1]$ is a trade-off parameter balancing the importance of localization and tracking.

However, this joint optimization problem is generally non-convex and tightly coupled, as the tracking loss J_2 depends on the current estimates of the agent positions from J_1. To solve this problem, we explore two strategies:

- **Sequential Optimization:** First optimize J_1 to update agent positions, then use these positions to compute and optimize J_2.
- **Integrated Optimization:** Reformulate the joint problem using consensus constraints and solve it using distributed ADMM to enable simultaneous updates of $\mathbf{p}_i$ and $\boldsymbol{\theta_i}$ across agents.

This CLDT formulation provides a unified and scalable approach for decentralized estimation in GNSS-denied environments, such as cooperative UAV tracking in maritime search and rescue scenarios.

3 Existing Methods

In this chapter, we present the methodological framework proposed to address two fundamental challenges in multi-agent systems: *target tracking* and *cooperative localization*. The chapter is structured into two main parts, each focusing on one of these closely related research problems.

First, Section 3.1 focuses on the problem of *distributed target tracking*. We begin by introducing the basic principles of target tracking and reviewing several representative approaches, highlighting their strengths and limitations. The potential of Gaussian processes for modeling target dynamics is then discussed. Building on this foundation, we explore a distributed tracking strategy for multi-agent systems, including target motion modeling, learning-based prediction using distributed Gaussian processes, and a set of simulation studies. We further compare different deployment strategies to evaluate the robustness and performance of the proposed approach in various scenarios.

Next, Section 3.2 addresses the problem of *cooperative localization*. The section starts with a discussion of the motivation and key challenges, followed by the introduction of a distributed localization method based on inter-agent distance measurements. This method enables agents to iteratively refine their position estimates through local observations and cooperative interactions. To assess the applicability of the approach, we also conduct a comprehensive analysis of its performance under different network topologies and noise conditions, supported by simulation results that demonstrate its robustness and generalizability.

3.1 Distributed Tracking

Distributed target tracking is a fundamental problem in cooperative multi-agent systems, where a group of agents collaboratively estimates the dynamic state of a moving target using partial and noisy observations. Unlike centralized methods that rely on a fusion center, distributed approaches operate under limited communication and sensing constraints, making them more scalable and robust to failures.

This section first provides a brief overview of conventional target tracking methods and discusses their applicability in distributed scenarios. Particular attention is given to Gaussian process (GP) models due to their flexibility in capturing nonlinear and uncertain target dynamics. We then introduce a learning-based distributed tracking framework built upon GP regression, where each agent learns local models and exchanges information with neighbors to improve prediction accuracy. Simulation results are presented to evaluate the effectiveness of the proposed method, followed by a comparative analysis of various agent deployment strategies in terms of tracking accuracy and robustness.

3.1.1 Target Tracking

Target tracking in dynamic and uncertain environments remains a fundamental challenge in robotics, autonomous systems, and sensor networks. Many real-world targets exhibit *nonlinear motion patterns*, are subject to *intermittent observations*, and experience *noisy and uncertain measurements*. These challenges demand estimation algorithms that are robust, adaptive, and capable of handling nonlinear and stochastic dynamics.

The Extended Kalman Filter (EKF) is one of the earliest and most widely used approaches for nonlinear state estimation [17]. By linearizing the system dynamics and observation models around the current estimate using a first-order Taylor expansion, the EKF provides a computationally efficient recursive filtering framework. However, its performance may degrade significantly in the presence of strong nonlinearities or non-Gaussian noise, due to its reliance on local approximations.

To address these limitations, the Unscented Kalman Filter (UKF) was proposed as a derivative-free alternative [17, 18]. The UKF uses a deterministic sampling technique based on sigma points to better capture the posterior distribution after a nonlinear transformation. Compared to EKF, the UKF typically achieves higher accuracy in strongly nonlinear scenarios, albeit at a higher computational cost.

More recently, data-driven methods such as Gaussian Processes (GPs) have emerged as flexible and powerful alternatives for learning system dynamics directly from data [19]. GP-based methods do not require an explicit motion model; instead, they learn the underlying dynamics from trajectory data and provide uncertainty-aware predictions. In distributed multi-agent settings, each agent can independently train a local GP model based on its own observations, and share information with neighbors to improve global estimation accuracy.

To combine the advantages of data-driven learning and recursive estimation, hybrid methods such as GP-EKF and GP-UKF have been proposed [20, 21]. These approaches integrate GP-learned dynamics into filtering frameworks, enabling accurate and data-efficient tracking even under noisy and partially observable conditions.

3.1.1.1 Extended Kalman Filter (EKF)

The Extended Kalman Filter (EKF) approximates nonlinear system dynamics using first-order Taylor expansion. It assumes the system follows a nonlinear discrete-time state-space model:

$$\begin{aligned}\mathbf{x}_k &= f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) + \mathbf{w}_{k-1},\\ \mathbf{z}_k &= h(\mathbf{x}_k) + \mathbf{v}_k,\end{aligned}$$

where $\mathbf{x}_k$ is the state, $\mathbf{u}_k$ is the control input, $\mathbf{z}_k$ is the observation, $f(\cdot)$ is the dynamic model and $h(\cdot)$ is the measurement model, and $\mathbf{w}_k, \mathbf{v}_k$ are zero-mean Gaussian noises with covariances $\mathbf{Q}_k, \mathbf{R}_k$, respectively.

Prediction step:

$$\begin{aligned}\hat{\mathbf{x}}_{k|k-1} &= f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}),\\ \mathbf{P}_{k|k-1} &= \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^{\top} + \mathbf{Q}_{k-1},\end{aligned}$$

where $\mathbf{F}_{k-1} = \frac{\partial f}{\partial \mathbf{x}}\big|_{\hat{\mathbf{x}}_{k-1|k-1}}$ is the Jacobian of the dynamic model.

Update step:

$$\begin{aligned}\mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{H}_k^\top \left(\mathbf{H}_k \mathbf{P}_{k|k-1}\mathbf{H}_k^\top + \mathbf{R}_k\right)^{-1}, \\ \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \left(\mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1})\right), \\ \mathbf{P}_{k|k} &= \left(\mathbf{I} - \mathbf{K}_k \mathbf{H}_k\right) \mathbf{P}_{k|k-1},\end{aligned}$$

with $\mathbf{H}_k = \frac{\partial h}{\partial \mathbf{x}}\big|_{\hat{\mathbf{x}}_{k|k-1}}$ as the Jacobian of the measurement model.

3.1.1.2 Unscented Kalman Filter (UKF)

The Unscented Kalman Filter (UKF) addresses the limitations of EKF by avoiding linearization. It uses a set of deterministic sample points (sigma points) to propagate uncertainty through nonlinear functions.

Given a prior state estimate $\hat{\mathbf{x}}_{k-1} \in \mathbb{R}^L$ and its covariance $\mathbf{P}_{k-1}$, sigma points are constructed as:

$$\mathcal{X}_{k-1}^{(i)} = \hat{\mathbf{x}}_{k-1} \pm \left(\sqrt{(L+\lambda)\mathbf{P}_{k-1}}\right)_i, \quad i = 0, \ldots, 2L,$$

where $\lambda = \alpha^2(L+\kappa) - L$, and α, κ are scaling parameters. L denotes the dimension of the state vector, determining the number of sigma points $(2L+1)$, and κ is a scaling parameter that controls the spread of the sigma points around the mean.

Each sigma point is propagated:

$$\mathcal{X}_{k|k-1}^{(i)} = f(\mathcal{X}_{k-1}^{(i)}, \mathbf{u}_{k-1})$$

Then, the predicted mean and covariance are:

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2L} W_i^{(m)} \mathcal{X}_{k|k-1}^{(i)},$$

$$\mathbf{P}_{k|k-1} = \sum_{i=0}^{2L} W_i^{(c)} \left(\mathcal{X}_{k|k-1}^{(i)} - \hat{\mathbf{x}}_{k|k-1}\right)\left(\mathcal{X}_{k|k-1}^{(i)} - \hat{\mathbf{x}}_{k|k-1}\right)^\top + \mathbf{Q}_k.$$

The same procedure is applied for the measurement update using transformed sigma points.

3.1.1.3 Gaussian Process Regression (GP)

Gaussian Process (GP) regression is a non-parametric Bayesian approach for learning dynamic models directly from data. A GP defines a distribution over functions:

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')),$$

where $m(\cdot)$ is the mean function (often set to zero), and $k(\cdot, \cdot)$ is a positive-definite kernel function. The kernel function $k(\cdot, \cdot)$ is a key component of Gaussian processes,

used to characterize the similarity between inputs. There is a wide variety of kernel choices, such as linear kernels, polynomial kernels, Matérn kernels, and the commonly used Radial Basis Function (RBF) kernel. Different kernels encode different prior assumptions and smoothness properties of the function.

In our experiments, we adopt the RBF kernel due to its smoothness and good generalization capability, which is expressed as

$$k_{\text{RBF}}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\ell^2}\right),$$

where σ_f^2 controls the output variance and ℓ is the length-scale hyperparameter that determines the smoothness of the function.

Given training data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$, the predictive distribution for a test input $\mathbf{x}_*$ is Gaussian with:

$$\mu(\mathbf{x}_*) = \mathbf{k}_*^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{y},$$
$$\sigma^2(\mathbf{x}_*) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{k}_*,$$

where $\mathbf{K}$ is the Gram matrix with entries $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, and $\mathbf{k}_*$ is the covariance vector between $\mathbf{x}_*$ and training points.

GPs are especially valuable when the system dynamics are partially unknown or affected by unmodeled disturbances, which is common in oceanic environments.

Kernel Interpolation for Scalable Structured GP (KISS-GP) Despite the flexibility of standard Gaussian Processes, their computational cost scales cubically with the number of training points n, due to the inversion of the $n \times n$ kernel matrix. This makes GP impractical for large-scale datasets or online learning scenarios typical in multi-agent tracking systems.

KISS-GP addresses this issue by introducing a scalable approximation framework that exploits structured kernel interpolation and sparse inducing points placed on a regular grid [22]. It approximates the full kernel matrix $\mathbf{K}$ as:

$$\mathbf{K} \approx \mathbf{W}\mathbf{K}_u\mathbf{W}^\top,$$

where: - $\mathbf{K}_u$ is a structured (e.g., Kronecker or Toeplitz) kernel matrix computed over grid-based inducing points, - $\mathbf{W}$ is a sparse interpolation weight matrix that maps training inputs to nearby grid points.

This decomposition enables:$\mathcal{O}(n)$ memory complexity, $\mathcal{O}(n)$ matrix-vector multiplication for inference, Scalability to datasets with millions of points.

In the context of distributed target tracking, KISS-GP allows each UAV to efficiently learn and update local GP models from high-rate observations, while maintaining predictive accuracy comparable to full GP. It is especially suitable for embedded platforms with limited memory and compute resources.

3.1.1.4 Motivation

To better select the most suitable algorithm for the search and rescue scenario, we first evaluate target tracking under the ideal condition where each agent has access to all data points. To evaluate the performance of different tracking algorithms under realistic nonlinear dynamics, we simulate the motion of a drifting life raft whose behavior is approximated by a weakly nonlinear unicycle model. The target state at time step k is represented by

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k, \qquad \mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}) \tag{3.1}$$

$\mathbf{x}_k = [x_k, y_k, \theta_k]^\top$, which corresponds to its 2D position and heading angle. The state evolves according to the following discrete-time nonlinear dynamics:

$$x_{k+1} = x_k + v\cos(\theta_k) \cdot \Delta t + w_x \tag{3.2}$$
$$y_{k+1} = y_k + v\sin(\theta_k) \cdot \Delta t + w_y \tag{3.3}$$
$$\theta_{k+1} = \theta_k + \omega \cdot \Delta t + w_\theta \tag{3.4}$$

The control input is fixed as $\mathbf{u}_k = [v_k, \omega_k]^\top = [1.0, 0.1]^\top$, representing constant linear and angular velocities. The process noise $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q})$ models environmental disturbances, with covariance matrix $\mathbf{Q} = \mathrm{diag}(0.01, 0.01, 0.0005)$. The simulation starts from the initial state $[x_0, y_0, \theta_0]^\top = [0, 0, \pi/4]^\top$, and runs for $T = 100$ seconds with a time step of $\Delta t = 0.1$ seconds, resulting in 1000 discrete steps.

In this experiment, we assume that all local measurements from agents are centrally collected and used, i.e., the system has full global access to the target's observations at each time step.

We compare the performance of several tracking algorithms, including standard Gaussian Process (GP), KISS-GP with varying numbers of inducing points, and classical filtering methods such as the Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF). The comparison results in Figure 3.1 and Table 3.1 show the mean squared errors (MSE) in X, Y, and θ, as well as the computation time of each method. The results are summarized as follows.

Table 3.1: Comparison of Tracking Accuracy and Runtime for Different Estimation Methods

Method	**MSE$_X$**	**MSE$_Y$**	**MSE$_\theta$**	**Time (s)**
GP	1.1e-4	4.1e-5	7.8e-6	565.79
KISSGP-10	0.47	1.2	6.5e-4	**82.43**
KISSGP-25	1.9e-3	1.8e-3	6.5e-4	**82.41**
KISSGP-50	4.4e-4	3.7e-4	3.6e-4	**82.68**
KISSGP-100	1.3e-4	2.0e-4	4.6e-5	**80.35**
KISSGP-200	1.2e-4	2.0e-4	4.4e-5	**82.78**
KISSGP-500	1.1e-4	1.8e-4	4.0e-5	**82.13**
EKF	1.1e-3	9.9e-3	1.3e-3	486.89
UKF	1.2e-3	1.0e-2	1.2e-3	458.68

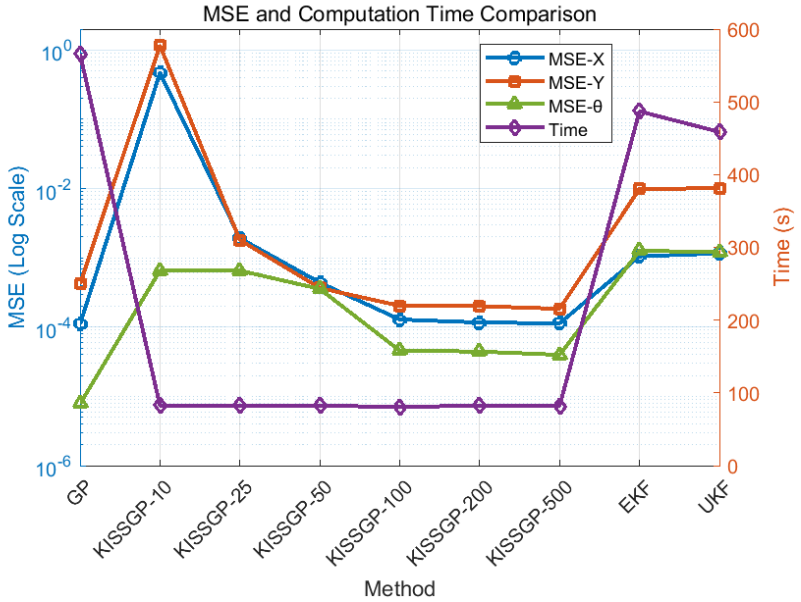


Figure 3.1: Comparison of MSE and computation time across different tracking methods. The MSE values are shown on a log scale (left axis), and computation time is plotted on the right axis.

As shown in Table 3.1 and Figure 3.1, GP method exhibits superior performance in target tracking tasks. It consistently achieves the lowest mean squared errors across all state dimensions (x, y, and θ), highlighting its strong capability in modeling nonlinear dynamics. This advantage is particularly critical in maritime search and rescue (SAR) scenarios, where drifting targets such as life rafts are often influenced by mesoscale anticyclonic eddies, resulting in quasi-periodic, nonlinear, and uncertain trajectories. In such cases, conventional filtering methods like EKF and UKF exhibit significantly larger errors, making it difficult to maintain accurate tracking performance.

Furthermore, the results indicate that GP-based methods not only provide consistently low error levels but also demonstrate strong scalability through their structured variants, such as KISS-GP. As the number of inducing points increases, the estimation accuracy of KISS-GP approaches that of the full GP, validating its potential for use in large-scale distributed systems. This scalability is especially important in multi-agent tracking frameworks where each agent has limited observation and communication capacity.

In summary, this study adopts the Gaussian Process (GP) framework as the core methodology due to its high accuracy, robustness to environmental uncertainties, and good scalability in complex tracking tasks. One major advantage of GPs is their ability to flexibly learn system dynamics directly from data without relying on predefined motion equations, which enables the model to handle arbitrarily complex target behaviors based solely on observational data. This property makes GP models particularly well suited for ocean environments where drift patterns are partially known and continuously evolving. With the integration of structured kernel methods, GP-based approaches show great potential for improving real-time coordination and estimation

accuracy in multi-UAV search-and-rescue (SAR) systems.

3.1.2 Distributed Gaussian Process

Despite the strong accuracy and modeling advantages of Gaussian Processes, directly applying standard GP methods in multi-agent systems presents several challenges. In practical maritime search and rescue missions, unmanned aerial vehicles (UAVs) are spatially distributed and constrained by factors such as limited communication range, bandwidth, and onboard energy resources. These limitations prevent the centralized collection and processing of all sensor observations in real time. Furthermore, the dynamic evolution of target states and environmental uncertainties demand that information be collaboratively fused across agents in a distributed manner to ensure both tracking performance and system robustness.

To address these constraints without compromising estimation accuracy, this research introduces the Distributed Gaussian Process (DGP) framework. DGP enables each UAV to construct local surrogate models based on partial observations and to perform cooperative modeling and inference through limited inter-agent communication. This decentralized learning approach preserves the expressive power of GPs while ensuring scalability and resilience, making it a promising solution for complex multi-agent tracking tasks in ocean environments.

3.1.2.1 Traditional Distributed Gaussian Process

An alternative to sparse approximations is distributed Gaussian processes, where independent local "expert" models operate on subsets of the data to distribute computation. These local models typically require stationary kernels to define concepts of "distance" and "locality." Shen et al. (2006) [23]used KD trees to recursively partition the data space into a multi-resolution tree structure, allowing GPs to scale up to $O(10^4)$ training points. However, this approach does not provide variance prediction solutions and is limited to stationary kernels.

Along the lines of exploiting locality, mixture-of-experts (MoE) models (Jacobs et al., 1991)[24] have been applied to GP regression (Rasmussen & Ghahramani, 2002[25]; Meeds & Osindero, 2006[23]; Yuan & Neubauer, 2009[26]). However, these models are primarily used to enhance model expressiveness, such as by allowing heteroscedasticity and non-stationarity, rather than to accelerate GP regression. Each local model has its own set of hyperparameters that need to be optimized. Predictions are made by aggregating the predictions of all local expert models, weighted by responsibilities assigned by a gating network. Closed-form inference in such models is intractable and typically requires MCMC approximations. Nguyen & Bonilla (2014) [27]avoided MCMC inference and accelerated the GP-MoE model by (i) fixing the number of GP experts and (ii) combining it with the pseudo-input sparse approximation by Snelson & Ghahramani (2006)[28]. This approach assigns data points probabilistically to experts using proximity information provided by stationary kernels, enabling GPs to scale up to $O(10^5)$ data points.

Product-of-GP-experts models (PoEs) bypass the weight assignment problem in mixture models: since PoEs obtain the overall prediction by multiplying predictions

made by independent GP experts, each expert's contribution is naturally weighted. However, this model tends to be overconfident (Ng & Deisenroth, 2014[29]). Cao and Fleet (2014)[30] recently proposed a generalized PoE-GP model in which each expert's contribution to the overall prediction can be individually weighted. This model is often too conservative, i.e., it overestimates variances. Tresp's Bayesian Committee Machine (BCM) [31] can be considered a PoE-GP model that provides a consistent framework for combining independent estimators within the Bayesian framework, but it suffers from the "weak experts" problem.

3.1.2.2 Scalable GP framework with ADMM

Early distributed GP models were either trained with the full dataset or entirely in a computing center [32]. In [33], the computational load was distributed to multiple agents by formulating the problem as a consensus optimization problem, which was solved using the Alternating Direction Method of Multipliers (ADMM). Specifically, the framework consists of a central node and multiple local nodes running independently on parallel agents.

During the training phase, each local node trains a GP model based on a subset of the data split from the training set and performs joint optimization with the central node via the ADMM algorithm. During the prediction phase, each local node performs predictions on validation points and test points based on its trained GP model. Subsequently, the central node first calculates the optimal fusion weights based on prediction performance on validation points, and then combines the local predictions on test points using the calculated weights to produce a better and more robust global prediction.

