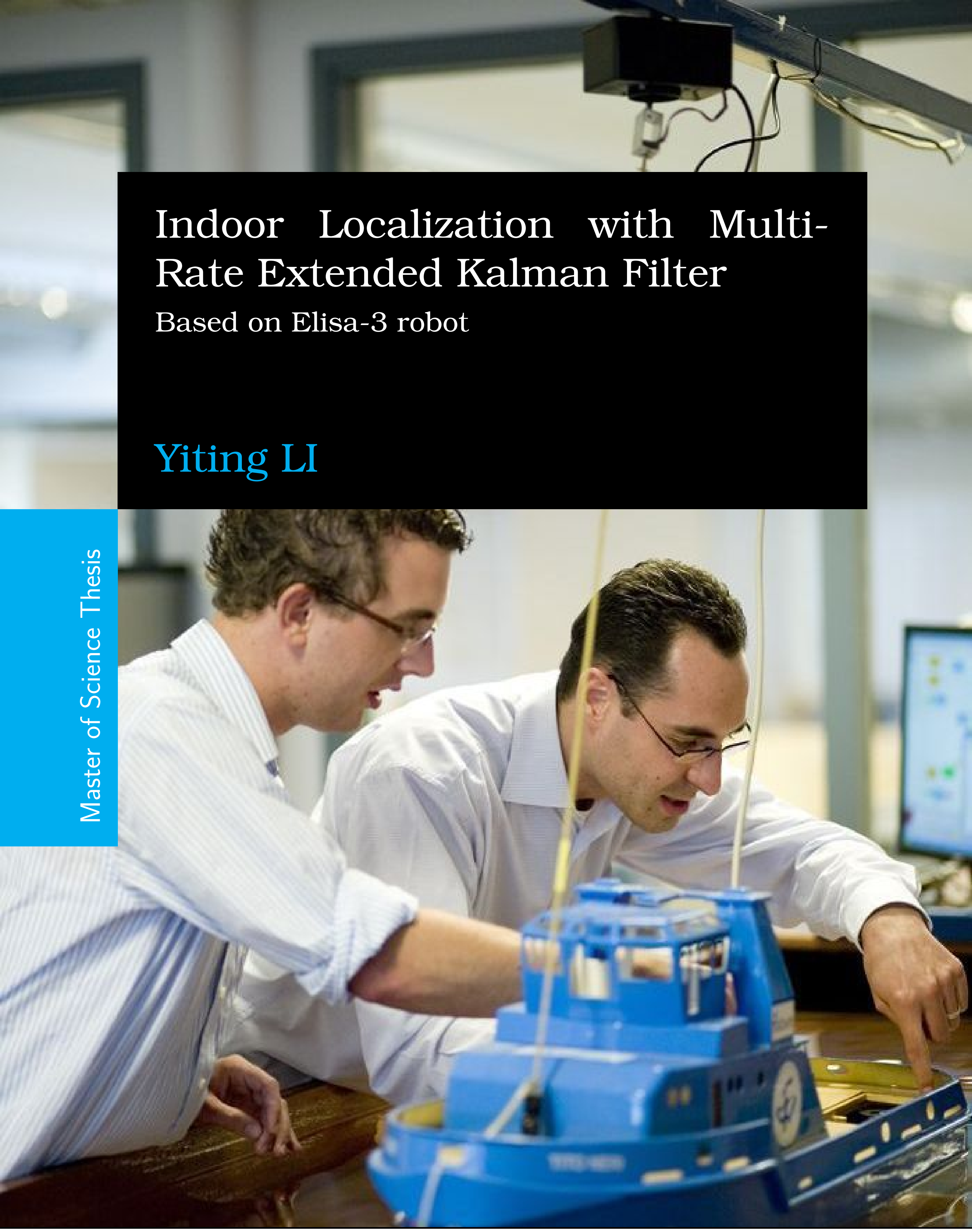

Indoor Localization with Multi-Rate Extended Kalman Filter

Based on Elisa-3 robot

Yiting LI

Master of Science Thesis

Delft University of Technology

Delft Center for Systems and Control

Indoor Localization with Multi-Rate Extended Kalman Filter

Based on Elisa-3 robot

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Yiting LI

June 5, 2023

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

Localization is one of the most fundamental competencies required by an autonomous robot, providing crucial information about its position for decision-making in indoor environments. In the current literature, an indoor localization system utilizes exteroceptive sensors such as GNSS(Global Navigation Satellite System), a camera, or an ultrasonic sensor and onboard sensors such as odometry or accelerometer to observe its environment. These sensor data are typically fused using Extended Kalman Filter (EKF) techniques. To solve the challenges posed by multiple sensors, different EKF variants such as Multi-Rate EKF and Single-Rate EKF have been proposed. Additionally, localization architectures like Cascade and OWA (Ordered Weighted Averaging) have been introduced to enhance the fusion process.

This thesis aims to develop an indoor localization system for Elisa-3 robots. The proposed localization system is evaluated through simulations and real-world experiments conducted using the Elisa-3 robots platform. Furthermore, the effectiveness of the localization system is assessed in a control task involving the formation of a consensus within a robot swarm.

The simulated and experimental results indicate that using a Single-Rate Extended Kalman Filter under an OWA architecture can generate an accurate and precise trajectory across a wide range of scenarios.

Table of Contents

List of Figures

List of Tables

Preface & Acknowledgements

I would like to thank my supervisor Dr. ir Manuel Mazo Espinosa for his assistance during this thesis, thank you Manuel for helping me through this thesis and letting me know what is project and thesis looks like. I would also like to thank my other committee member Guohao Lan, Ph.D for taking the time to serve as a committee member.

During the thesis, I spent time working with Elisa-3 robots in the DCSC lab. During that period I receive a lot of help from Steven Adams and Eva Zwetsloot. Thanks for their help, otherwise this process will be more difficult.

And thanks to developing technology, with assistance from ChatGPT, the clarity and coherence of my writing are improved.