In general, the main responsibilities of the central node include:

1. Distributing subsets of the full dataset to non-central nodes at the initial stage;
2. Updating global ADMM parameters based on the local ADMM parameters received from other non-central nodes during the training phase;
3. Deriving fusion weights based on the local prediction results from other non-central nodes during the prediction phase.

Thus, by simply redirecting the data flow from other non-central nodes to a newly selected central node, the functionalities of the central node can be transferred to any non-central node, and the global ADMM parameters and fusion weights can be fully recovered. On the other hand, a failure in a non-central node only results in the loss of information for one subset rather than a total failure. However, it should be noted that the centralized scheme can be transformed into a fully distributed scheme by allowing each local node to individually select data subsets, update global parameters, and fuse weights. For example:

1. Each local node can independently select its subset from the full dataset;
2. Each local node can broadcast its local ADMM parameters, enabling all nodes to individually compute the global ADMM parameters;

3. Each local node can broadcast its local prediction results, enabling all nodes to independently output the final prediction result.

Learning objectives

In general, when predicting different data profiles, the hyper-parameters can be initialized with a set of universal default values. However, specifying the initial hyper-parameters according to the observed primary patterns can help the tuning process start from a better initial point, thereby improving learning efficiency.

After initialization, the dominant method for tuning model parameters is to maximize the marginal likelihood function, which can be expressed in closed form as:

$$\ell(\boldsymbol{y};\boldsymbol{\theta}) = -\frac{1}{2}(\log|\mathbf{K}(\boldsymbol{\theta})| + \boldsymbol{y}^T\mathbf{K}^{-1}(\boldsymbol{\theta})\boldsymbol{y} + n\log(2\pi)),$$

where $|\cdot|$ denotes the determinant of a matrix. The model hyper-parameters $\boldsymbol{\theta}$ can equivalently be tuned by minimizing the negative log-likelihood function. Therefore, the learning objective of our prediction model can be written as:

$$\begin{aligned} \mathcal{P}_0 \quad & \min_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}) \\ \text{s.t.} \quad & \boldsymbol{\theta} \in \Theta, \end{aligned} \tag{3.5}$$

It is noteworthy that for most kernel functions, whether standalone or composite, the problem $\mathcal{P}_0$ is non-convex and often lacks favorable structures with respect to the hyper-parameters. Consequently, classical gradient descent methods, such as the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm and conjugate gradient methods, can be used to solve for the hyper-parameters, but they do not guarantee finding the global minimum of $\mathcal{P}_0$.

The workflow of gradient descent-based methods is as follows. At each iteration, the hyper-parameters are updated according to the formula:

$$\boldsymbol{\theta}_i^{r+1} = \boldsymbol{\theta}_i^r - \eta \cdot \frac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_i}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}^r}, \quad \forall i = 1, 2, \cdots, p,$$

where η is the step size. The derivative of the i-th hyper-parameter is computed as:

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_i} = \mathrm{Tr}\left(\mathbf{K}^{-1}(\boldsymbol{\theta}) - \boldsymbol{\gamma}\boldsymbol{\gamma}^T\right)\frac{\partial \mathbf{K}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_i}, \tag{3.6}$$

where $\mathrm{Tr}(\cdot)$ represents the trace of a matrix, and $\gamma \triangleq \mathbf{K}^{-1}(\boldsymbol{\theta})y$. Note that $\mathbf{K}^{-1}(\boldsymbol{\theta})$ must be re-evaluated at each gradient step. Such matrix inversion requires $O(N^3)$ computations, where N is the number of training data points. This dominates the computational complexity of the standard GP.

ADMM-Based Scalable Training Framework

We aim to distribute the computation cost of solving problem $\mathcal{P}_0$ evenly across a set of parallel sensors to accelerate training. Given the full training dataset $D \triangleq \{X, y\}$, we define a set of N training subsets, denoted as $S \triangleq \{D^{(1)}, D^{(2)}, \cdots, D^{(N)}\}$. Each subset $D^{(i)} \triangleq \{X^{(i)}, y^{(i)}\}$ is sampled from the full dataset D, and each local GP model

is trained on its respective subset $D^{(i)}$. The standard parallel hyper-parameter training of the PoE can be expressed as

$$\mathcal{P}_1 : \min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \ell^{(i)}(\boldsymbol{\theta}), \quad \text{s.t. } \boldsymbol{\theta} \in \Theta, \tag{3.7}$$

where

$$\ell^{(i)}(\boldsymbol{\theta}) = \boldsymbol{y}^{(i)^\top} \left(\mathbf{K}^{(i)}(\boldsymbol{\theta})\right)^{-1} \boldsymbol{y}^{(i)} + \log \left|\mathbf{K}^{(i)}(\boldsymbol{\theta})\right|. \tag{3.8}$$

Thus, each local GP model only needs to optimize its own cost function with respect to the hyper-parameters. This optimization only requires operations on a small covariance matrix $K^{(i)}$ of size $m_i \times m_i$, where m_i denotes the number of data points in subset $D^{(i)}$, and $m_i \ll M$. Therefore, using standard GP implementation, the computational complexity for each local GP model is reduced to $O(m_i^3)$. Note that in this paper, all local GP models use equal-sized subsets, i.e., $m_i = \frac{M}{N}, \forall i = 1, 2, \cdots, N$, which could be further optimized in future work.

The core idea of PoE is to approximate the covariance matrix of the full dataset with a block-diagonal matrix of the same size. Each block corresponds to its respective subset. Therefore, ideally, the well-trained local hyper-parameters should be the same, i.e., $\boldsymbol{\theta}_i - \boldsymbol{\theta}_j = 0, \forall i, j$, to ensure that the block-diagonal approximation is consistent with the full matrix. Hence, existing PoE-based methods [34], [29], [30] assume that all local GP models are trained jointly and share the same set of global hyper-parameters $\boldsymbol{\theta}$. However, such joint training can only be realized through rigorous gradient consensus. Specifically, for each gradient step in Eq.3.6, the local cost and derivative information must be collected and coordinated to form the global cost and derivative. This global information is then used to update the globally shared hyper-parameters $\boldsymbol{\theta}$, which are subsequently sent back to each local GP model. However, the gradient consensus steps require $n_{\text{grads}} \cdot (2 \cdot \dim(\boldsymbol{\theta}) + 1)$ communication overheads, where n_{grads} is the number of gradient steps and $\dim(\boldsymbol{\theta})$ is the number of hyper-parameters to optimize. Moreover, the strict synchronization required for each gradient step limits its practical application in real systems.

The aim is to empower the distributed GP model with the powerful ADMM algorithm to develop a principled parallel training framework. The ADMM-enabled framework allows each distributed unit to train independently and coordinate with much lower communication overhead. The proposed training framework could lay the foundation for future scalable GP system design.

In general, ADMM takes the form of a decomposition-coordination procedure, where the original large problem is decomposed into smaller local subproblems that can be solved in a coordinated manner [35]. Based on ADMM, problem P_1 can be equivalently reformulated by introducing local variables $\boldsymbol{\theta}_i$ and a common global variable z as:

$$\begin{aligned} \mathcal{P}_2 : \quad & \min_{\boldsymbol{\theta}_i} \sum_{i=1}^{N} \ell^{(i)}(\boldsymbol{\theta}_i), \\ \text{s.t.} \quad & \boldsymbol{\theta}_i - \boldsymbol{z} = 0, \quad \boldsymbol{\theta}_i \in \Theta, \quad i = 1, 2, \ldots, N. \end{aligned} \tag{3.9}$$

Note that the above problems $\mathcal{P}_1$ and $\mathcal{P}_2$ are equivalent. However, with this new formulation, each local GP model is free to train its local hyper-parameters $\boldsymbol{\theta}_i$ based

on the local subset $D^{(i)}$, and the local hyper-parameters will eventually converge to the global hyper-parameters $\boldsymbol{z}$ after a few ADMM iterations.

To solve problem $\mathcal{P}_2$ using ADMM, we define the augmented Lagrangian as:

$$\mathcal{L}(\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_N, \boldsymbol{\zeta}_1, \cdots, \boldsymbol{\zeta}_N, \boldsymbol{z}) \triangleq \sum_{i=1}^{N} \left[\ell^{(i)}(\boldsymbol{\theta}_i) + \boldsymbol{\zeta}_i^T(\boldsymbol{\theta}_i - \boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{\theta}_i - \boldsymbol{z}\|_2^2 \right]. \tag{3.10}$$

where ζ_i is the dual variable, and $\rho > 0$ is a fixed augmented Lagrangian parameter.

The sequential updates of the ADMM parameters at the $(r+1)^{th}$ iteration can be expressed as:

$$\boldsymbol{\theta}_i^{r+1} := \arg\min_{\boldsymbol{\theta}_i} \left[\ell^{(i)}(\boldsymbol{\theta}_i) + \boldsymbol{\zeta}_i^T(\boldsymbol{\theta}_i - \boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{\theta}_i - \boldsymbol{z}\|_2^2 \right], \tag{3.11}$$

$$\boldsymbol{z}^{r+1} := \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\theta}_i^{r+1} + \frac{1}{\rho}\boldsymbol{\zeta}_i^r, \tag{3.12}$$

$$\boldsymbol{\zeta}_i^{r+1} := \boldsymbol{\zeta}_i^r + \rho\left(\boldsymbol{\theta}_i^{r+1} - \boldsymbol{z}^{r+1}\right). \tag{3.13}$$

The ADMM-based consensus only requires $n_{\text{cons}} \cdot (2 \cdot \dim(\boldsymbol{\theta}) + 1)$ communication overheads, where n_{cons} is the number of ADMM iterations, and $n_{\text{cons}} \ll n_{\text{grads}}$. Another advantage of the ADMM-based consensus is that if a local GP model gets stuck in a bad local minimum, the global consensus can restart from a more reasonable point to achieve better convergence in the next iteration.

The optimality conditions for the ADMM solution are determined by the primal residuals Δ_p and dual residuals Δ_d [35]. For each local GP model, these residuals can be expressed as:

$$\Delta_{i,p}^{r+1} = \boldsymbol{\theta}_i^{r+1} - \boldsymbol{z}^{r+1}, \quad i = 1, 2, \cdots, K, \tag{3.14}$$

$$\Delta_d^{r+1} = \rho(\boldsymbol{z}^{r+1} - \boldsymbol{z}^r). \tag{3.15}$$

These residuals will converge to zero as ADMM iterates. Thus, the stopping criteria for our problem include $\|\Delta_p^r\|_2 \leq \text{pri}$ and $\|\Delta_d^r\|_2 \leq \text{dual}$, where pri and dual are feasibility tolerance constants for the primal and dual residuals, respectively. As suggested in [35], they can be set as:

$$\epsilon^{\text{pri}} = \sqrt{p}\epsilon^{\text{abs}} + \epsilon^{\text{rel}} \max\left\{\|\boldsymbol{\theta}_i^r\|_2, \|\boldsymbol{z}^r\|_2\right\}, \tag{3.16}$$

$$\epsilon^{\text{dual}} = \sqrt{p}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\rho\|\boldsymbol{\zeta}^r\|_2, \tag{3.17}$$

where p is the dimension of θ in l_2 norm.

The ADMM-based GP training procedure is summarized in Algorithm 1. It should be noted that even for convex problems, the convergence of ADMM can be slow [35]. However, in practical applications, a few iterations are often sufficient to achieve an acceptable accuracy level.

Algorithm 1 ADMM GP Training

Initialization: $r \leftarrow 0$, N, θ_i^0, ζ_i^0, ρ, $\boldsymbol{z}^0 \leftarrow \frac{1}{N}\sum_{i=1}^{N}\left(\theta_i^0 + \frac{1}{\rho}\zeta_i^0\right)$, tolerances ϵ^{abs}, ϵ^{rel}.
Iteration:
while $\|\Delta_{i,p}^r\|^2 \geq \epsilon^{\text{pri}} \ \vee \ \|\Delta_d^r\|^2 \geq \epsilon^{\text{dual}}$ **do**
    $r \leftarrow r + 1$
    for $i \leftarrow 1$ to K **do**
        Obtain the parameters for local GP i by Eq. (3.11).
    end for
    Obtain the global parameters by Eq. (3.12).
    Obtain the dual variable by Eq. (3.13).
    Calculate the primal residuals $\Delta_{i,p}^{r+1}$ and the dual residuals Δ_d^{r+1} by Eq. (3.14) and Eq. (3.15), respectively.
    Update the feasibility tolerances ϵ^{pri} and ϵ^{dual} by Eq. (3.16) and Eq. (3.17).
end while
Output: Global parameter $\boldsymbol{z}^r$.

3.1.3 Simulation

3.1.3.1 simulation setup

To better investigate the problem of cooperative perception and tracking in multi-agent systems, we design our simulation scenario based on a widely studied and practically significant application: **Search and Rescue (SAR)** operations at sea. In such tasks, the target (e.g., a life raft or a person in the water) typically loses active control and drifts passively under the influence of ocean currents and wind. Previous studies have shown that these targets often exhibit *nearly circular trajectories* with *approximately constant speeds* [36, 37].

Target Dynamic Model

Motivated by these characteristics, we model the target as a weakly nonlinear system moving at a constant linear and angular velocity, which approximates the drifting behavior of a life raft in a marine environment. The target state is defined as $\mathbf{x}_k = [x_k, y_k, \theta_k]^\top$, representing its 2D position and heading angle. Its motion is governed by the following discrete-time nonlinear dynamics:

$$x_{k+1} = x_k + v\cos(\theta_k)\cdot\Delta t + w_x \tag{3.18}$$
$$y_{k+1} = y_k + v\sin(\theta_k)\cdot\Delta t + w_y \tag{3.19}$$
$$\theta_{k+1} = \theta_k + \omega\cdot\Delta t + w_\theta \tag{3.20}$$

The control input is fixed as $\mathbf{u}_k = [v_k, \omega_k]^\top = [1.0, 0.1]^\top$, corresponding to constant forward and angular velocities. The process noise $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q})$ models environmental disturbances, with covariance matrix $\mathbf{Q} = \text{diag}(0.01, 0.01, 0.0005)$. The initial state is set to $[x_0, y_0, \theta_0]^\top = [0, 0, \pi/4]^\top$, and the simulation runs for $T = 100$ seconds with a time step of $\Delta t = 0.1$ seconds, resulting in 1000 discrete steps.

This model realistically simulates the weakly controlled drifting motion of rescue targets in SAR scenarios, providing a reliable basis for evaluating distributed tracking algorithms.

Agent Measurement Model

To observe the target, multiple cooperative agents are deployed, representing UAVs or floating sensors. Each agent is capable of measuring the full target state, and the observation model is given by:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{3.21}$$

Here, $\mathbf{H} = \mathbf{I}_{3\times 3}$ indicates that all state components (position and heading) are directly observable, and the measurement noise is modeled as $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R})$. This simplified linear measurement model enables tractable simulation of multi-agent information fusion and collaborative estimation.

Besides, to better simulate realistic conditions in maritime search and rescue missions, we designed and evaluated two representative distributed tracking scenarios.

In the first scenario, eight unmanned aerial vehicles (UAVs) are evenly spaced around the drifting target in a structured formation. Each UAV is equipped with an inertial measurement unit (IMU) and is capable of directly observing the target whenever it is within its communication radius. The UAVs are allowed to exchange information with neighboring agents and collaboratively estimate the target's state using a distributed tracking algorithm. This setup mimics a well-coordinated UAV fleet with organized deployment and stable inter-agent communication.

In the second scenario, the eight UAVs are randomly deployed within a predefined area, simulating an unstructured and reactive response to a rescue mission. Each UAV still has onboard IMU sensing and can measure the target when within communication range, but the network topology is randomly determined based on initial positions, introducing higher variability and potential challenges for collaborative estimation.

In both settings, we tested the performance of a distributed Gaussian Process (DGP)-based tracking algorithm and quantitatively evaluated its estimation accuracy. We also analyzed how varying the communication radius impacts the overall tracking performance. This comparative study aims to reveal how communication constraints and agent deployment strategies influence distributed tracking outcomes, providing insights for the design and optimization of UAV coordination strategies in real-world SAR applications.

3.1.3.2 Scenario 1: Uniform UAV Deployment

In this scenario, eight UAVs are placed at equal intervals around the target. Each UAV has the same measurement range and is able to communicate with its neighboring UAVs within a predefined communication radius. The network thus forms a symmetric topology suitable for cooperative distributed tracking.

In practical applications, an agent's sensing or communication radius is often constrained by sensor performance, energy budget, and environmental occlusions, making global or unlimited observation impossible. Consequently, each agent can only collect local information about the target within a limited area around its position, and the

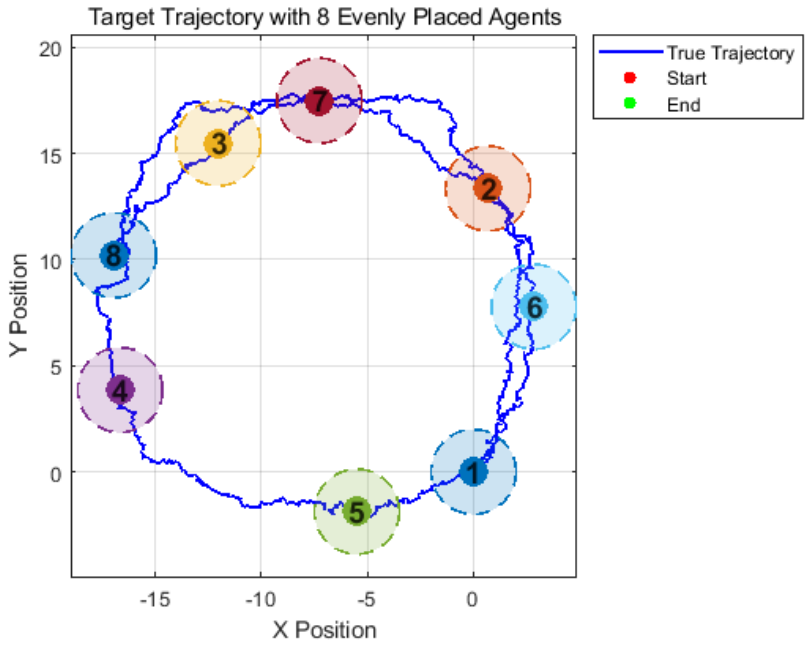
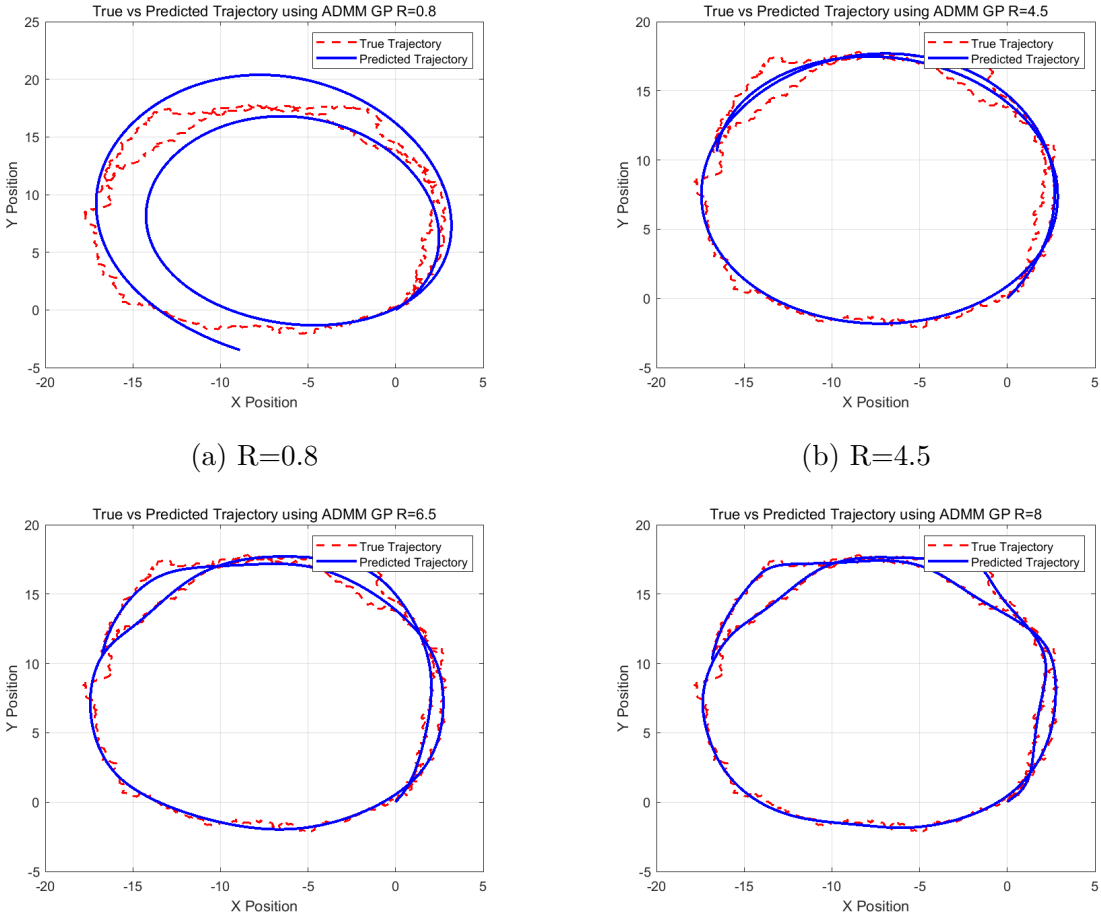


Figure 3.2: Uniform UAV Deployment

tracking accuracy inevitably depends on the size of this measurement radius. A radius that is too small leads to incomplete network coverage and sparse observations, hindering high-precision estimation; conversely, a radius that is too large increases the amount of observation data but incurs higher energy consumption and hardware costs. Therefore, this study systematically investigates the effects of different measurement radii—ranging from $R = 2.0$ to 8.0—on distributed tracking performance (RMSE) and sample size, aiming to provide quantitative guidance for designing sensor networks under resource constraints and to identify the optimal trade-off between accuracy and cost.

Table 3.2: Effect of Communication Radius R on Distributed Tracking Accuracy

R	**RMSE$_X$**	**RMSE$_Y$**	**RMSE$_\theta$**	**Overall RMSE**	**Sample Size**
2.0	4.214	3.498	0.066	3.162	461
2.5	3.215	2.857	0.022	2.483	628
3.0	0.739	0.700	0.129	0.592	774
3.5	0.631	0.649	0.409	0.574	913
4.0	0.647	0.688	0.029	0.546	1051
4.5	**0.413**	**0.504**	**0.011**	**0.376**	**1178**
5.0	0.717	0.690	0.023	0.575	1327
5.5	0.827	0.569	0.010	0.580	1462
6.0	0.497	0.299	0.019	0.335	1596
7.0	0.381	0.268	0.010	0.269	1886
8.0	0.189	0.220	0.009	0.168	2200



(a) R=0.8 (b) R=4.5

(c) R=6.5 (d) R=8

Figure 3.3: Scenario 1: Uniform UAV Deployment

Table 3.2 reports the influence of the communication/measurement radius R on the tracking accuracy and the total number of valid observations (sample size) in the distributed estimation network. Here,

- R is the maximum distance within which each agent can directly measure the target.
- RMSE_X, RMSE_Y, and RMSE_θ denote the root-mean-square errors in the X, Y, and orientation θ components, respectively.
- *Overall RMSE* is the combined root-mean-square error over all state dimensions.
- *Sample Size* is the total number of valid measurements collected by all agents at the given radius R.

As R grows from 2.0 to 4.5, the overall RMSE steadily decreases (from approximately 2.93 to 2.04) while the sample size increases (from 334 to 1163). Beyond $R = 4.5$, although sample size continues to grow (up to 2248 at $R = 8.0$), the reduction in

overall RMSE becomes marginal, indicating diminishing returns: once network connectivity and coverage approach saturation, further increases in R mainly add redundant observations without significant accuracy gains.

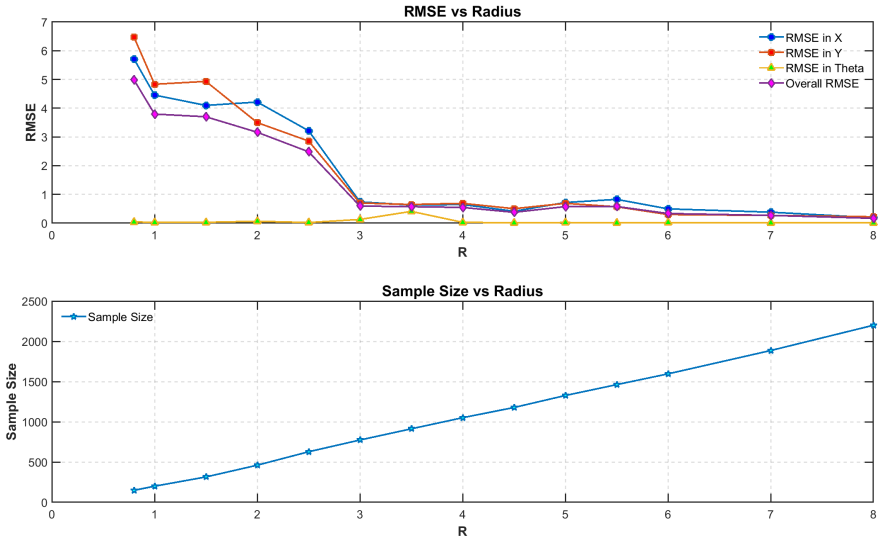


Figure 3.4: Top: RMSE of different state dimensions (X, Y, θ) and overall RMSE as a function of communication radius R. Bottom: Corresponding number of valid samples observed by the uniform UAV network.

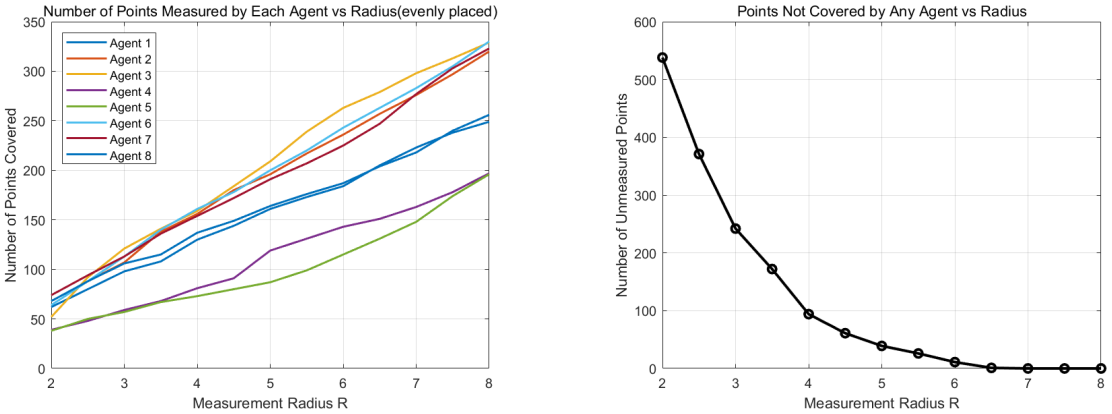
As illustrated in Figure 3.9, the RMSEs of target state estimation across all dimensions (x, y, and θ) generally decrease as the communication radius R increases. A significant drop in error occurs around $R = 3$, indicating that when the communication range is limited, each UAV has access to only a small number of observations, which restricts the effectiveness of local modeling and distributed information fusion. As R increases, the number of valid samples per agent rises considerably (as shown in the bottom plot), enabling better learning of the target dynamics and thereby improving tracking accuracy.

Interestingly, beyond $R = 4.5$, the RMSE does not continue to decrease. Instead, it fluctuates or even slightly increases. This suggests that an overly large communication radius may introduce redundant or noisy observations, or potentially disrupt the sparsity structure that benefits local fusion. Such over-communication could lead to degraded estimation performance or inefficient resource usage.

Overall, the effect of communication radius on distributed tracking accuracy is nonlinear. There exists an optimal range in which estimation performance is maximized. In this experiment, the lowest overall RMSE is achieved at $R = 4.5$, indicating that a moderate communication range strikes a balance between data availability and network efficiency. This insight is highly relevant for real-world maritime UAV deployments, where communication constraints and estimation accuracy must be carefully balanced.

Table 3.3: Coverage, Overlap and RMSE vs Measurement Radius(Scenario 1)

Radius	**Uncovered Points**	**Overlap ($\geq$ 2 Agents)**	**RMSE**
2.0	538	0	4.214076
2.5	371	0	3.215460
3.0	242	17	0.739453
3.5	172	86	0.631314
4.0	94	146	0.647200
4.5	61	240	**0.413160**
5.0	39	367	0.717163
5.5	26	489	0.827449
6.0	11	608	0.496550
6.5	1	734	0.495950
7.0	0	859	0.381459
7.5	0	915	0.188687
8.0	0	938	0.188687



(a) Number of Points Measured vs Radius

(b) Number of Unmeasured Points vs Radius

Figure 3.5: Coverage Analysis with Varying Measurement Radius

Then we focus on the influnce of measurement radius to the performance.The experimental results (Table 3.3 and Figure 3.4) show that as the measurement radius R increases from 2.0 to 4.0, the number of uncovered points rapidly decreases from 538 to 94, shifting network coverage from sparse to near-global, while the overall RMSE decreases from 4.21 to 1.78; meanwhile, the overlap count rises from 0 to 146, indicating the onset of redundant observations that enhance robustness against noise and single-point failures. Further increasing R to the 4.5–6.0 interval eliminates nearly all uncovered points and boosts overlap to 608, reducing the RMSE to its minimum of 0.413; beyond $R = 6.5$, both coverage and redundancy saturate (uncovered ≈ 0, overlap ≈ 938) and RMSE reduction plateaus at 0.189, demonstrating diminishing returns. Therefore, choosing $R \approx 4.5$–6.0 balances global coverage and sufficient redundancy for high-precision tracking while avoiding unnecessary energy and hardware costs, making it the ideal design range for resource-constrained distributed tracking systems.

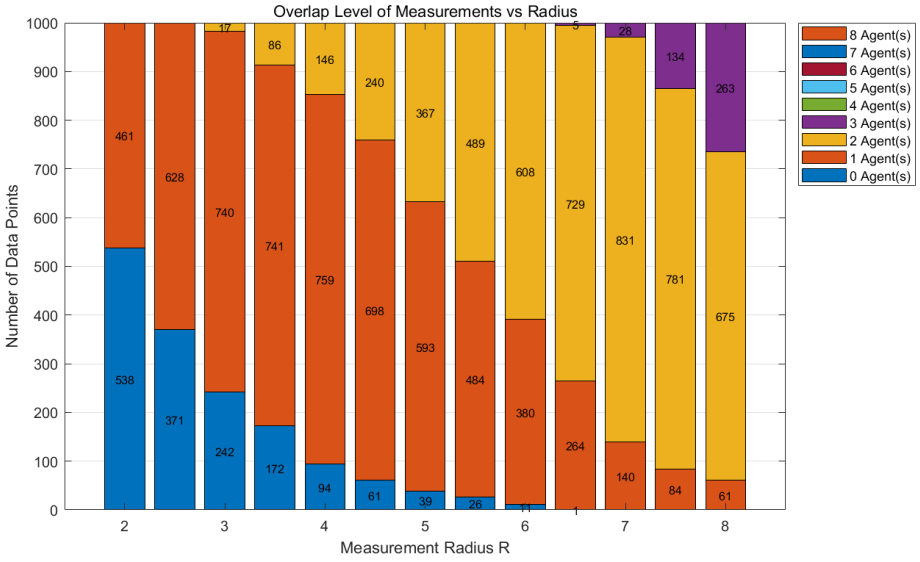


Figure 3.6: Stacked bar chart showing how many data points are covered by 0 to 8 agents at different measurement radii. The bottom segments represent uncovered points, while upper segments indicate increasing overlap.

Figure3.6 stacked bar chart illustrates how the number of data points covered by 0–8 agents evolves as the measurement radius R varies, and explains its direct link to tracking accuracy (RMSE). At small radii ($R = 2.0$–3.0), the chart is dominated by blue and light-blue segments—538 and 371 uncovered points at $R = 2.0$, 242 and 740 at $R = 3.0$—indicating sparse coverage and many singly observed or unobserved points; correspondingly, the RMSE remains high (4.21 at $R = 2.0$, 3.22 at $R = 2.5$). As R grows to 4.0–4.5, uncovered points plunge (to 94 and 61), and moderate overlap (146–240 multi-agent overlaps) appears, coinciding with the lowest RMSE values (0.647 at $R = 4.0$, 0.413 at $R = 4.5$). Beyond $R = 5.0$, overlap segments swell—reaching 608 at $R = 6.0$ and 938 at $R = 8.0$—but RMSE improvements taper off (0.496→0.381→0.189), revealing diminishing returns: excessive redundancy no longer yields meaningful accuracy gains yet incurs extra energy and communication costs. Thus, an intermediate radius ($R \approx 4.0$–5.5) optimally balances full coverage, beneficial overlap, and minimal RMSE for resource-constrained distributed tracking systems.

3.1.3.3 Scenario 2: Random UAV Deployment

In this scenario, eight UAVs are randomly deployed within the area surrounding the target. Each UAV has the same measurement range and is able to communicate with neighboring UAVs within a predefined communication radius. Since the spatial positions of the UAVs are no longer regularly arranged, the resulting network topology is asymmetric and varies with deployment configuration. This setup more closely reflects the distributed nature of multi-agent systems in real-world search and rescue missions and is well-suited for studying the robustness and adaptability of cooperative

distributed tracking under uncertain conditions.

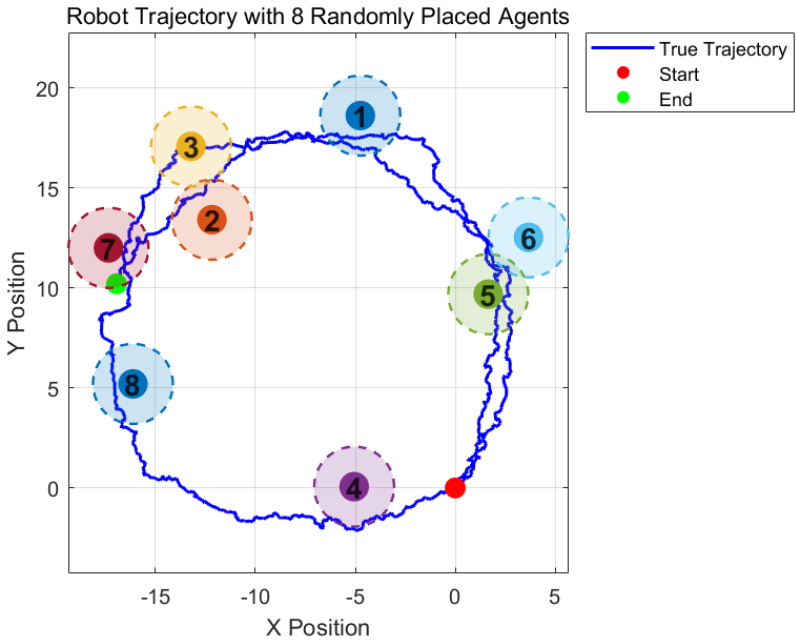


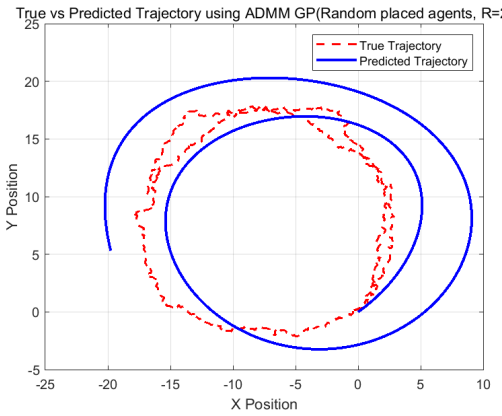
Figure 3.7: Random UAV Deployment

Table 3.4: Distributed Tracking RMSE Under Varying Communication Radius R (Scenario 2)

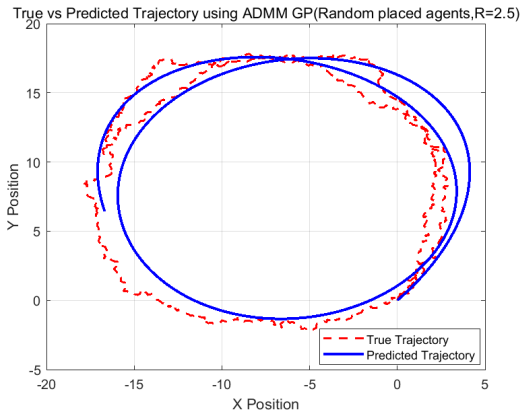
R	RMSE_X	RMSE_Y	RMSE_θ	Overall RMSE	Sample Size
2.0	4.059	3.017	0.082	2.920	334
2.5	2.296	2.284	0.062	1.870	498
3.0	8.193	4.855	0.039	5.498	649
3.5	**0.631**	**0.649**	**0.409**	**0.574**	**913**
4.0	2.079	2.272	0.076	1.778	1006
4.5	6.383	2.857	0.060	4.038	1163
5.0	10.829	6.209	0.031	7.207	1292
5.5	1.896	1.263	0.013	1.315	1462
6.0	1.525	1.113	0.015	1.090	1623
7.0	0.194	0.223	0.012	0.171	1941
8.0	0.191	0.218	0.010	0.167	2248

As shown in Figure 3.9 and Table 3.4, the influence of communication radius R on distributed tracking performance exhibits greater fluctuation and uncertainty under randomly deployed UAV networks. When the communication radius is small (e.g., $R = 2.0 \sim 2.5$), the number of available observations is limited, resulting in relatively high overall RMSE and restricted estimation performance.

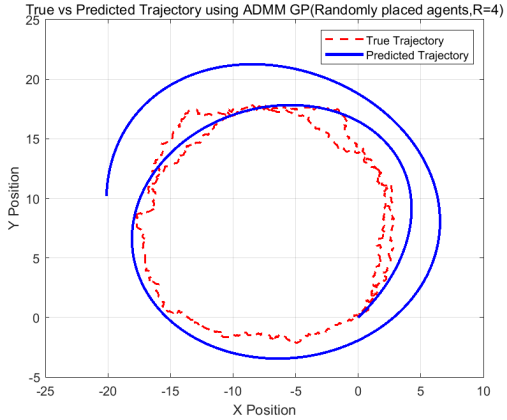
At $R = 3.5$, the estimation accuracy improves significantly, suggesting a favorable balance between network connectivity and sample coverage. However, as R increases



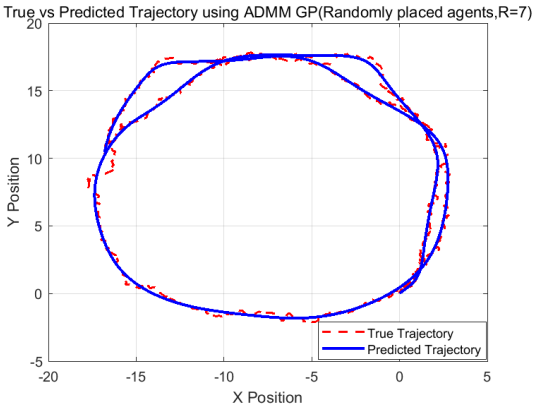
(a)R=2



(b) R=2.5



(c) R=4



(d) R=7

Figure 3.8: Scenario 2: Random UAV Deployment

further to 4.5 and 5.0, the RMSE spikes sharply—especially in the x and y dimensions—indicating that in random topologies, excessive communication or unbalanced connectivity may introduce redundant or conflicting information, leading to instability in local learning models.

Beyond $R = 6.0$, the RMSE drops again and stabilizes, implying that the system gradually regains cooperative estimation capability as the network becomes more fully connected.