Finally, I would like to express my deep gratitude to my family, and friends who always support me to stay, live and study on board for the whole time. Without them, this three-year journey would be less cheerful.

Dear reader, I hope the content discussed in this thesis will be of use to you for your own work, and hope this localization system will assist your future study.

Best wishes,

Yiting LI

Delft, University of Technology
June 5, 2023

Yiting LI

Chapter 1

Introduction

In the introduction chapter, the motivation and goals for this thesis are given. In the beginning, section 1-1 provides the reader with the motivation behind this research. Later, section 1-2 describes the goal of the thesis and possible solutions, and then the structure of the thesis is given.

1-1 Motivation

In an era of rapidly advancing technology, localization is becoming increasingly important for a wide range of applications, including self-driving cars, service robots, and autonomous drones.

Accurate localization is crucial for enabling the robot to navigate its environment effectively and reach its desired destination. Moreover, good localization enables the robot to make informed decisions about its movements and actions, resulting in more efficient task completion, which is important in the experiment and algorithm verification process.

For instance, in the validation experiment of the robot swarm consensus algorithm, it is crucial to have accurate localization for each robot. This is because valid and successful communication and broadcast between robots can only occur if each robot has an accurate estimation of its position. Therefore, the development of a reliable localization architecture is essential for conducting robot experiments.

The Elisa-3 robot produced by GCtronic is chosen as the platform for this thesis. The default system of the Elisa-3 robot for localization is the onboard odometry system. However, a stand-alone onboard sensor cannot always offer a precise and accurate position estimation.

A commonly used approach to raise the total estimation accuracy is employing additional sensors to improve the localization through sensor fusion. Sensors often used for achieving the above purpose are global sensors, such as Global Navigation Satellite System (GNSS) or cameras, those sensors can give absolute coordination. In this thesis, the additional sensor is the Optitrack camera system. However, the tracking accuracy of the Optitrack system can be affected by crowded agents and environmental light due to its infrared nature and utilization of refraction to track objects.