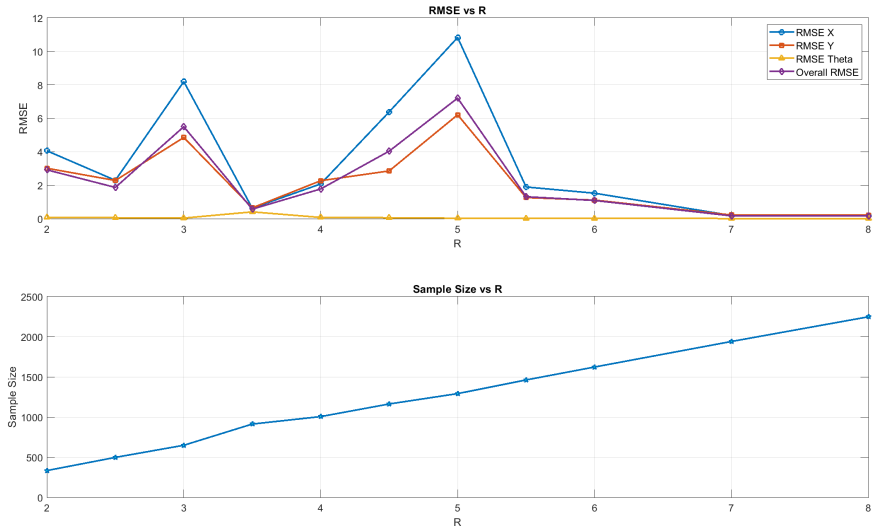
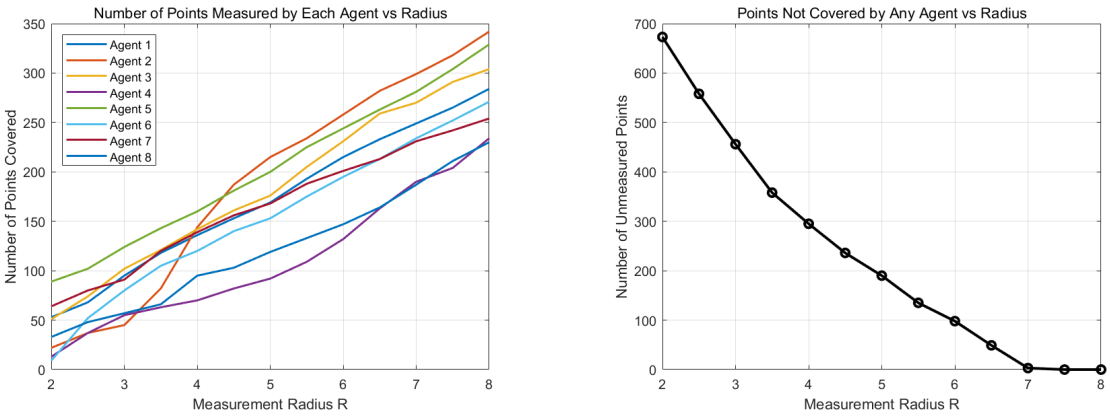


Figure 3.9: Top: RMSE of different state dimensions (X, Y, θ) and overall RMSE as a function of communication radius R. Bottom: Corresponding number of valid samples observed by the random UAV network.

Radius	**Uncovered**	**Overlap ($\geq$ 2 Agents)**	**RMSE**
2.0	673	8	2.920
2.5	558	57	1.870
3.0	456	106	5.498
3.5	358	174	**0.574**
4.0	295	283	1.778
4.5	236	352	4.038
5.0	190	418	7.207
5.5	135	475	1.315
6.0	98	515	1.090
7.0	3	579	0.171
8.0	0	722	0.168

Table 3.5: Coverage, Overlap and RMSE vs Measurement Radius(Scenario 2)



(a) Number of Points Measured vs Radius

(b) Number of Unmeasured Points vs Radius

Figure 3.10: Coverage Analysis with Varying Measurement Radius(with randomly placed agents)

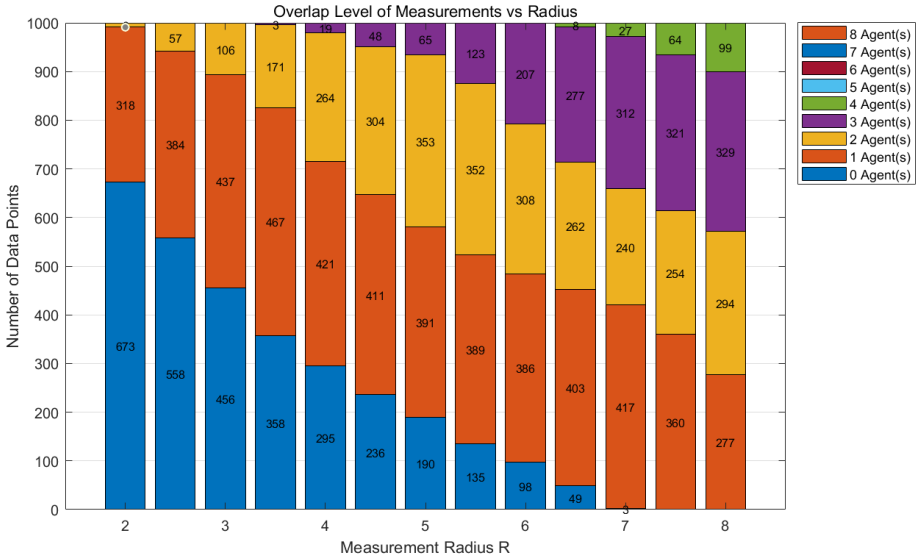


Figure 3.11: Stacked bar chart showing how many data points are covered by 0 to 8 agents at different measurement radii. The bottom segments represent uncovered points, while upper segments indicate increasing overlap.

Next, we analyze the effect of the measurement radius on the performance of the distributed tracking algorithm. Specifically, we focus on how the measurement coverage and observation redundancy (i.e., the number of target points simultaneously observed by multiple agents) jointly influence the error trends, and what underlying factors may explain the observed behaviors.

As shown in Figure 3.10 and the accompanying table, increasing the measurement radius R significantly enhances the system's coverage of the target space: the number of unmeasured points decreases from 673 at $R = 2.0$ to 0 at $R = 8.0$, indicating that the target area becomes fully observable. This expansion in coverage correlates with a substantial reduction in the overall RMSE, which drops from 2.920 to 0.168, suggesting that broader spatial observability leads to more accurate state estimation. In addition, the number of points observed by each individual agent also increases with R, reflecting a global increase in available information. This growing coverage is a primary driver of performance improvement, especially in the range $R = 2.0 \sim 4.0$, where the system transitions from sparse to near-complete measurement.

However, the error trend is not strictly monotonic. At certain measurement radii—particularly $R = 3.0$ and $R = 5.0$—the RMSE unexpectedly spikes to 5.498 and 7.207, respectively, despite a clear reduction in unmeasured points and an increase in overlapping observations (106 and 418 points observed by at least two agents). This indicates that while coverage and redundancy are necessary, they are not sufficient to guarantee consistent estimation performance. A deeper investigation reveals that changes in measurement radius not only affect whether a target point is observed, but also alter the spatial distribution of those observations. At certain values of R, observation points may become heavily concentrated in specific local regions, while distant areas remain sparsely measured, leading to imbalanced information and esti-

mation bias. Furthermore, when redundant observations are primarily concentrated in regions with high measurement noise or systematic bias, they may amplify errors rather than suppress them. This is particularly critical when the fusion model lacks robustness against correlated or skewed inputs. Therefore, even under high coverage and redundancy, estimation performance may degrade due to poor spatial distribution or low-quality overlapping measurements. These findings highlight the importance of not only increasing coverage and redundancy, but also ensuring balanced spatial sampling and stable fusion processes to achieve robust and accurate multi-agent tracking performance.

In contrast to structured deployments, randomly placed UAV networks are more sensitive to the selection of communication radius, facing dual risks of "Measurement overload" and "insufficient observations." This highlights the need for communication strategies that balance robustness with efficiency in decentralized tracking systems.

3.1.4 Comparative Discussion and Analysis of Deployment Strategies

By comparing the experimental results under the two UAV deployment strategies, we observe that the uniformly distributed UAV network demonstrates more stable and controllable estimation performance in distributed target tracking tasks. As the communication radius R increases, the overall RMSE shows a monotonic decrease, reaching its minimum around $R = 4.5$. This indicates that the symmetric spatial layout and balanced network structure facilitate consistent information fusion and local modeling, thereby improving the performance of the distributed Gaussian process framework in cooperative tracking.

In contrast, the random deployment scenario exhibits significantly greater fluctuations in RMSE, especially in the mid-range values of R (e.g., $R = 3 \sim 5$), where sudden spikes in estimation error are observed. Although the RMSE tends to decrease again as $R \geq 6$, the overall performance remains highly sensitive to the choice of communication radius and lacks robustness compared to the uniform case.

The underlying reason for this performance difference lies in the structural properties of the network topology. In the uniform deployment, UAVs are symmetrically spaced, ensuring evenly distributed connections and continuous coverage of the target area. This facilitates efficient information sharing and stable inference. In contrast, random deployment leads to potential clustering and sparse zones within the network. Under certain communication ranges, this irregular connectivity may result in redundant or conflicting observations, degrading local estimation and increasing global RMSE.

These findings suggest that structured deployment offers a more robust foundation for collaborative estimation and is well-suited for predefined mission formations. Meanwhile, random deployment better reflects realistic emergency scenarios and poses higher demands on algorithmic adaptability and robust communication strategies. Our study highlights the critical impact of deployment geometry on distributed tracking performance and emphasizes the need for careful balance between topological control and environmental adaptability in multi-agent system design.

3.2 Cooperative Localization

In this section, we investigate cooperative self-localization in multi-agent systems where reliable GNSS is unavailable. To address the limitation that individual agents can only obtain noisy relative observations, we propose a neighbor-distance-based localization model that formulates position estimation as an optimization problem minimizing the discrepancy between measured and estimated inter-agent distances, and implement a decentralized iterative update algorithm. Through simulations, we analyze the impact of communication radius and measurement noise on localization performance, showing that network connectivity is the key to ensuring convergence, while measurement noise primarily determines the achievable accuracy. Overall, the results demonstrate that the proposed method enables robust self-localization in infrastructure-free, decentralized environments, thereby providing reliable position information for subsequent cooperative target tracking and formation control.

3.2.1 Cooperative-localization

In the previous section, we described an idealized scenario in which each agent has access to the complete trajectory of the target within its measurement range, including full-state information such as position and orientation. However, such an assumption is rarely met in real-world multi-agent systems. In practice, due to distance measurement errors, sensor field-of-view limitations, occlusions, and environmental interference—especially in challenging maritime conditions—agents typically only observe partial aspects of the target state, most commonly relative quantities such as direction or distance, rather than full absolute coordinates in the global frame [38, 39].

This limitation places higher demands on the system's ability to infer global target dynamics from decentralized and noisy local observations. In maritime search and rescue (SAR) scenarios, for example, the sea surface environment is highly dynamic, communication is often intermittent, and GNSS signals may be unreliable or completely denied due to interference or terrain masking [40]. Under these conditions, single-agent observations are often sparse, noisy, and time-varying, making it difficult to construct a reliable target trajectory without multi-agent cooperation. Therefore, efficient information fusion among agents becomes essential for improving the robustness and accuracy of distributed target tracking [41, 42].

More critically, when an agent lacks access to the absolute position of the target, it becomes essential that the agent has accurate knowledge of its own location. Otherwise, the relative observations it collects are rendered meaningless in a global context, degrading the consistency and reliability of the overall tracking framework. This interdependence highlights the importance of agent self-localization in cooperative tracking tasks. A poorly localized agent introduces uncertainty not only into its own observations, but also into the global tracking estimate [43].

This motivates our focus on *cooperative self-localization*, wherein agents iteratively estimate their own positions based on inter-agent communication and local ranging. Accurate self-localization establishes a shared spatial reference frame across the network, thereby enabling consistent data association and improving overall tracking fidelity. The following section presents the modeling assumptions, simulation framework, and

performance analysis of the proposed self-localization system, which forms a critical backbone for subsequent distributed perception and coordination tasks.

3.2.2 Motivation

In practical scenarios, GNSS signals are often rendered unreliable due to multipath effects, occlusions, or intermittent loss, necessitating that agents rely on onboard sensors and relative measurements for state estimation. In open environments without fixed infrastructure, centralized localization systems or pre-deployed anchors are generally infeasible. This motivates the development of *self-localization* capabilities in multi-agent systems, wherein each agent autonomously estimates its position through local observations and inter-agent communication.

This process typically involves two stages: a measurement phase and a state update phase. Under conditions of limited communication range and sparse anchor availability, cooperative self-localization has been widely recognized as an effective strategy to enhance both localization accuracy and system robustness, and it has emerged as a central research problem in distributed multi-agent systems.

3.2.3 Neighbor-Distance-Based Cooperative Localization

In distributed multi-agent systems, where global positioning infrastructure such as GNSS is unavailable or unreliable, agents must estimate their positions using local sensing and inter-agent ranging. To address this challenge, we develop a neighbor-based cooperative self-localization method that enables decentralized position estimation through iterative distance-based optimization among agents.

3.2.3.1 Problem Formulation

We consider a two-dimensional network of N agents, the whole network is defined as $\mathcal{F}$, among which a subset $\mathcal{A} \subset \{1, 2, ..., N\}$ are anchors with known positions. The remaining agents$\mathcal{M} = \mathcal{F} \setminus \mathcal{A}$ must estimate their positions $\hat{\mathbf{p}}_i \in \mathbb{R}^2$ using noisy relative distance measurements with their neighbors. Each agent i communicates with agents j within a communication radius R_c, defining its neighbor set as:

$$\mathcal{N}_i = \{j \in \{1, \ldots, N\} \setminus \{i\} : \|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j\| \leq R_c\}. \tag{3.22}$$

The distance measurement between agents i and j is modeled as:

$$d_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\| + \varepsilon_{ij}, \quad \varepsilon_{ij} \sim \mathcal{N}(0, \sigma^2), \tag{3.23}$$

where $\mathbf{p}_i$ denotes the true position of agent i, and ε_{ij} is zero-mean Gaussian measurement noise.

3.2.3.2 Optimization-Based Localization

Each non-anchor agent i estimates its position by minimizing the discrepancy between the measured and estimated distances to its neighbors. The local cost function is defined as:

$$J(\hat{\mathbf{p}}_i) = \sum_{j \in \mathcal{N}_i} \left(\|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j\| - d_{ij} \right)^2 . \tag{3.24}$$

This objective penalizes inconsistencies between the estimated positions and noisy measurements. Agents minimize J_i using local solvers such as quasi-Newton methods. Anchors do not update their positions and act as reference nodes.

3.2.4 Iterative Localization Scheme

The cooperative-localization algorithm proceeds iteratively as follows:

1. Initialize all agent positions with random offsets; anchor positions remain fixed.
2. For each iteration $t = 1, 2, \ldots, T$:
   - Each non-anchor agent i exchanges its estimated local positions of its current neighbors $\mathcal{N}_i$.
   - True inter-agent distances are computed and corrupted with Gaussian noise to simulate measurements.
   - Agent i minimizes $J(\hat{\mathbf{p}}_i)$ to obtain an updated position estimate.
3. Repeat until convergence or maximum iteration count is reached.

Algorithm 2 Iterative Cooperative-Localization Algorithm

Require: Anchor positions $\{\mathbf{p}_a\}_{a \in \mathcal{A}}$; max iterations T; noise variance σ^2; convergence threshold ϵ
Ensure: Estimated positions $\{\hat{\mathbf{p}}_i\}_{i \in \mathcal{N}}$ for non-anchor agents

Initialize $\hat{\mathbf{p}}_i$ with random offsets for all $i \in \mathcal{M}$
for $t \leftarrow 1$ to T **do**
  for all $i \in \mathcal{N}$ **do**
    Retrieve neighbor estimates $\{\hat{\mathbf{p}}_j\}_{j \in \mathcal{N}_i}$
    Measure distances $d_{ij} \leftarrow \|\mathbf{p}_i - \mathbf{p}_j\| + \mathcal{N}(0, \sigma^2)$ for $j \in \mathcal{N}_i$
    Store $\hat{\mathbf{p}}_i^{\text{prev}} \leftarrow \hat{\mathbf{p}}_i$
    Update estimate: $\hat{\mathbf{p}}_i \leftarrow \arg\min_{\mathbf{p}} J(\mathbf{p}; \{d_{ij}\}, \{\hat{\mathbf{p}}_j\})$
  end for
  Compute $\Delta \leftarrow \sum_{i \in \mathcal{F}} \|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_i^{\text{prev}}\|$
  if $\Delta < \epsilon$ **then**
    break ▷ Convergence reached
  end if
end for

3.2.5 Simulation

For a comprehensive evaluation of the algorithm's performance in complex network environments, we first fixed the ranging noise at $\sigma^2 = 0.1$ and increased the communication radius from 5m to 30m. The results exhibited a pronounced "peak–valley"

effect, with large error fluctuations in the critical connectivity region and rapid convergence once the network became fully connected. Next, under a communication radius sufficient to guarantee global connectivity (e.g., $r_c = 30$m), we gradually increased the ranging noise intensity from noiseless to $\sigma^2 = 0.5$. We observed that the localization error increased monotonically with noise level, yet remained relatively stable under low to moderate noise. These two complementary experiments clearly reveal the distinct roles of network connectivity and measurement uncertainty in distributed self-localization, and provide intuitive guidance for subsequent investigations.

3.2.5.1 Cooperative-localization performance under different communicational radius

In this experiment, we investigate how the cooperative-localization performance varies with different communication radii. Specifically, we fix the measurement noise at a constant level ($\sigma^2 = 0.1$) and progressively increase the communication radius from 5 m to 30 m. By analyzing the average localization error and the network connectivity at each radius, we aim to reveal the critical relationship between communication range and localization accuracy. This evaluation provides insights into how the cooperative localization transitions from local, loosely-connected clusters to a globally connected network, highlighting the trade-off between communication costs and positioning precision.

R_c	A1	A2	A3	A4	A5	A6	A7	A8	Avg Error (m)
5.0	0	0	0	0	0	0	0	0	4.2317
6.0	1	1	1	0	1	1	1	0	4.2742
7.0	1	1	1	1	1	1	1	1	5.6557
8.0	1	1	2	1	1	1	1	2	4.7698
9.0	2	2	2	1	1	2	2	2	3.2684
10.0	2	2	2	1	1	2	2	2	4.4638
11.0	2	2	2	1	1	2	2	2	4.7229
12.0	2	2	2	1	1	2	2	2	9.1719
13.0	2	3	4	3	3	3	3	3	8.6574
14.0	3	4	4	3	3	3	3	3	7.8351
15.0	3	4	4	3	3	4	4	3	4.8025
16.0	3	4	4	3	3	4	4	3	3.9122
17.0	3	5	5	4	5	5	5	4	0.3624
18.0	4	6	5	5	5	5	5	5	0.2169
19.0	5	6	6	5	6	5	6	5	0.3005
20.0	7	7	7	7	7	7	7	7	0.2062
30.0	7	7	7	7	7	7	7	7	0.2062

Table 3.6: Number of neighbors per agent and average localization error under various communication radii.

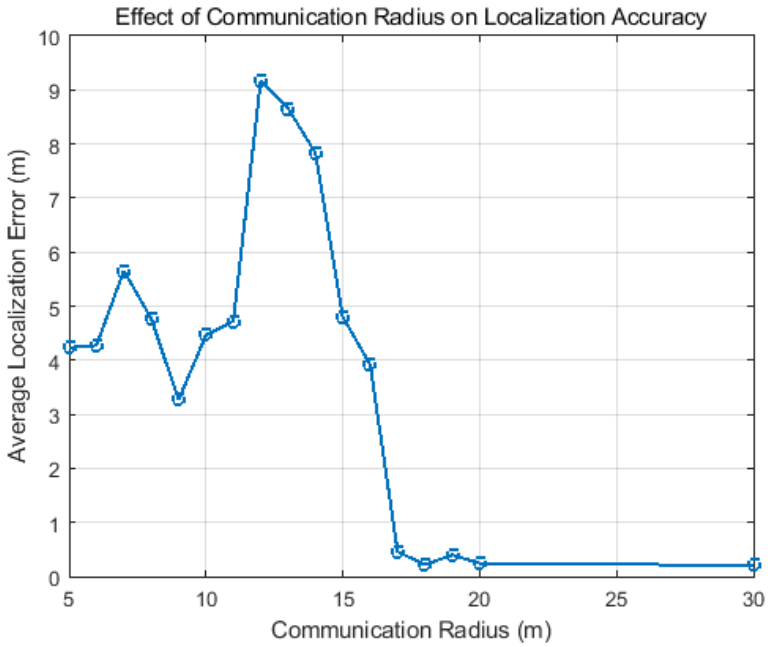


Figure 3.12: Combined plot of the number of neighbors per agent and the average localization error as a function of the communication radius.

The observed non-monotonic relationship between communication radius r_c and average localization error arises from the interplay of network connectivity and noisy distance measurements. When r_c is very small (below approximately 6m), most agents have few or no neighbors, so each non-anchor agent relies solely on its random initial offset; as a result, the error remains at roughly the magnitude of that initial perturbation (around 4m).

As r_c increases into the mid-range (6–12m),local clusters begin to form but the network is not yet globally connected. In this regime, agents receive more range constraints—each corrupted by Gaussian noise—but these constraints are still confined to small, disjoint subgraphs. The conflict among noisy, locally consistent measurements leads the optimizer to different local minima, causing the average error to rise and fluctuate, peaking near 9–10m.

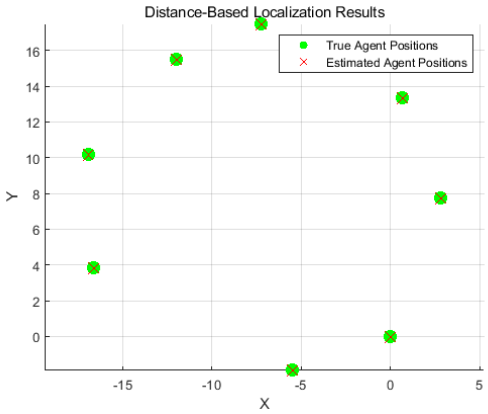
Beyond a critical threshold (around 13–16m), the communication graph undergoes a phase transition from disconnected or weakly connected fragments into a single connected component. Once every non-anchor agent has at least one path to an anchor, true anchor positions propagate through multiple hops, imposing a coherent global constraint. Consequently, the collective optimization aligns all estimates with the true trajectory, driving the average error sharply down to near zero for $r_c \gtrsim 17$m.

3.2.5.2 Cooperative-localization Performance under Varying Noise Levels

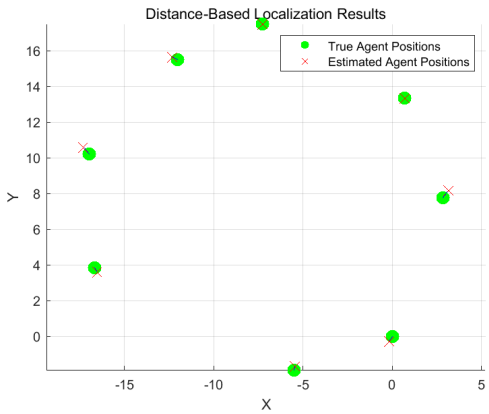
To evaluate the proposed method, we simulate a scenario with $N = 8$ UAV agents, randomly initialized within a bounded region. Two agents are selected as anchors. All agents use the same communication radius $R_c = 30$ meters, and Gaussian noise with

standard deviation $\sigma^2 = 0.1$ is added to simulate measurement uncertainty.

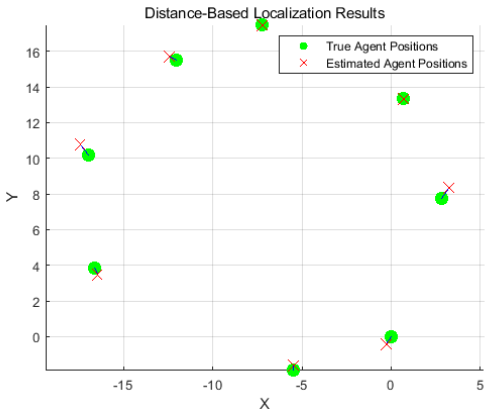
After 500 iterations, the algorithm achieves stable convergence with an average localization error below 1 kilometer. These results confirm the robustness and scalability of the proposed method in decentralized settings. More importantly, it demonstrates that in GNSS-denied maritime environments, such as UAV-based search and rescue missions, reliable agent localization can be achieved solely through local ranging and communication.



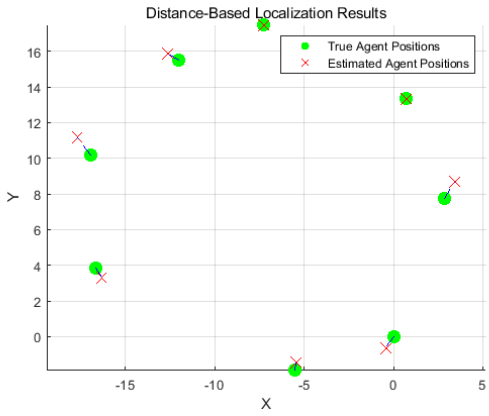
(a) Noiseless case, RMSE = 0



(b) $\sigma = 0.1$, RMSE = 0.2062



(c) $\sigma = 0.2$, RMSE = 0.3890



(d) $\sigma = 0.5$, RMSE = 0.6190

Figure 3.13: Self-localization results under different noise levels: (a) noiseless case, (b) $\sigma^2 = 0.1$, (c) $\sigma^2 = 0.2$, and (d) $\sigma^2 = 0.5$. The figures show estimated agent positions versus ground truth.

Figure 3.13 illustrates the spatial self-localization results of the UAV agents under different levels of distance measurement noise. In the noiseless case (Figure 3.13a), the algorithm achieves perfect reconstruction of the true positions, resulting in zero RMSE. This confirms the correctness of the underlying optimization model and validates the consistency of the algorithm when ideal measurements are available.

As the measurement noise increases, the localization accuracy gradually degrades. When $\sigma = 0.1$ (Figure 3.13b), the agents can still approximate their true positions with

relatively small errors (RMSE = 0.2741), indicating that the algorithm maintains good robustness under mild noise. However, with $\sigma = 0.2$ (Figure 3.13c), estimation errors become visibly larger and more unevenly distributed across agents, especially those located at the periphery of the communication network. In the high-noise scenario ($\sigma = 0.5$; Figure 3.13d), some agents show significant deviation from their true positions, leading to an overall RMSE of 0.6190.

These results suggest that the performance of the neighbor-based self-localization algorithm is sensitive to the accuracy of pairwise distance measurements. Nevertheless, even in moderately noisy environments, the algorithm is able to maintain a sub-meter average error in estimating each agent's own position. This level of accuracy provides a reliable basis for subsequent tasks such as cooperative target tracking and formation control, where knowing the agent's own position is essential.

3.3 Conclusion

This chapter reviews and compares existing localization and tracking methods and distills key insights for distributed settings. For tracking, we assess EKF/UKF versus GP/KISS-GP: under strong nonlinearity and uncertainty, GP-based approaches offer superior accuracy and uncertainty quantification, while KISS-GP achieves large-scale scalability via structured kernel interpolation. In distributed learning, we summarize the strengths and limitations of expert-fusion models (PoE, gPoE, BCM) and present an ADMM-based scalable training framework that reduces the communication cost of strict per-step gradient consensus.

Through simulations with two UAV deployment topologies (uniform and random) and varying communication and measurement radii, we systematically analyze the relationships among coverage, neighbor count, and estimation error. The results show that reasonably increasing the measurement radius substantially boosts the number of observable samples and the stability of information fusion, but with diminishing returns and rising cost.

For cooperative localization, we construct a neighbor–distance–based objective and update only non-anchor nodes to align to a global frame. Simulations indicate that enlarging the communication radius significantly improves coverage and the number of observable samples, thereby reducing localization error. The average localization error decreases as the number of neighbors increases, and higher noise levels require stronger consensus and priors to maintain stability.

Distributed Gaussian Process with Multi-agents Localization and Tracking

4

In the previous chapter, we examined two fundamental components in distributed multi-agent systems: *Cooperative Localization* (CL) and *Distributed Tracking* (DT). CL enables agents to estimate their own positions in GNSS-denied environments through local ranging and neighbor-to-neighbor communication, while DT allows agents to estimate the dynamic states of noncooperative targets based on local observations and peer-to-peer information fusion. Although these modules are architecturally independent, they are closely interrelated in practice and together form the foundation of spatial awareness in distributed systems.

Consider a typical *Search and Rescue (SAR)* mission at sea. A drifting object, such as a life raft, may become entrained in a mesoscale anticyclonic eddy and exhibit slow, nonlinear, quasi-circular motion. In such environments—characterized by limited GNSS coverage, complex sea-state dynamics, and slow-moving targets—conventional tracking methods often prove inadequate. To improve mission reliability, multiple unmanned aerial vehicles (UAVs) are deployed from a coastal command center to continuously monitor the target from different vantage points. However, the effectiveness of these observations critically depends on each UAV's ability to accurately localize itself. Target observations without spatial context are meaningless; conversely, localization errors can propagate through the tracking pipeline, leading to degraded or even divergent target state estimates.

Moreover, in such dynamic environments, the target's motion can theoretically provide reverse constraints to aid in relative localization among agents. This indicates an inherent coupling between localization and tracking. Treating them as fully decoupled modules often fails to exploit their synergistic potential. A unified estimation framework is therefore required—one that allows both tasks to be jointly addressed under a shared reasoning process, reducing redundancy and improving overall system performance.

To establish a performance baseline, this chapter first introduces a widely used and relatively advanced method: the *Distributed Invariant Kalman Filter (DIKF)*. This approach jointly models agent and target states and uses Kalman filtering for joint estimation. However, its reliance on linear system assumptions makes it prone to performance degradation in highly nonlinear scenarios.

To address this limitation, we propose an *Integrated Cooperative Localization and Distributed Tracking (ICLDT)* framework. Unlike traditional sequential approaches, ICLDT employs a first-order Taylor expansion to linearly approximate the localization error terms, thereby formulating a unified optimization problem that simultaneously solves for both agent positions and target states. It is worth emphasizing that one key difference is that the traditional method requires a known model, whereas ICLDT is non-parametric, making it more flexible in scenarios with unknown or complex dynam-

ics. All computations are performed locally, and communication is limited to neighboring agents, ensuring full decentralization and scalability.

The remainder of this chapter is organized as follows:

- **Baseline Comparison:** We present a joint Kalman filter approach as a benchmark for evaluating the proposed method;
- **Sequential Optimization(SCLDT):** A decoupled strategy that performs localization followed by tracking, along with an analysis of its limitations;
- **Integrated Optimization(ICLDT):** A integrated optimization framework using linear approximation to enable unified, distributed estimation;
- **Discussion:** We summarize the advantages and applicable scenarios of the proposed approach and discuss its potential extensions to other cooperative perception tasks.

This integrated framework not only improves robustness and estimation accuracy in GNSS-denied environments but also provides a unified and scalable solution for cooperative perception in dynamic and uncertain multi-agent systems.

4.1 Problem Formulation

We consider a decentralized multi-agent system consisting of N agents operating in a two-dimensional environment. A subset of agents, denoted by $\mathcal{A} \subset \{1, 2, \ldots, N\}$, are anchors with known static positions. The remaining agents, denoted by $\mathcal{M} = \{1, 2, \ldots, N\} \setminus \mathcal{A}$, must estimate their own positions collaboratively. In addition, the network is tasked with tracking the dynamic state of a noncooperative target that is moving under unknown control and environmental forces.

Let $\mathbf{p}_i \in \mathbb{R}^2$ denote the true position of agent i and $\hat{\mathbf{p}}_i$ its estimate.

Each agent is equipped with onboard sensors (e.g., UWB, IMU, vision) that allow it to:

- measure noisy relative distances to neighboring agents within a communication radius R_c;
- observe noisy measurements of the target state, either in absolute or relative coordinates, depending on sensor modality and spatial visibility.

The agent network forms a undirected graph, where nodes represent agents and edges denote communication links based on current relative distances. Two agents i and j are considered neighbors if:

$$\mathcal{N}_i = \{j \in \{1, \ldots, N\} \setminus \{i\} : \|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j\| \leq R_c\}. \tag{4.1}$$

Let d_{ij} denote the distance measurement between agent i and agent j:

$$d_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\| + \varepsilon_{ij}, \quad \varepsilon_{ij} \sim \mathcal{N}(0, \sigma^2),$$

and let y_{ik} denote the observation of the target made by agent i at time k:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k,$$

where, $\mathbf{H} = \mathbf{I}_{3\times 3}$, $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R})$.
The overall objective is to jointly estimate:

- the unknown agent positions $\{\hat{\mathbf{p}}_i\}_{i\in\mathcal{N}}$, and
- the sequence of target states $\{\hat{\mathbf{x}}_k\}_{k=1}^{T}$,

by fusing all available measurements in a distributed and online manner.

This formulation integrates both cooperative-localization and target tracking into a unified estimation problem. In the following sections, we introduce the joint probabilistic model and describe how distributed inference is performed using message-passing and local optimization techniques.

4.2 Objective Function Formulation

To formally describe the joint estimation problem of cooperative-localization and target tracking, we introduce two coupled objective functions: J_1 and J_2, corresponding to the cooperative-localization and distributed tracking tasks, respectively.

Cooperative-localization Objective (J_1): In the absence of external positioning infrastructure, non-anchor agents estimate their positions by measuring inter-agent distances within their communication neighborhood. This process can be formulated as the following nonlinear least-squares optimization:

$$J_1(\mathbf{p}) = \sum_{i\in\mathcal{M}} \sum_{j\in\mathcal{N}_i} \left(\|\mathbf{p}_i - \mathbf{p}_j\| - d_{ij}\right)^2, \tag{4.2}$$

where $\mathbf{p}_i$ denotes the estimated position of agent i, d_{ij} is the noisy distance measurement between agents i and j, and $\mathcal{N}_i$ is the set of neighbors of agent i. This cost encourages consistency between estimated positions and observed distances and serves as the core constraint for cooperative localization.

Target Tracking Objective (J_2): For distributed target tracking, each agent maintains a local Gaussian Process (GP) model that predicts the state of the target. Based on the GP prediction, each agent constructs a local negative log-likelihood (NLL) loss function defined as:

$$\ell^{(i)}(\boldsymbol{\theta}) = (\mathbf{z}_i + \mathbf{p}_i)^\top \Sigma^{(i)}(\boldsymbol{\theta})^{-1}(\mathbf{z}_i + \mathbf{p}_i) + \log\left|\Sigma^{(i)}(\boldsymbol{\theta})\right|, \tag{4.3}$$

where $\mathbf{z}_i$ is the relative observation of the target made by agent i, $\mathbf{p}_i$ is the agent's current position estimate, and $\Sigma^{(i)}(\boldsymbol{\theta})$ is the predictive covariance matrix from the GP model. This objective captures both the data fit and uncertainty associated with target state estimation.

The global tracking objective is then formulated as the sum of local losses:

$$J_2(\mathbf{p}_i, \boldsymbol{\theta}) = \sum_{i=1}^{N} \ell^{(i)}(\boldsymbol{\theta}). \tag{4.4}$$

Coupling Between J_1 and J_2: These two objectives are inherently coupled through the agent position $\mathbf{p}_i$, which appears both in the localization term and in the tracking prediction. Accurate target tracking requires precise agent localization, while temporally coherent target dynamics can, in turn, serve as a soft constraint to aid cooperative-localization. Therefore, the complete estimation problem must jointly optimize J_1 and J_2 under a unified inference framework, which we introduce in the subsequent sections.

4.3 Benchmark

4.3.1 Method

The Joint Kalman Filter (Joint-KF) is a classical method for joint state estimation, where both agent self-localization and target tracking states are integrated into an augmented state vector and estimated through standard linear-Gaussian Kalman filtering. Owing to its simplicity and computational efficiency, Joint-KF has been widely used as a benchmark in cooperative localization and target tracking tasks [44, 45].

Many recent studies have built upon the Joint-KF framework to address limitations in specific applications. For instance, Zhao et al. [46] introduced a robust joint filter based on the maximum correntropy criterion to cope with non-Gaussian noise. Tang et al. [47] developed a federated cubature Kalman filter to fuse IMU and UWB data in multi-sensor systems. Zhou et al. [48] proposed a distributed invariant EKF on Lie groups to enhance consistency under nonlinear dynamics, while Tzoumas et al. [49] formulated a covariance-intersection-based invariant filter for decentralized fusion in multi-agent networks.

Despite these efforts, most of these methods still inherit limitations from the original Joint-KF, such as reliance on linear approximations, structural symmetry, or centralized priors. In highly nonlinear, partially observed, and communication-constrained multi-agent settings—as considered in this work—these assumptions often fail. This motivates the development of a more robust and flexible joint optimization framework to improve both localization and tracking performance under realistic distributed constraints.

The general idea behind such methods is as follows: each agent maintains a local joint state vector that includes both the dynamic state of the target and its own (unknown) position:

$$\mathbf{x}^{(i)} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \\ p_{i,x} \\ p_{i,y} \end{bmatrix} \in \mathbb{R}^5,$$

where (x_t, y_t, θ_t) represents the target's position and orientation, and $(p_{i,x}, p_{i,y})$ denotes the static position of agent i.

The Joint-KF approach applies the standard Kalman filtering procedure. At each time step, the state prediction and update are given by:

$$\begin{aligned}
\text{Prediction:} \quad & \hat{\mathbf{x}}^{(i)}_{k|k-1} = \mathbf{F}\hat{\mathbf{x}}^{(i)}_{k-1|k-1} \\
& \mathbf{P}^{(i)}_{k|k-1} = \mathbf{F}\mathbf{P}^{(i)}_{k-1|k-1}\mathbf{F}^\top + \mathbf{Q} \\
\text{Update:} \quad & \mathbf{K}^{(i)}_k = \mathbf{P}^{(i)}_{k|k-1}\mathbf{H}^\top \left(\mathbf{H}\mathbf{P}^{(i)}_{k|k-1}\mathbf{H}^\top + \mathbf{R}\right)^{-1} \\
& \hat{\mathbf{x}}^{(i)}_{k|k} = \hat{\mathbf{x}}^{(i)}_{k|k-1} + \mathbf{K}^{(i)}_k \left(\mathbf{z}^{(i)}_k - \mathbf{H}\hat{\mathbf{x}}^{(i)}_{k|k-1}\right) \\
& \mathbf{P}^{(i)}_{k|k} = \left(\mathbf{I} - \mathbf{K}^{(i)}_k\mathbf{H}\right)\mathbf{P}^{(i)}_{k|k-1}
\end{aligned}$$

Here, $\mathbf{F}$ and $\mathbf{H}$ denote the state transition and observation matrices. For a five-dimensional state vector, a typical choice is

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & -v\sin\theta_t\Delta t & 0 & 0 \\ 0 & 1 & v\cos\theta_t\Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

assuming the target follows a simple motion model while the agent's own position is static.

If the observation corresponds to relative position measurements $\mathbf{z}^{(i)}_k = \begin{bmatrix} x_t - p_{i,x} \\ y_t - p_{i,y} \end{bmatrix} + \mathbf{r}_k$, then the observation matrix is

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 \end{bmatrix}.$$

The process noise covariance and observation noise covariance are typically chosen as diagonal matrices:

$$\mathbf{Q} = \mathrm{diag}(\sigma_x^2, \sigma_y^2, \sigma_\theta^2, \epsilon, \epsilon), \quad \mathbf{R} = \mathrm{diag}(\sigma_{z_x}^2, \sigma_{z_y}^2),$$

where ϵ is set to a small value to account for static agent positions.

Kalman Filter Equations. Each agent independently runs a standard Kalman Filter:

$$\begin{aligned}
\text{Prediction:} \quad & \hat{\mathbf{x}}^{(i)}_{k|k-1} = F_k\hat{\mathbf{x}}^{(i)}_{k-1|k-1}, \\
& P^{(i)}_{k|k-1} = F_k P^{(i)}_{k-1|k-1}F_k^\top + Q. \\
\text{Update:} \quad & K^{(i)}_k = P^{(i)}_{k|k-1}H^\top \left(HP^{(i)}_{k|k-1}H^\top + R\right)^{-1}, \\
& \hat{\mathbf{x}}^{(i)}_{k|k} = \hat{\mathbf{x}}^{(i)}_{k|k-1} + K^{(i)}_k \left(\mathbf{z}^{(i)}_k - H\hat{\mathbf{x}}^{(i)}_{k|k-1}\right), \\
& P^{(i)}_{k|k} = \left(I - K^{(i)}_k H\right)P^{(i)}_{k|k-1}.
\end{aligned}$$

4.3.2 Results

It is worth noting that the experimental setup (including the scenario, dynamic model, measurement model, and noise settings) for the Joint-KF baseline is kept identical to that used in the previous experiments to ensure a fair comparison.