To overcome the sensors' flaws and obtain a good position estimation, applying a data-fusing method to those sensors' data proves to be an efficient approach. This fusion of different sensors is widely performed in many fields, such as data association, state estimation, and decision fusion [1]. Under the data fusion techniques category, there is a wide range of possible solutions, of which the Kalman filter developed by [2] is probably the most well-known and is employed in many cases [3], [4].

Through years of development, many advanced versions of Kalman filters have been proposed for solving non-linear systems, such as the Extended Kalman filter (EKF) [5]. Position estimation using odometry and cameras as measurements is often combined with a unicycle-like mobile robot. Due to the nonlinearity property of the robot model, a non-linear Kalman filter is preferred. Many studies utilizing a non-linear Kalman filter for localization can be found, For instance, in the paper [6], the data provided by odometry and sonar sensors are fused employing an extended Kalman filter to localize itself for a coordinate system.

In a multi-sensor system, sensors' measurement would often come at a different rate, e.g. the camera sampling rate might be lower than the odometry sampling rate due to the communication limitation. A relevant study on this topic is presented in [7], where the authors propose an approach that involves estimating the state vector using a neural network that combines the outputs of multiple multi-rate Kalman filters. Each sensor system is associated with its own Kalman filter in this framework.

Apart from the multi-rate property, the sensor's measurement might get lost, e.g. Optitrack measurement might get lost due to crowded agents. In [8], the author provides a way to deal with missing measurements situation based on multi-rate non-linear Kalman filter for localization tasks.

1-2 Thesis Goals and Thesis Structure

1-2-1 Thesis Goals

With the above description of the thesis's motivation, the goal of this thesis can be summarized as follows:

The goal of this thesis is to develop a localization architecture specifically designed for Elisa-3 robots to enhance their localization capabilities in indoor environments. The architecture leverages the measurements from onboard odometry sensors as well as the global Optitrack camera system to achieve accurate and reliable robot localization. By fusing the information from these sensors, the thesis aims to overcome the limitations of individual sensors and improve the overall localization performance.

To reach this purpose, the thesis working flow can be further divided into two subsections:

- **Step 1: Construct and simulate the localization architecture**

  The localization architecture should satisfy the requirements below:

  1. Be able to overcome the flaw of each sensor.

     The odometry and Optitrack camera could give biased measurements, the localization architecture should extract useful information from the sensors' readings which could be wrong.
  2. Handle multi-rate data and missing sensor measurement.

     The localization technique should also provide reliable estimation even when some measurements are not available.

  Based on the literature, the development of a Multi-Rate Extended Kalman filter with a non-linear kinematic unicycle-like robot model should allow for the required modularity as well as the ability to deal with GNSS outages.

- **Step 2: Validate the architecture on a swarm of Elisa-3 robots**

  After the creation and testing of a filter, this filter should also perform well in a real robot platform. The position estimation should be accurate and in real-time.

1-2-2 Thesis Structure

The structure of the thesis is as follows:

In chapter 2, the studies and research about approaches, algorithms, and techniques for robot localization, possible solutions and discussion are presented.

Chapter 3 provides a detailed description of the mathematical definitions and models used in the localization architecture. The kinematic and dynamic models of the Elisa-3 robots are presented, along with the formulation of the localization problem. The chapter also introduces the Optitrack system, which is the exteroceptive sensor used for robot location tracking.

Then in chapter 4, the description of all EKF localization structures used in the following chapters is given. Different variations and modifications of the EKF used in the subsequent chapters, such as the multi-rate EKF and the OWA EKF, are explained in this chapter.

Later, the effect of the proposed localization architecture is first evaluated through simulation in chapter 5. The basic setup of the simulation is presented and the simulations include different noise levels, different sampling rates, and the presence of measurement losses. The results are analyzed and compared to demonstrate the effectiveness of the proposed architecture.