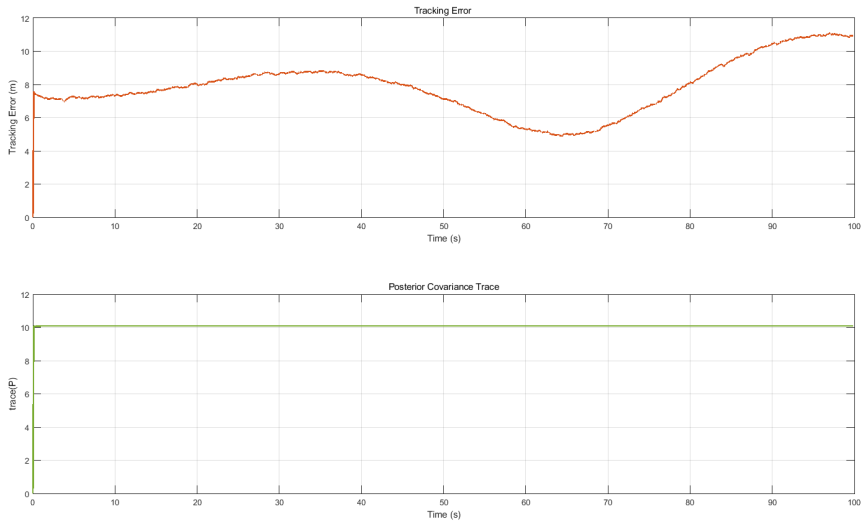


Figure 4.1: Tracking error and posterior covariance trace of the Joint-KF baseline under nonlinear target dynamics.

As shown in Fig.4.1, the Joint Kalman Filter exhibits poor performance in this scenario. The tracking error remains consistently high throughout the episode, starting around 7 meters and rising to nearly 10 meters by the end, with significant fluctuations. This indicates that the filter fails to accurately estimate the target state over time. Meanwhile, the posterior covariance trace stays nearly constant at a value close to 10, showing no meaningful reduction in uncertainty as more measurements become available. Ideally, a well-functioning estimator should demonstrate a decreasing trend in the posterior covariance trace, reflecting improved confidence through information accumulation. The lack of covariance contraction suggests that the filter is not effectively incorporating observation data. This failure may be attributed to factors such as model linearization inaccuracies, insufficient responsiveness to measurement residuals, or overly optimistic uncertainty modeling. Together, these issues reveal the limitations of the Joint-KF framework under challenging tracking conditions, particularly when the system involves nonlinear dynamics and noisy or ambiguous observations.

4.4 Sequential Cooperative Localization and Distributed Tracking(SCLDT)

4.4.1 Framework

The joint estimation of agent positions and target state forms a coupled multi-objective optimization problem composed of two components:

- The **localization loss**:

$$J_1(\mathbf{p}) = \sum_{i \in \mathcal{M}} \sum_{j \in \mathcal{N}_i} \left(\|\mathbf{p}_i - \mathbf{p}_j\| - d_{ij} \right)^2 \tag{4.5}$$

- The **tracking loss**:

$$J_2(\mathbf{p}_i, \boldsymbol{\theta}) = \sum_{i=1}^{N} \left[(\mathbf{z}_i + \mathbf{p}_i)^\top \Sigma^{(i)}(\boldsymbol{\theta})^{-1} (\mathbf{z}_i + \mathbf{p}_i) + \log |\Sigma^{(i)}(\boldsymbol{\theta})| \right] \tag{4.6}$$

In this formulation, the target tracking loss J_2 is explicitly dependent on the agent position estimates $\mathbf{p}_k$, which also serve as the optimization variables in J_1. To address this coupling in a tractable and scalable manner, we adopt a *sequential optimization* approach that decomposes the problem into two stages:

Stage I – Agent localization: We begin by minimizing $J_1(\mathbf{p})$ independently, using only inter-agent distance measurements. This optimization can be carried out in a distributed fashion by each non-anchor agent using local ranging information and iterative update schemes. The result is a globally consistent set of agent positions $\hat{\mathbf{p}}$.

Stage II – Target tracking: With the agent positions fixed to $\mathbf{p}_i = \hat{\mathbf{p}}_i$, we then solve the tracking problem by minimizing $J_2(\mathbf{p}_i, \boldsymbol{\theta})$ with respect to the target state $\boldsymbol{\theta}$. Each agent applies its Gaussian Process model and computes a local negative log-likelihood (NLL) loss based on its own relative measurement and current position. A consensus-based optimization method, ADMM, is used to reach agreement on the global target state.

This sequential scheme avoids the need to jointly solve a large non-convex problem, while still capturing the dependency structure between localization and tracking. It also aligns well with the practical constraints of real-world multi-agent systems, where localization typically precedes high-level inference tasks.

Figure 4.2 illustrates the complete flow of this sequential optimization strategy.

4.4.2 Simulation

To better reflect realistic conditions encountered in maritime search-and-rescue (SAR) operations, we begin by evaluating the proposed sequential optimization framework in a structured deployment scenario where eight UAV agents are evenly spaced around the target drift region. This circular configuration ensures that the entire area of interest

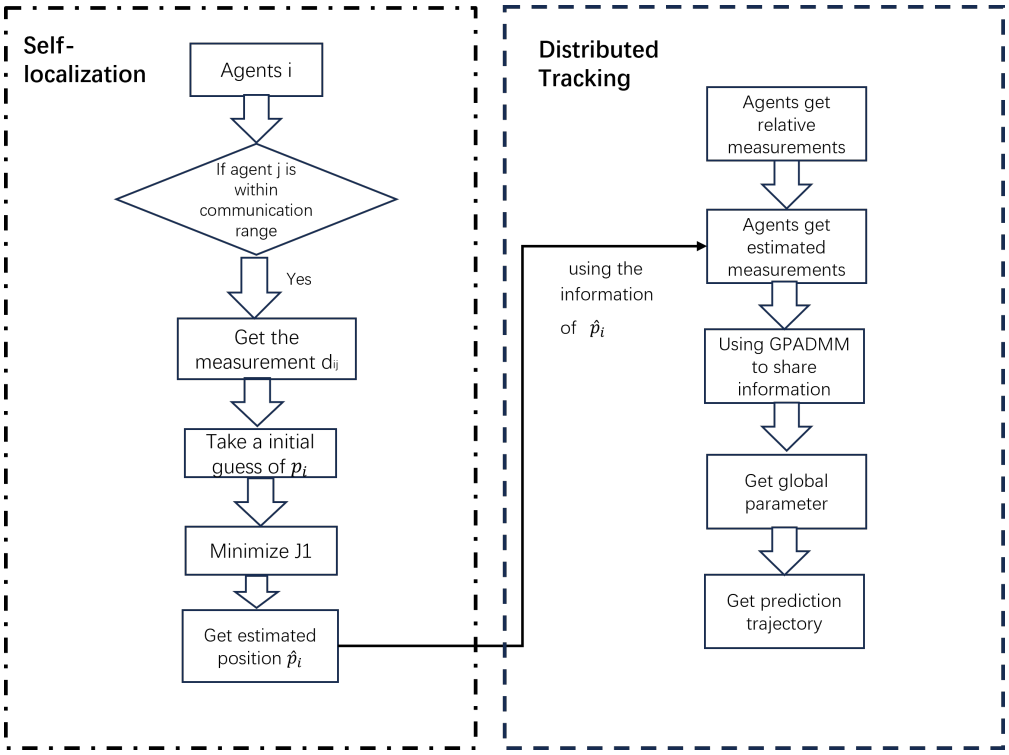


Figure 4.2: Sequential optimization flow: the left branch shows the localization stage based on inter-agent ranging; the right branch shows the tracking stage using fixed localization and GP-based prediction with consensus.

remains well covered, allowing agents to maintain consistent line-of-sight measurements and reliable communication with their neighbors.

Such a uniform deployment not only simplifies the analysis of algorithmic behavior under controlled conditions, but also emulates real-world UAV formations frequently used in coordinated SAR missions or environmental monitoring. The geometric symmetry of the layout helps maintain balanced observation density across the region, while minimizing spatial bias in data collection.

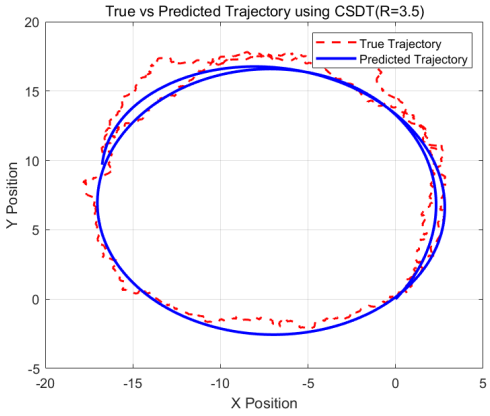
Moreover, this setting naturally decouples the two key subproblems in our framework: cooperative-localization and distributed target tracking. Since each UAV is able to receive relatively accurate distance measurements from multiple neighbors, the position estimates obtained from the cooperative-localization phase (J_1) are sufficiently reliable to serve as input for the subsequent GP-based tracking phase (J_2). This sequential design allows us to examine the interaction between localization accuracy and tracking performance in a clear and interpretable manner.

4.4.3 Performance Analysis

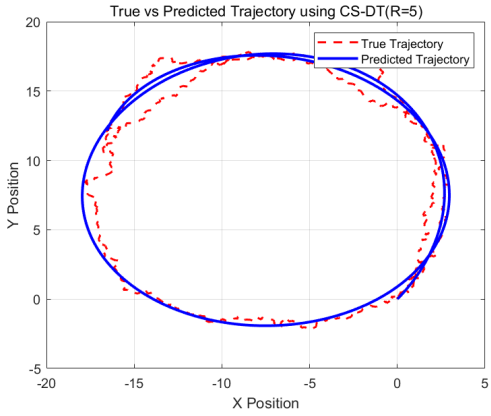
Figure 4.6 illustrates how the performance of the proposed CS-DT framework varies with respect to the communication radius R. The upper plot shows the RMSE of the estimated target trajectory in x, y, and θ components, as well as the overall RMSE.

Table 4.1: Agent Localization Results (Average localization error: 0.2062 m)

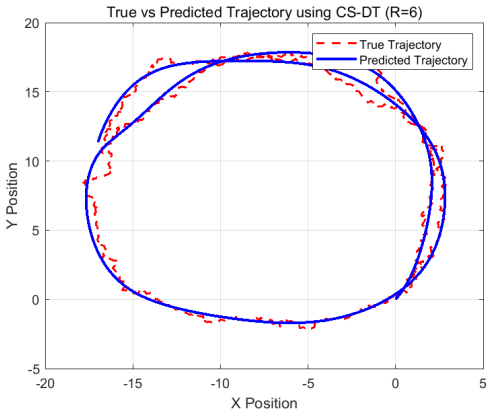
Agent	Actual Position (x, y)	Estimated Position (x, y)	Error	Remark
1	(0.0000, 0.0000)	(-0.1734, -0.2791)	0.3273	
2	(0.6760, 13.3549)	(0.6760, 13.3549)	0.0000	Known position
3	(-12.0280, 15.4859)	(-12.3165, 15.6705)	0.3275	
4	(-16.6466, 3.8581)	(-16.5002, 3.6168)	0.2785	
5	(-5.5003, -1.8768)	(-5.4741, -1.6776)	0.2004	
6	(2.8364, 7.7778)	(3.1226, 8.1952)	0.5075	
7	(-7.2575, 17.4908)	(-7.2575, 17.4908)	0.0000	Known position
8	(-16.9486, 10.2055)	(-17.2868, 10.6170)	0.6396	



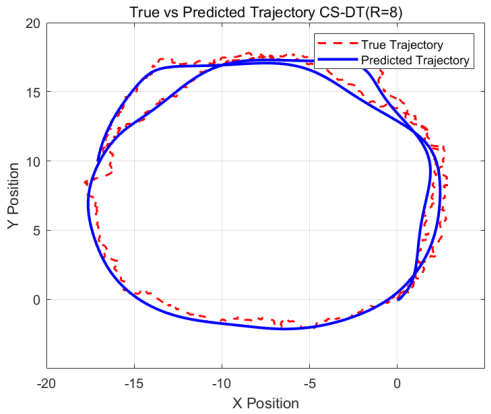
(a) SCLDT (R = 3.5)



(b) SCLDT (R = 5)



(c) SCLDT (R = 6)



(d) SCLDT (R = 8)

Figure 4.3: Trajectory tracking results under different communication radii R using the SCLDT framework. Subfigures (a)–(d) show the predicted vs. true trajectories when $R = 3.5$, 5, 6, and 8, respectively.

Algorithm 3 Sequential Distributed Optimization for Localization and Tracking

Require: Anchor positions $\{\mathbf{p}_a\}_{a\in\mathcal{A}}$, initial GP parameters $\{\boldsymbol{\theta}_i^0\}$, communication radius r, max iterations T

Ensure: Predicted target trajectory $\hat{\mathbf{s}}(t)$

Step 1: Iterative cooperative-Localization
Initialize agent positions $\hat{\mathbf{p}}_i$ with random offsets for $i \in \mathcal{F}$
for $t \leftarrow 1$ to T **do**
  for all $i \in \mathcal{F}$ **do**
    Retrieve neighbor positions $\{\hat{\mathbf{p}}_j\}$ for $j \in \mathcal{N}_i$
    Measure relative distance $d_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\| + \mathcal{N}(0, \sigma^2)$
    Update $\hat{\mathbf{p}}_i \leftarrow \arg\min_{\mathbf{p}} J(\mathbf{p}_i; \{d_{ij}\}, \{\hat{\mathbf{p}}_j\})$
  end for
  if converged **then**
    break
  end if
end for
Step 2: Distributed Target Tracking via ADMM-GP
Initialize $r \leftarrow 0$, GP priors $\boldsymbol{\theta}_i^0$, dual variables ζ_i^0, global variable z^0, penalty ρ
while $\|\Delta_p^r\|^2 \geq \epsilon_{\text{pri}} \ \vee \ \|\Delta_d^r\|^2 \geq \epsilon_{\text{dual}}$ **do**
  $r \leftarrow r + 1$
  for $i \leftarrow 1$ to N **do**
    Update GP parameters $\boldsymbol{\theta}_i^r$ using local data and $\hat{\mathbf{p}}_i$
  end for
  Update global parameter z^r via consensus
  Update dual variables ζ_i^r
  Compute residuals Δ_p^r, Δ_d^r
  Adjust tolerances if needed
end while
return $\hat{\mathbf{s}}(t)$

Table 4.2: Tracking performance under varying communication radius R using the SCLDT framework

R	RMSE_X	RMSE_Y	RMSE_θ	Overall RMSE	Sample Size
3.50	0.931	0.946	0.038	0.767	913
4.00	1.120	1.103	0.029	0.907	1051
4.50	**0.805**	**0.816**	**0.011**	**0.661**	**1178**
5.00	1.187	1.059	0.022	0.919	1327
5.50	0.816	0.818	0.010	0.667	1462
6.00	0.439	0.359	0.018	0.328	1596
6.50	0.386	0.328	0.012	0.293	1737
7.00	0.594	0.648	0.010	0.507	1886

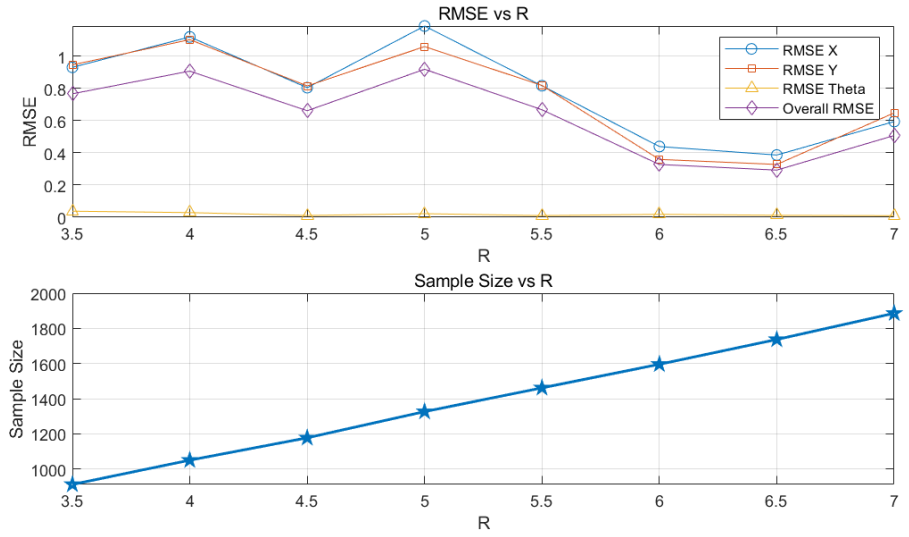


Figure 4.4: Top: RMSE of different state components (X, Y, θ) and overall RMSE under varying communication radius R in the CS-DT framework. Bottom: Corresponding number of valid samples observed by the UAV network.

The lower plot shows the corresponding number of valid samples used for training and inference.

We observe that as the communication radius increases, the number of valid measurements collected by the UAV network grows steadily. This is expected, as a larger R allows each UAV to access more neighbors and accumulate more relative measurements. However, the relationship between communication range and RMSE is not strictly monotonic.

The lowest overall RMSE is achieved around $R = 6.5$, where the system benefits from both sufficient sample density and high measurement diversity. In contrast, both too small (e.g., $R = 3.5$) and too large ($R = 7$) communication ranges yield higher RMSE values. This indicates a trade-off: while increasing R improves measurement coverage, excessive overlap may introduce redundant or less informative data, potentially leading to over-smoothing in the consensus process or increased noise accumulation.

Furthermore, the estimation of the angular component θ remains consistently accurate across all R values, demonstrating the robustness of the underlying GP model to orientation dynamics. These results highlight the importance of choosing an appropriate communication radius to balance accuracy, efficiency, and robustness in real-world UAV tracking networks.

4.5 Intergrated Cooperative Localization and Distributed Tracking(ICLDT)

4.5.1 Framework

While the sequential optimization strategy discussed earlier effectively decouples localization and tracking into two manageable subproblems, it introduces a fundamental limitation: the accuracy of the target state estimation (J_2) strongly depends on the prior agent position estimates $\mathbf{p}_i$, which are optimized without accounting for the subsequent tracking task. This separation may lead to error propagation and underutilization of the rich interdependence between agent positioning and target inference.

To overcome this limitation, we propose a **integrated optimization framework** that treats cooperative-localization and target tracking as a unified problem. The key idea is to optimize both components simultaneously, allowing bidirectional information flow and collaborative refinement during inference. To this end, we introduce a weighted multi-objective cost function:

$$J = \lambda J_1(\mathbf{p}) + (1 - \lambda) J_2(\mathbf{p}, \boldsymbol{\theta}), \tag{4.7}$$

where J_1 denotes the localization cost based on inter-agent ranging, and J_2 represents the negative log-likelihood of the target state prediction based on Gaussian Process regression. The scalar weight $\lambda \in [0, 1]$ balances the contribution of the two components.

This formulation embodies the principle of *tracking-aware localization and localization-enhanced tracking*, promoting consistency between agent position estimation and target inference. It enables the system to achieve improved accuracy and robustness by integrately considering the spatial structure of the network and the temporal evolution of the target, which is particularly important in distributed settings with limited observations or noisy measurements.

4.5.2 Convexity Analysis of the Objective Function

In order to understand the challenges associated with solving the joint optimization problem, we analyze the convexity of the two constituent objective functions J_1 and J_2 defined in Equation (4.7).

Localization objective J_1: The localization cost function $J_1(\mathbf{p})$ is defined based on the squared error between estimated inter-agent distances and observed measurements:

$$J_1(\mathbf{p}) = \sum_{(i,j)\in\mathcal{E}} \left(\|\mathbf{p}_i - \mathbf{p}_j\| - d_{ij} \right)^2, \tag{4.8}$$

where $\mathcal{E}$ denotes the set of agent pairs within communication range, and d_{ij} is the noisy measured distance between agents i and j.

This function is **non-convex** due to the presence of the Euclidean norm $\|\mathbf{p}_i - \mathbf{p}_j\|$, which is a non-linear operation. Even though the squared distance is convex,

the subtraction of the observed measurement d_{ij} followed by squaring introduces non-convexity in the global parameter space. The optimization landscape contains multiple local minima, especially when the initial estimates are far from the ground truth.

Tracking objective J_2: The tracking loss $J_2(\mathbf{p}, \boldsymbol{\theta})$ is defined as the sum of lNLL functions based on GP regression models:

$$J_2(\mathbf{p}, \boldsymbol{\theta}) = \sum_{k=1}^{N} \left[(\mathbf{z}_i + \mathbf{p}_i)^\top \Sigma^{(i)}(\boldsymbol{\theta})^{-1} (\mathbf{z}_i + \mathbf{p}_i) + \log |\Sigma^{(i)}(\boldsymbol{\theta})| \right]. \tag{4.9}$$

Here, the dependency of J_2 on both $\mathbf{p}_i$ and $\boldsymbol{\theta}$ introduces further **non-convexity**. Specifically:

- The GP covariance matrix $\Sigma^{(i)}(\boldsymbol{\theta})$ depends nonlinearly on the target state via the kernel function.
- The inversion and determinant operations inside the NLL make the objective analytically intractable and non-convex in general.
- The term $(\mathbf{z}_i + \mathbf{p}_i)$ couples the measurement and localization error, resulting in non-separable terms.

Due to these characteristics, the joint cost function J is inherently non-convex and challenging to minimize globally. As such, optimization methods must rely on good initialization and iterative local updates. In the next section, we discuss a convex relaxation of the localization objective

4.5.3 Convex Relaxation of the Localization Objective

To address the non-convexity of the localization term J_1, we propose a first-order Taylor-based relaxation that approximates the objective as a convex quadratic function, thereby enabling closed-form updates.

Recall the original form of J_1:

$$J_1(\mathbf{p}) = \sum_{(i,j)\in\mathcal{E}} \left(\|\mathbf{p}_i - \mathbf{p}_j\| - d_{ij} \right)^2, \tag{4.10}$$

where d_{ij} is the measured (noisy) distance between agents i and j. This function is non-convex due to the presence of the norm operator. To make the problem tractable, we linearize the distance term around the current position estimate $\mathbf{p}_j^{(0)}$ using a first-order Taylor expansion:

$$\|\mathbf{p}_i - \mathbf{p}_j\| \approx r_{ij}^{(0)} + \nabla r_{ij}^{(0)} \cdot (\mathbf{p}_i - \mathbf{p}_i^{(0)}), \tag{4.11}$$

where: $r_{ij}^{(0)} = \|\mathbf{p}_i^{(0)} - \mathbf{p}_j^{(0)}\|$, $\nabla r_{ij}^{(0)} = \frac{(\mathbf{p}_i^{(0)} - \mathbf{p}_j^{(0)})^\top}{\|\mathbf{p}_i^{(0)} - \mathbf{p}_j^{(0)}\|} = \mathbf{a}_{ij}^\top$.

Substituting this into J_1, we obtain a relaxed form:

$$\tilde{J}_1(\mathbf{p}) = \sum_{j \in \mathcal{N}_i} \left(-\mathbf{a}_{ij}^\top \mathbf{p}_i + b_{ij}\right)^2, \tag{4.12}$$

where $b_{ij} = r_{ij}^{(0)} - d_{ij} - \mathbf{a}_{ij}^\top \mathbf{p}_i^{(0)}$. This yields a standard convex quadratic form.

Closed-form solution: The above relaxed problem can be solved analytically using least-squares. Defining:

$$\mathbf{A}_i = \begin{bmatrix} \mathbf{a}_{i1}^\top \\ \mathbf{a}_{i2}^\top \\ \vdots \\ \mathbf{a}_{ij}^\top \end{bmatrix}, \quad \mathbf{b}_i = \begin{bmatrix} b_{i1} \\ b_{i2} \\ \vdots \\ b_{ij} \end{bmatrix}, \tag{4.13}$$

the optimal position update for agent i becomes:

$$\hat{\mathbf{p}}_i = (\mathbf{A}_i^\top \mathbf{A}_i)^{-1} \mathbf{A}_i^\top \mathbf{b}_i. \tag{4.14}$$

This closed-form update can be efficiently computed in each iteration and integrated into the joint optimization loop, significantly improving convergence behavior while maintaining a decentralized structure.

4.5.4 Distributed Intergrated Optimization via ADMM

To solve the coupled non-convex joint objective in a distributed manner, we adopt the Alternating Direction Method of Multipliers (ADMM). Specifically, we focus on the following joint optimization problem over agent states θ_i and positions $\mathbf{p}_i$:

$$\min_{\{\boldsymbol{\theta}_i, \mathbf{p}_i\}} \sum_{i=1}^{N} J^{(i)}(\boldsymbol{\theta}_i, \mathbf{p}_i), \tag{4.15}$$

subject to consensus constraints:

$$\boldsymbol{\theta}_i = \mathbf{z}, \quad \forall i = 1, \ldots, N, \tag{4.16}$$

where $\mathbf{z}$ is the global consensus variable representing the estimated target state.
We define the augmented Lagrangian:

$$\mathcal{L}(\{\boldsymbol{\theta}_i\}, \{\mathbf{p}_i\}, \{\boldsymbol{\zeta}_i\}, \mathbf{z}) = \sum_{i=1}^{N} \left[J^{(i)}(\boldsymbol{\theta}_i, \mathbf{p}_i) + \boldsymbol{\zeta}_i^\top (\boldsymbol{\theta}_i - \mathbf{z}) + \frac{\rho}{2} \|\boldsymbol{\theta}_i - \mathbf{z}\|_2^2 \right], \tag{4.17}$$

where $\boldsymbol{\zeta}_i$ are dual variables and $\rho > 0$ is the penalty parameter.
We update the variables iteratively as follows:

1. Local target update: Each agent independently updates its local target state θ_i using its own measurement and current estimate of $\mathbf{z}$ and $\boldsymbol{\zeta}_i$:

$$\boldsymbol{\theta_i}^{r+1} = \arg\min_{\boldsymbol{\theta_i}} \left(J^{(i)}(\boldsymbol{\theta_i}, \mathbf{p}_i^r) + \boldsymbol{\zeta}_i^{r\top} (\boldsymbol{\theta_i} - \mathbf{z}^r) + \frac{\rho}{2} \|\boldsymbol{\theta_i} - \mathbf{z}^r\|_2^2 \right). \tag{4.18}$$

2. Global consensus update: The shared variable $\mathbf{z}$ is updated by aggregating all local estimates:

$$\boldsymbol{z}^{r+1} = \frac{1}{N} \sum_{i=1}^{N} \left(\boldsymbol{\theta}_{\boldsymbol{i}}^{\boldsymbol{r+1}} + \frac{1}{\rho} \boldsymbol{\zeta}_i^r \right). \tag{4.19}$$

3. Dual variable update:

$$\boldsymbol{\zeta}_i^{r+1} = \boldsymbol{\zeta}_i^r + \rho \left(\boldsymbol{\theta}_{\boldsymbol{i}}{}^{r+1} - \boldsymbol{z}^{r+1} \right). \tag{4.20}$$

Output: After convergence, the algorithm returns optimal estimates of agent positions $\{\mathbf{p}_i\}$ and target state $\mathbf{z}$

This ADMM framework enables fully distributed, iterative updates with minimal coordination overhead, making it highly suitable for decentralized UAV tracking systems under partial observability.

Algorithm 4 Integrated Cooperative Localization and Distributed Tracking via ADMM

Require: Anchors $\{\mathbf{p}_a\}_{a\in\mathcal{A}}$, neighbor ranges $\{d_{ij}\}$, measurements $\{\boldsymbol{z}_i\}$, initial $\{\hat{\mathbf{p}}_i^0\}_{i\in\mathcal{F}}$, $\{\boldsymbol{\theta}_i^0\}$, $\{\boldsymbol{\zeta}_i^0\}$, global $\mathbf{z}^0$, penalty ρ, tolerances $\epsilon_{\text{pri}}, \epsilon_{\text{dual}}$
Ensure: Estimated positions $\{\hat{\mathbf{p}}_i\}_{i\in\mathcal{F}}$ and global parameter $\mathbf{z}^*$

$r \leftarrow 0$
repeat
  Closed-form localization ▷ parallel over $i \in \mathcal{F}$
  for all $i \in \mathcal{F}$ **do**
    Compute linearization coefficients a_{ij}, b_{ij} for $j \in \mathcal{N}_i$
    $\hat{\mathbf{p}}_i^{r+1} \leftarrow \arg\min_{\mathbf{p}} \sum_{j\in\mathcal{N}_i} \big(- a_{ij}^\top \mathbf{p} + b_{ij} \big)^2$ ▷ closed-form update of the relaxed J_1
  end for
  Local target update ▷ parallel over $i = 1, \ldots, N$
  for $i \leftarrow 1$ to N **do**
    $\boldsymbol{\theta}_i^{r+1} \leftarrow \arg\min_{\boldsymbol{\theta}_i} \Big[J^{(i)}\big(\boldsymbol{\theta}_i; \hat{\mathbf{p}}_i^{r+1}\big) + \boldsymbol{\zeta}_i^{r\top}(\boldsymbol{\theta}_i - \mathbf{z}^r) + \frac{\rho}{2} \|\boldsymbol{\theta}_i - \mathbf{z}^r\|_2^2 \Big]$
  end for
  Global consensus update
  $\mathbf{z}^{r+1} \leftarrow \frac{1}{N} \sum_{i=1}^{N} \Big(\boldsymbol{\theta}_i^{r+1} + \frac{1}{\rho} \boldsymbol{\zeta}_i^r \Big)$
  Dual variable update
  for $i \leftarrow 1$ to N **do**
    $\boldsymbol{\zeta}_i^{r+1} \leftarrow \boldsymbol{\zeta}_i^r + \rho\,(\boldsymbol{\theta}_i^{r+1} - \mathbf{z}^{r+1})$
  end for
  $r \leftarrow r + 1$
until $\|\Delta_p^r\|^2 < \epsilon_{\text{pri}} \ \wedge \ \|\Delta_d^r\|^2 < \epsilon_{\text{dual}}$
Output: $\{\hat{\mathbf{p}}_i\} \leftarrow \{\hat{\mathbf{p}}_i^r\}, \quad \mathbf{z}^* \leftarrow \mathbf{z}^r$

4.5.5 Results of ICLDT

Table 4.3 summarizes the actual positions, estimated positions (obtained via a closed-form solution), and the corresponding Euclidean errors for all agents. Among them,

Table 4.3: Comparison between actual and estimated positions for each agent ((Average localization error: 0.3831 m))

Agent	**Actual Position (x, y)**	**Estimated Position (x, y)**	**Error**	**Remark**
1	(0.0000, 0.0000)	(0.6691, 0.0217)	0.6694	
2	(0.6760, 13.3549)	(0.6760, 13.3549)	0.0000	Known position
3	(-12.0280, 15.4859)	(-12.1149, 15.0925)	0.4039	
4	(-16.6466, 3.8581)	(-16.5162, 3.6904)	0.2135	
5	(-5.5003, -1.8768)	(-4.8500, -2.1965)	0.7190	
6	(2.8364, 7.7778)	(3.1608, 7.4226)	0.4808	
7	(-7.2575, 17.4908)	(-7.2575, 17.4908)	0.0000	Known position
8	(-16.9486, 10.2055)	(-16.7840, 9.6553)	0.5734	

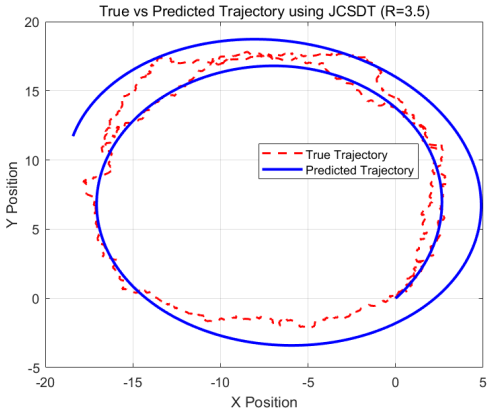
Agent 2 and Agent 7 serve as anchor nodes whose positions are known a prior and thus excluded from the estimation process. The positions of the remaining agents are computed using a closed-form method based on pairwise distance measurements and neighborhood constraints.

Overall, the localization accuracy is satisfactory. Most agents exhibit a positioning error below 0.7 meters. Notably, Agents 2 and 7 have zero error as expected. The largest error occurs at Agent 5 (approximately 0.72 m), which may result from limited anchor connectivity or noisy measurements. Agent 1 also shows a relatively higher error (around 0.67 m), indicating that nodes initialized far from anchors can be more susceptible to estimation inaccuracies.

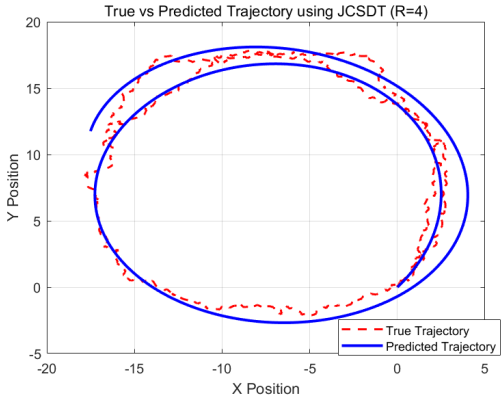
These results demonstrate that the closed-form localization approach performs effectively under partial anchor constraints. It achieves reasonable accuracy without requiring iterative computation, making it suitable for scenarios with limited computational resources or strict runtime constraints.

Table 4.4: Tracking Performance using ICLDT under Varying Measurement Radius R

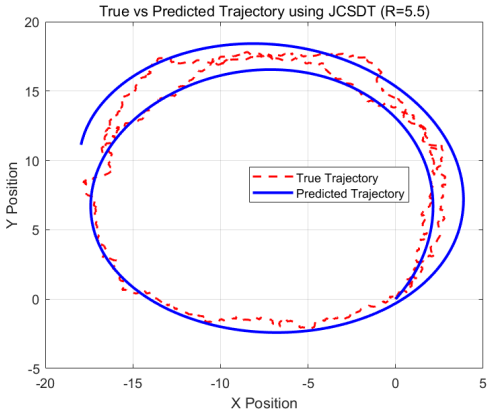
R	RMSE_X	RMSE_Y	RMSE_θ	Overall RMSE	Sample Size
3.50	1.098	1.081	0.038	0.890	913
4.00	0.975	0.950	0.029	0.786	1051
4.50	1.041	0.934	0.990	0.989	1178
5.00	1.067	0.966	0.023	0.831	1327
5.50	0.748	0.832	0.010	0.646	1462
6.00	0.808	0.940	0.019	0.716	1596
6.50	0.864	1.050	0.013	0.785	1737
7.00	0.667	0.776	0.010	0.591	1886



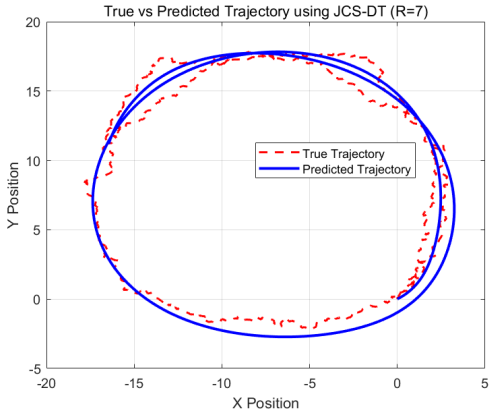
(a) ICLDT (R = 3.5)



(b) ICLDT (R = 4)



(c) ICLDT (R = 5.5)



(d) ICLDT (R = 7)

Figure 4.5: Trajectory tracking results under different communication radii R using the ICLDT. Subfigures (a)–(d) show the predicted vs. true trajectories when $R = 3.5$, 5, 6, and 8, respectively.

Table 4.4 presents the tracking performance under different communication radii R. As R increases from 3.5 to 7.0, the overall root-mean-square error (RMSE) significantly decreases from 0.89 m to 0.59 m, showing a clear monotonic improvement. This trend highlights the critical impact of communication capability on distributed tracking performance. A larger communication radius provides richer neighbor observations, which enhances global model consistency and improves target estimation accuracy.

Moreover, the trends of RMSE_x and RMSE_y show similar downward behavior, indicating that the estimator performs consistently across both spatial dimensions. The direction estimation error RMSE_θ remains relatively low throughout, suggesting that the GP+ADMM structure maintains strong stability in modeling directional dynamics.

Finally, as the communication radius increases, the sample size also grows steadily (from 913 to 1886), which not only improves statistical robustness but also demonstrates the scalability of the proposed framework. These results confirm that the joint cooperative-localization and distributed tracking algorithm maintains good per-

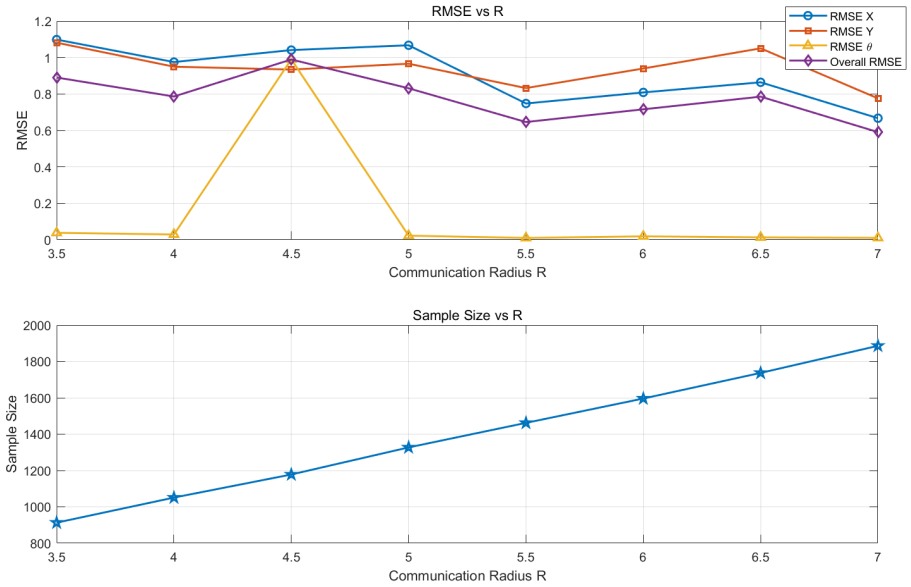


Figure 4.6: Top: RMSE of different state components (X, Y, θ) and overall RMSE under varying communication radius R in the ICLDT framework. Bottom: Corresponding number of valid samples observed by the UAV network.

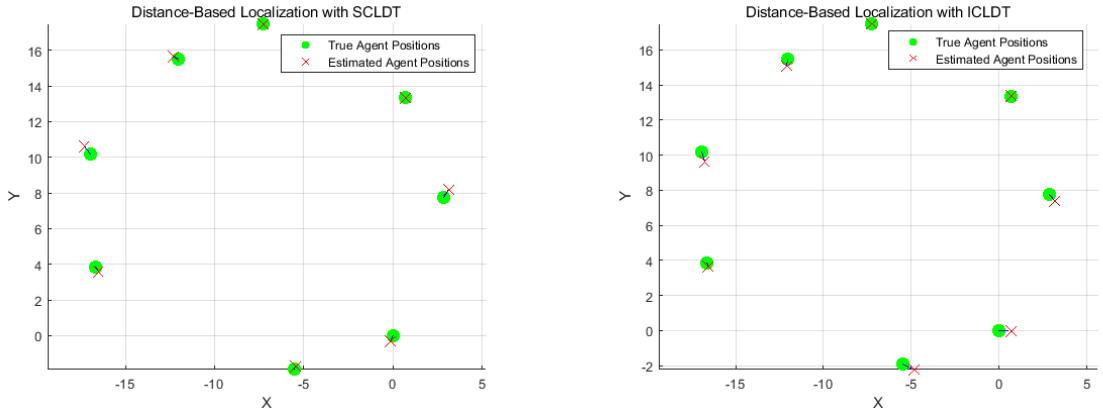
formance even under limited communication conditions, making it promising for deployment in large-scale, real-world multi-agent systems.

4.6 Discussion

To establish a clear performance benchmark, we begin by comparing the proposed method with a baseline approach. Although the baseline method is theoretically valid for linear systems, it fails in the tracking task due to the nonlinear nature of the target's actual motion. In contrast, our method can effectively handle such nonlinear dynamics. By employing a more appropriate modeling and estimation process, it maintains both tracking and localization errors at a low level throughout the experiment.In this section, we focus on the two methods proposed in this work that remain effective under nonlinear system dynamics. We conduct a systematic comparison of their performance in terms of algorithmic accuracy, communication overhead, and computational complexity. Through this comparative analysis, we further explore their suitability for different application scenarios and system constraints, aiming to provide practical guidance for real-world deployment.