In chapter 6, the implementation of the proposed localization architecture on the Elisa-3 robot platform is presented. The experiments include different architectures, different EKF versions, and their effects on localization performance.

Then the proposed localization architecture is applied to a real-world control task to verify the applicability of the localization architecture. The introduction of the control task, results and summary are shown in this chapter.

Hereafter, chapter 7 sums up the results and the added value of the proposed solution to the available literature. Furthermore, this chapter presents possible improvements as well as provides avenues for future research. Lastly, this chapter restates the goals of the research and summarizes the achieved results.

Chapter 2

Literature Review

In the literature review chapter, the basic concept of the robots localization problem, possible solutions and discussion will be given. In the first section 2-1. Then the next section 2-2 will elaborate on one technique that is widely applied in the localization field, data fusion techniques.

Later in the section 2-4, more details about the Kalman filter-based method, a sub-case under the data fusion method category, will be illustrated. Next, in section 2-5, several ways of combing different filters' output are introduced. In the end, section 2-6 will evaluate those methods and lay out the proposed solution contained in the thesis.

2-1 Localization Task

Basic Definition

Localization is one of the most fundamental competencies required by an autonomous robot, as the knowledge of the robot's own location is an essential precursor to making decisions about future actions [9].

Robot localization is the process of determining where a mobile robot is located concerning its environment, and it provides an answer to the question: Where is the robot now?

Localization tasks can be categorized into two classes, considering whether the global map is available or not.

When the global map is unknown, robots can't give absolute location information until the global map is established. Therefore, the robot needs to estimate both its location and landmark locations simultaneously. In this case, methods such as SLAM [10], [11] are often applied. It would be more difficult and time-consuming for one robot to report its location now, compared to the case where the whole map is available since the robot needs to explore the environment first and gather enough information and then it can exploit the environmental information to localize itself.

Another category is that the environment map is known and available, as this article studies the case where the lab field is already well-established, we won't use the robot to explore the field, and the localization problem then becomes estimating the robot pose (position and orientation) relative to the coordinate frame in which the map is defined.

To restrict the problem, the robot is assumed to move in a 2-dimensional flat field. With this setup, the robot itself should have a relatively accurate estimation of its global 2-D coordination.

Under the category that the global map is known, we can further divide the problem into two sub-categories, considering whether the initial position is known to one robot. In this article, the initial position information is known to robots, and then the 2-D localization problem is converted into a position-tracking problem, which is mainly discussed in this article.

If the initial position is unknown to robots, then the localization problem becomes a global localization problem. Besides that, one tracked robot could be abducted (kidnapped) to an unidentified place, then the robot needs to re-track its location, and then the problem turns into a Kidnap recovery problem.

Position Tracking

In the position tracking problems, the robot's initial position is known, and the objective is to track the robot or a swarm of robots at each instance of time during its navigation in the environment [11].

In a typical robot tracking scenario, the information, including coordination and orientation, can be gathered using onboard or global sensors available for computing the robot's location.

However, those sensor measurements are not always reliable, and they might be noisy, biased or even lost. For example, the onboard odometry may encounter a position accuracy degradation problem caused by the drifting situation, which is plagued by bias and noise, leading to unbounded growth in error when some variables are double integrated with controller[12].

Therefore, considering the above flaw, the robot should localize itself concerning the map of its environment. To achieve that, the robot can utilize exteroceptive sensors such as a laser sensor, GNSS, vision and an ultrasonic sensor to observe its environment. These sensors' information can be combined with the robot's odometry to localize the robot and also results in keeping the position uncertainty from growing unbounded.

The robot's accurate position can't be measured directly, even with a global positioning sensor (GPS). The robot can only extract its sensors' data to gain knowledge regarding the best estimate of its real location.

Therefore, to get a more accurate estimation of position, the robot localization techniques need to deal with noisy, biased, or even missing observations and integrate that information to generate not only an estimate of the robot's location but also a measure of the uncertainty of the location estimation.