4.6.1 Algorithm Performance Comparison

This work proposes two optimization strategies for cooperative localization and target tracking in multi-agent systems: the Sequential Optimization and the Integrated



(a) Localization with SCLDT (b) Localization with ICLDT

Figure 4.7: Comparison of localization results

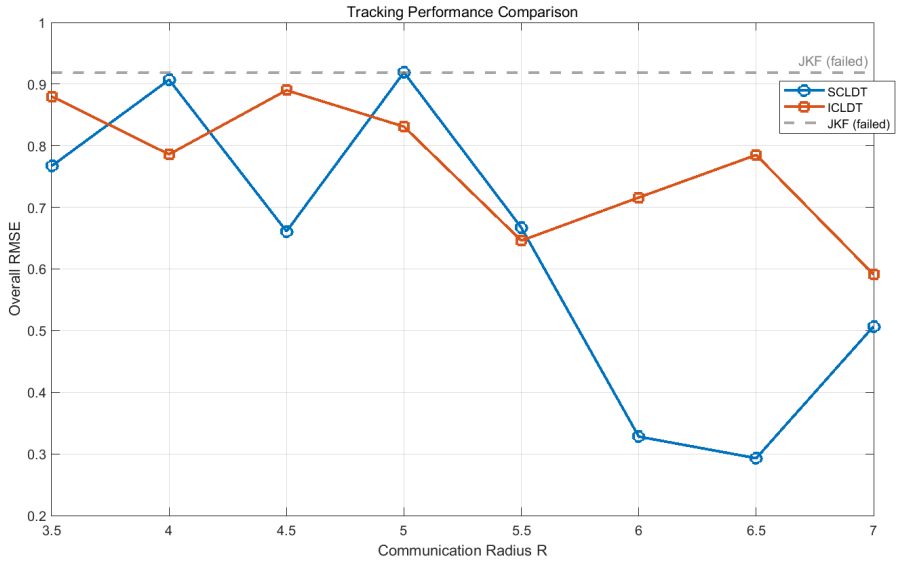


Figure 4.8: Tracking performance comparison.

Optimization. In this section, we analyze and compare their performance in terms of localization accuracy and tracking accuracy based on simulation results.

The sequential optimization method explicitly separates the localization and tracking tasks into two independent stages.

In localization phase, a distance-based optimization framework is employed, where non-anchor nodes iteratively update their positions by communicating with neighboring agents. Eventually, the positions of all nodes converge to a globally consistent coordinate system. Simulation results demonstrate that this method achieves high precision and robustness in static localization tasks: the average localization error is 0.2062 meters, with most nodes achieving errors below 0.5 meters. This indicates reliable localization performance under limited communication and measurement accuracy.

After localization is completed and node positions are fixed, the system enters the tracking phase. In this stage, a Gaussian Process (GP) model combined with the Alternating Direction Method of Multipliers (ADMM) is used for distributed optimization, allowing agents to collaboratively predict the target trajectory. Simulations show that with a communication radius of $R = 6.5$ meters, the method achieves its best tracking performance, with the overall RMSE reaching 0.2926 meters—the lowest among all tested configurations—demonstrating excellent tracking accuracy.

In summary, the sequential optimization method benefits from its stage-wise structure, achieving high precision and low error in both localization and tracking tasks.

The integrated optimization method aims to streamline the overall process by embedding a closed-form solution of the localization problem into the tracking objective, thereby achieving a formal integration of the two tasks. Although structurally more compact, the method does not perform explicit iterative localization, which leads to some compromises in accuracy.

Instead of iterative refinement, the localization task is relaxed into a convex form and solved in closed-form, which is then directly embedded into the joint objective function. While this approach improves execution efficiency, it comes at the cost of reduced accuracy. Simulation results show that the localization errors under the integrated optimization method generally range from 0.4 to 0.7 meters, with the maximum node error reaching 0.72 meters—significantly higher than the average error in the sequential approach.

Due to the degraded localization accuracy, the tracking performance also suffers. Across all tested communication radii, the best RMSE achieved by the integrated optimization method is 0.5907 meters (at $R = 7$ meters), nearly twice the best performance of the sequential optimization method. This indicates that although the integrated approach conceptually couples the two tasks, it does not outperform the traditional stage-wise strategy in terms of accuracy.

4.6.2 Communication Efficiency Comparison

In multi-agent systems, the communication topology plays a critical role in determining overall algorithm performance. This is especially true for joint localization and tracking tasks, where each agent can only obtain relative measurements of the target's position with respect to itself. Consequently, accurate cooperative-localization becomes essential. To evaluate communication efficiency, we begin by analyzing the localization component of both algorithms.

The sequential optimization algorithm is highly adaptable to sparse communication networks. It relies solely on local inter-agent distance measurements and employs iterative updates to propagate position estimates throughout the network. Even without a fully connected communication graph, the algorithm can still achieve globally consistent localization.

As shown in Figure 3.12 and Table 3.6, when the communication radius is set to 17 meters, each agent has only 3 to 5 neighbors on average. Under this sparse communication condition, the sequential method still achieves a localization error of 0.3624 meters, which is already lower than the 0.3831 meters obtained by the closed-form solution used in the integrated optimization method. This demonstrates that

sequential optimization can approximate—if not outperform—the closed-form solution in terms of accuracy, even under partial connectivity.

Moreover, at this communication level (17 meters), the tracking accuracy of the sequential algorithm is already comparable to its full-communication performance. This reinforces the algorithm's robustness and efficiency in scenarios with limited bandwidth or incomplete communication.

In contrast, the integrated optimization algorithm exhibits a stronger dependence on a fully connected communication graph. Its localization process relies on a closed-form solution derived from a convex relaxation of the original problem. To ensure the validity and convergence of this solution, comprehensive information exchange across the entire network is required.

Specifically, to accurately constrain the optimization problem and maintain global consistency, the integrated method assumes that each of the 8 agents is able to communicate with all others—effectively requiring a fully connected topology. According to simulation results, this necessitates a communication radius of at least 20 meters. Any relaxation of this requirement may lead to degraded performance or even failure of the localization module.

In summary, the sequential optimization method demonstrates clear advantages in terms of communication efficiency. It can achieve accurate localization and tracking even under partial communication, showing strong adaptability to limited-connectivity networks. The integrated optimization method, although structurally compact, requires full inter-agent communication, which may limit its deployment in real-world scenarios with constrained bandwidth or dynamic topologies.

Therefore, for resource-limited or large-scale multi-agent systems with sparse or unreliable communication links, the sequential optimization method provides greater practicality and scalability.

4.6.3 Computational Cost Comparison

In addition to performance and communication efficiency, computational complexity is an important metric for evaluating the practicality of multi-agent algorithms. This section compares the asymptotic computational costs of the Sequential Optimization and Integrated Optimization approaches, based on their algorithmic structure and simulation design.

Sequential Optimization. This method separates the joint estimation task into two distinct stages: static cooperative-localization and target tracking, which are executed sequentially.

Cooperative-localization: Each non-anchor agent iteratively updates its position estimate by minimizing the squared error between estimated and measured distances to its neighbors. For a single agent, each iteration requires evaluating a nonlinear cost over k neighbors in 2-dimensional space, with complexity $O(k \cdot 2)$. Assuming T_L outer iterations and T inner optimization steps per agent, the total localization complexity is:

$$O(T_L \cdot N_u \cdot T \cdot k \cdot 2)$$

where N_u is the number of non-anchor agents.

Target tracking: Each agent models the target dynamics using Gaussian Processes (GPs). The GP training requires $O(M^3)$ time per agent, where M is the number of local training samples. Distributed ADMM is used for consensus tracking, with T_T iterations, each costing $O(M^2)$ per agent. Hence, the total tracking complexity is:

$$O(N \cdot (M^3 + T_T \cdot M^2))$$

Total complexity:

$$O\left(T_L \cdot N_u \cdot T \cdot k \cdot d + N \cdot (M^3 + T_T \cdot M^2)\right)$$

Integrated Optimization. This method embeds a closed-form solution for localization directly into the tracking optimization, avoiding iterative localization and enabling single-round execution.

Localization (closed-form): Each agent computes a position estimate based on linearized distance constraints. The per-agent complexity is $O(k \cdot d)$, leading to total complexity:

$$O(N \cdot k \cdot d)$$

Joint tracking optimization: Similar to the sequential case, ADMM is used for distributed tracking with T_T iterations and complexity $O(M^2)$ per iteration per agent:

$$O(N \cdot T_T \cdot M^2)$$

Total complexity:

$$O(N \cdot k \cdot d + N \cdot T_T \cdot M^2)$$

Compared to the sequential strategy, the integrated approach avoids repeated nonlinear optimization and costly GP training, resulting in significantly lower computational load. The sequential method, although computationally heavier, offers better localization accuracy due to its explicit iterative refinement.

Table 4.5: Comparison of Computational Complexity

Metric	**Sequential Optimization**	**Integrated Optimization**
Localization method	Iterative nonlinear optimization	Closed-form linear solution
Localization cost	$O(T_L N_u T k d)$	$O(Nkd)$
Tracking cost	$O(N(M^3 + T_T M^2))$	$O(NT_T M^2)$
Optimization rounds	Two	One
Strengths	Higher accuracy	Faster execution

Both proposed methods successfully accomplish the coupled tasks of cooperative-localization and target tracking in nonlinear environments. The Sequential Cooperative Localization and Distributed Tracking (SCLDT) method, with its explicit two-stage structure, demonstrates superior accuracy and robustness in scenarios where precise estimation is critical or where reliable initialization can be ensured. It is well-suited for long-duration missions, low-communication settings, or deployments requiring high precision.

In contrast, the Integrated Cooperative Localization and Distributed Tracking (ICLDT) method leverages a closed-form localization strategy to enable single-round optimization. This results in significantly lower computational and communication overhead, making it more appropriate for real-time applications, resource-constrained platforms, or large-scale agent networks. The two methods thus offer complementary advantages and provide effective solutions under different operational constraints in distributed nonlinear systems.

4.7 Conclusion

This chapter proposed and evaluated two distributed solutions: a Sequential Cooperative Localization and Distributed Tracking (SCLDT) pipeline and an Integrated Cooperative Localization and Distributed Tracking (ICLDT) framework. SCLDT is modular and lightweight but is susceptible to one-way error propagation from localization to tracking. ICLDT couples both tasks in a single optimization, leverages consensus variables, and employs ADMM for distributed coordination; a convex relaxation of the localization subproblem yields closed-form updates that improve numerical stability.

Across diverse network topologies, radii, and noise levels, both methods outperform a Joint-KF baseline under nonlinear dynamics. ICLDT consistently achieves superior accuracy, faster convergence, and better robustness to sparse/partial observations, while maintaining favorable communication and computation trade-offs through local updates plus limited consensus. These results provide practical guidance for parameterization and deployment of decentralized perception systems in GNSS-denied scenarios such as maritime search-and-rescue.

5 Conclusion

In GNSS-denied scenarios such as maritime search-and-rescue (SAR), multi-agent systems must collaboratively perceive and track drifting targets under limited communication ranges, which makes centralized solutions difficult to deploy. The central question is how—without relying on a fusion center—to use only neighbor ranging and each agent's local observations to simultaneously estimate the agents' fixed positions and the trajectory of a non-cooperative target, while preserving scalability and robustness; this has become an important research direction in recent years. Thus, this work presents a systematic study of cooperative self-localization and distributed target tracking in multi-agent systems. Two optimization frameworks were proposed: the Sequential Cooperative localization and Distributed Tracking (SCLDT) method and the Integrated Cooperative localization and Distributed Tracking (ICLDT) method. A detailed comparison was conducted from the perspectives of estimation performance, communication efficiency, and computational complexity.

The SCLDT algorithm adopts a modular two-stage design, performing localization followed by tracking. This structure effectively prevents error amplification between tasks and leads to higher accuracy in both positioning and tracking. Simulation results demonstrate that SCLDT achieves an average localization error of 0.2062 meters, and the best tracking RMSE of 0.2926 meters when the communication radius is 6.5 meters, making it suitable for scenarios requiring high-precision estimation under limited communication.

In contrast, the ICLDT algorithm simplifies the optimization pipeline by embedding a closed-form localization solution into the tracking objective. While it does not jointly optimize localization and tracking in the strict sense, the integrated structure significantly reduces computational load and allows faster convergence. However, the accuracy is slightly compromised, with positioning errors ranging between 0.4 and 0.7 meters, and a best tracking RMSE of 0.5907 meters at a communication radius of 7 meters. Moreover, ICLDT requires a fully connected communication graph to guarantee its effectiveness.

In summary, the sequential method offers superior estimation accuracy and robustness, while the integrated method provides a lightweight alternative for real-time or large-scale systems.

Future work will focus on four main directions: (1) generalizing the current framework to three-dimensional space to support aerial and underwater robot applications; (2) extending the framework to handle multi-target tracking with robust data association and identity management under partial observability; (3) integrating learning-based dynamic models, such as deep Gaussian processes or neural surrogates, to better adapt to unknown or nonlinear environments; and (4) deploying the algorithms on real robotic platforms and conducting field experiments to evaluate performance under practical conditions such as communication dropouts, drift, and occlusions.

This work lays a theoretical and algorithmic foundation for resilient distributed perception and opens up promising directions for future research in cooperative autonomy.

Bibliography

[1] Z. Zhang, N. Li, and G. Yan, "The development of distributed cooperative localization algorithms for multi-uav systems in the past decade," *Measurement*, vol. 256, p. 118040, 2025.

[2] "Distributed localization for uav–ugv cooperative systems using information consensus filter," *Aerospace*, vol. 8, no. 4, p. 166, 2024.

[3] N. Farmani, L. Sun, and D. Pack, "A scalable multi-target tracking system for cooperative unmanned aerial vehicles," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 4, pp. 1947–1961, 2017.

[4] K. Su and F. Qian, "Multi-uav cooperative searching and tracking for moving targets based on multi-agent reinforcement learning," *Applied Sciences*, vol. 13, no. 21, p. 11905, 2023.

[5] "Multi-robot cooperative localization and target tracking," 2022, https://escholarship.org/uc/item/0q69h911.

[6] "Integrated design of cooperative area coverage and target tracking for multi-uavs," *Journal of Intelligent & Robotic Systems*, 2023.

[7] D. Tardioli, L. Riazuelo, A. Gil, J. L. Blanco, J. Gonzalez-Jimenez, and J. L. Martinez, "Robot teams for inspection in disaster scenarios," *IEEE Robotics & Automation Magazine*, vol. 22, no. 4, pp. 24–39, 2015.

[8] D. A. Paley, N. E. Leonard, R. Sepulchre, D. Grunbaum, and J. K. Parrish, "Collective motion: Modeling and analysis of mixed animal-human systems," *IEEE Control Systems Magazine*, vol. 28, no. 4, pp. 89–105, 2008.

[9] Z. Wu, B. Yuan, J. Lv, and Y. Yang, "Multi-robot coordination for intelligent warehouse: A survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 3, pp. 393–411, 2022.

[10] Y. Fan, X. Wang, and W. Zhang, "Distributed target tracking for border surveillance with heterogeneous mobile sensor networks," *Sensors*, vol. 20, no. 12, p. 3435, 2020.

[11] Y. Zhou, Y. Liu, P. Zhu, and X. Wang, "Distributed invariant kalman filter for cooperative localization using matrix lie groups," *arXiv*, vol. abs/2405.04000, 2024.

[12] R. Sharma, C. N. Taylor, and D. W. Casbeer, "Distributed cooperative slam using an information consensus filter," 2017, researchGate Upload.

[13] S. S. Kia, S. Rounds, and S. Martinez, "Cooperative localization for mobile agents: a recursive decentralized algorithm based on kalman filter decoupling," *arXiv*, vol. abs/1505.05908, 2015.

[14] S. Särkkä, A. Vehtari, and J. Lampinen, "Rao-blackwellized particle filter for multiple target tracking," Helsinki University of Technology, Tech. Rep., 2004.

[15] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[16] M. Kok *et al.*, "Rao-blackwellized particle smoothing for slam with gaussian processes," *Cambridge University Press*, 2024.

[17] S. J. Julier and J. K. Uhlmann, "A new extension of the kalman filter to nonlinear systems," in *Proceedings of SPIE AeroSense: 11th International Symposium on Aerospace/Defense Sensing, Simulation, and Controls*, vol. 3068, 1997, pp. 182–193.

[18] E. A. Wan and R. V. D. Merwe, "The unscented kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (AS-SPCC)*. IEEE, 2000, pp. 153–158.

[19] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.

[20] J. Ko and D. Fox, "Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models," *Autonomous Robots*, vol. 27, no. 1, pp. 75–90, 2009.

[21] M. P. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011, pp. 465–472.

[22] A. G. Wilson and H. Nickisch, "Kernel interpolation for scalable structured gaussian processes," *arXiv preprint arXiv:1503.01057*, 2015, available at https://arxiv.org/abs/1503.01057.

[23] Y. Shen, M. Seeger, and A. Ng, "Fast gaussian process regression using kd-trees," in *Advances in Neural Information Processing Systems*, 2006, pp. 1225–1232.

[24] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[25] C. Rasmussen and Z. Ghahramani, "Infinite mixtures of gaussian process experts," vol. 14, pp. 881–888, 2002.

[26] C. Yuan and C. Neubauer, "Variational mixture of gaussian process experts," in *Advances in Neural Information Processing Systems (NIPS)*, 2009, pp. 1897–1904.

[27] T. Nguyen and E. Bonilla, "Fast allocation of gaussian process experts," in *Proceedings of the 31st International Conference on Machine Learning (ICML)*, vol. 32, 2014, pp. 145–153.

[28] E. Snelson and Z. Ghahramani, "Sparse gaussian process using pseudo-inputs," pp. 1257–1264, 2006.

[29] J. Ng and M. Deisenroth, "Distributed gaussian processes," vol. 37, pp. 1481–1490, 2015.

[30] Y. Cao and D. Fleet, "Generalised product of gaussian process experts," 2014, unpublished manuscript / tech. report.

[31] V. Tresp, "Bayesian committee machine," *Neural Computation*, vol. 12, no. 11, pp. 2719–2741, 2000.

[32] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[33] W. Yin, M. Hong, and Z.-Q. Luo, "A decentralized consensus optimization framework for distributed machine learning," *IEEE Transactions on Signal Processing*, vol. 63, no. 22, pp. 5783–5798, 2015.

[34] M. P. Deisenroth and J. W. Ng, "Distributed gaussian processes," *Proceedings of Machine Learning Research*, vol. 37, pp. 1481–1490, 2015, 32nd International Conference on Machine Learning (ICML).

[35] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[36] Breivik, A. A. Allen, C. Maisondieu, and J. C. Roth, "Wind-induced drift of objects at sea: the leeway field method," *Applied Ocean Research*, vol. 33, pp. 100–109, 2012.

[37] B. Brushett, A. Allen, V. C. Futch, and C. J. Lemckert, "Determining the leeway drift characteristics of tropical pacific island craft," *Applied Ocean Research*, vol. 43, pp. 46–55, 2014.

[38] S. Zhao, J. Chen, and X. Wang, "Bearing-only target localization and tracking in wireless sensor networks," *Sensors*, vol. 15, no. 4, pp. 8742–8763, 2015.

[39] F. Xu, Q. Sun, and D. Zhang, "Maritime target tracking using asynchronous and intermittent observations in cluttered environments," *IEEE Sensors Journal*, vol. 21, no. 5, pp. 6331–6341, 2021.

[40] D. W. Casbeer, R. W. Beard, T. W. McLain, S. Li, and R. K. Mehra, "Cooperative forest fire surveillance using a team of small unmanned air vehicles," *International Journal of Systems Science*, vol. 37, no. 6, pp. 351–360, 2006.

[41] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[42] E. Dimitri, G. Battistelli, and L. Chisci, "Distributed target tracking by mobile sensors with limited field of view and bandwidth constraints," *Automatica*, vol. 152, p. 111016, 2023.

[43] Y. Tian, L. Ma, C. Wang, Y. Tan, and Y. Zhang, "Robust cooperative localization of uav swarm in gnss-denied environments," *Sensors*, vol. 20, no. 3, p. 854, 2020.

[44] S. Haykin, *Kalman filtering and neural networks*. John Wiley & Sons, 2001.

[45] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[46] L. Zhao, Y. Zhang, and Y. Wang, "Robust joint state and parameter estimation with correntropy filtering," *Signal Processing*, vol. 111, pp. 30–41, 2015.

[47] Z. Tang *et al.*, "Federated cubature kalman filter for multi-sensor imu/uwb fusion," *Sensors*, vol. 21, no. 14, p. 4823, 2021.

[48] W. Zhou and H. Zhang, "Distributed invariant extended kalman filter on lie groups for multi-agent systems," *arXiv preprint arXiv:2405.04000*, 2024.

[49] V. Tzoumas *et al.*, "Covariance intersection–based invariant kalman filtering for multi-agent pose estimation," *arXiv preprint arXiv:2409.07933*, 2024.