Then we can summarize the general process of the robot's position as follows:

- prediction (action) update
- measurement correction update

The robot makes a location prediction via the system dynamics model in the first step. For robots that are unable to give a clear system model, the prediction is generated using the onboard sensor, such as odometry. In this step, the uncertainty grows, and it accumulates over time.

In the second step, the robot corrects the estimated position by combining multiple sensors' information. The localization technique that the robot unitizes here is the data fusion technique. Studies and research have been done for decades to find an efficient way to fuse and combine multiple sensors' data on different occasions.

2-2 Fusion Techniques

Data fusion is extensively employed in different fields, such as navigation, air traffic control, process control, and robotics [13], [14]. In the paper [13], the author proposed an information fusion algorithm called IF-Matching for map-matching. In the paper, [14], a fusion technique combines time-of-arrival (TOA) and received-signal-strength (RSS) to localize a mobile target through a wireless network.

Furthermore, integrating multiple sensors contributes to more reliable results than a separate sensor due to complementary information among the resources. Different sensors can provide complementary information on original measurements and details that compensate for the weaknesses of the others. It can improve the signal-to-noise ratio, decrease uncertainty and ambiguity, and increase reliability, robustness, resolution, accuracy, and other desirable properties [15], [16], [17].

Data fusion is a multidisciplinary area that involves several fields, and it is difficult to establish a clear and strict classification. The employed methods and techniques can be divided according to the many criteria, such as the architecture type, data fusion levels defined by the JDL model (Joint Directors of Laboratories model) or relations between the input data sources.

But according to the paper [1], we can categorize the available data fusion techniques into three nonexclusive categories based on the problem where the fusion technique will be applied:

- Data association: Hall and Llinas [18] provided the following definition of data association: "The process of assigning and computing the weights that relate the observations or tracks (A track can be defined as an ordered set of points that follow a path and are generated by the same target.) from one set to the observation of tracks of another set."
- State estimation: State estimation techniques aim to determine the state of the target given the observation or measurements. The state estimation phase is a common stage in data fusion algorithms because the target's observation could come from different sensors or sources, and the final goal is to obtain a global target state from the observations.
- Decision fusion: A decision is typically taken based on the knowledge of the perceived situation, which is provided by many sources in the data fusion domain. These techniques aim to make a high-level inference about the events and activities that are produced from the detected targets.

In this article, to solve the 2-D localization problem for a robot swarm, the main task is to let each robot has an accurate state estimation of itself, and then the problem goes into the classification of State estimation.

Most of the state estimation methods are based on control theory and employ the laws of probability to compute the estimation of a vector state from a vector measurement or a stream of vector measurements.

2-3 Probabilistic-Based Fusion Method

Under the category of State estimation, the most well-known method is Probabilistic based fusion method. Probabilistic methods rely on the probability distribution/density functions to express data uncertainty. At the core of these methods lies the Bayes estimator, which enables the fusion of pieces of data [19].

Assuming the system has a state-space model representation, the Bayes estimator provides a method for computing the posterior (conditional) probability distribution/density of the hypothetical state $x(t)$ at time t, based on the set of measurements $Z^t = \{z_1, z_2, \dots z_t\}$ up to time t and prior probability distribution $p(x(t)|Z^t)$of current state $x(t)$. The prior probability distribution is as follows:

$$p(x(t)|Z^{t-1}) = \frac{p(Z^t|x(t))p(x(t)|Z^{t-1})}{p(Z^t|Z^{t-1})} \tag{2-1}$$

where $p(Z^t|x(t))$ is called the likelihood function and is based on the given sensor measurement model, $p(x(t)|Z^{t-1})$ is called the prior distribution and incorporates the given transition model of the system.

One can apply the Bayes estimator each time and update the probability distribution or density of the system state by fusing the latest coming piece of data, i.e. z_t, recursively. Based on this idea, fusion algorithms are developed, such as Bayesian reasoning methods[20], [21], Dempster–Shafer method[22] and least square fusion methods, Kalman filtering.

Among these methods, the Kalman filter is one of the most popular fusion methods mainly due to its simplicity, ease of implementation, and optimality in a mean-squared error sense, when measurement noise follows Gaussian distributions and the system dynamics are linear. It is a very well-established data fusion method whose properties are deeply studied and examined both theoretically and in practical applications. However, the optimal statistical estimations are only obtained by the recursive Kalman filter if the main assumption is satisfied: The system could be described as a linear model and the error could be modelled as the Gaussian noise. It might be difficult to fulfil these assumptions in real applications.

When dealing with non-linear system dynamics, one usually has to resort to approximation techniques. For instance, the Extended Kalman filter (EKF) [5] and Unscented Kalman filter (UKF) [23], which are extensions of the Kalman filter applicable to non-linear systems, are based on the first-order and second-order approximations as a Taylor series expansion about the current estimate, respectively. The EKF is one of the most often employed methods for fusing data in robotic applications.

While dealing with both non-linear and non-Gaussian noise, another well-known alternative for Kalman filtering is the Particle Filter. Particle filters [24] is a recursive implementation of the SMC algorithm. This method builds the posterior density function using multiple random samples called particles, which will be propagated over time with a combination of sampling and resampling steps. At each iteration, the sampling step is employed to discard some particles, e.g. particles that violate physical constraints, increasing the relevance of regions with a higher posterior probability. In the filtering process, several particles of the same state variable are employed, and each particle has an associated weight that indicates the quality of the particle. Therefore, the estimation is the result of a weighted sum of all of the particles.

The Particle filters are very flexible as they do not make any assumptions regarding the probability densities to be approximated. However, they have some disadvantages. When compared to the Kalman filter, particle filters are computationally expensive as they may require a large number of random particles to estimate the desired posterior probability density and obtain a small variance in the estimator.

2-4 Kalman Filter Based Methods

The Kalman filter is the most popular estimation technique, even outside the state estimation category. It was originally proposed by Kalman [2] and it is an efficient filter, which can estimate the state of a dynamic system in the combined information of many uncertain situations, and is a powerful and versatile tool.

Kalman filters are ideal for systems that are continuously changing. They have the advantage that they are light on memory (they don't need to keep any history other than the previous state), and they are very fast, making them well-suited for real-time problems and embedded systems.

The Kalman filter works in an iterative way, it can be divided into two parts, firstly it's the prediction part, where the predicted state estimate is evolved from the updated previous updated state estimate via the system model. Secondly, in the update stage, the measurement residual (difference between expected measurement and real measurement) multiplied by the Kalman gain K is taken into account to provide the correction of the estimation in the first stage.

The Kalman filter has been widely studied and applied since its creation. Due to the existence of a large number of nonlinear systems, many variants of the Kalman filter were invented, such as EKF and UKF.

For example in [6], to let an autonomous mobile robot localizes itself with respect to a coordinate system, the data provided by odometry and sonar sensors are fused together by means of an extended Kalman filter. In this paper, the experiments confirmed that high performances of the localization algorithm are really obtainable in a wide range of experimental situations.

The [25], deals with the problem of mobile-robot localization in structured environments. The EKF is used to localize the four-wheeled mobile robot equipped with encoders for the wheels and a laser range finder sensor for scanning the environment. Through the experiments, the good localization results demonstrate the applicability of the EKF-based method for mobile robot localization.

UKF is another alternative to EKF facing a non-linear system. The comparison between EKF and UKF was widely invested. In [26], the author compares various simultaneous localization and mapping (SLAM) algorithms, the author concludes that UKF-based FastSLAM is observed to be the most efficient algorithm, while the EKF-based FastSLAM can perform as well as UKF SLAM algorithm under certain conditions. The paper [27] compares the EKF and UKF performance on a localization problem which is tackled on the basis of angles-only measurements, and paper [28] compares these two algorithms on underwater navigation systems, both papers conclude that experimental results exhibited a satisfactory localization accuracy for both EKF and UKF, the latter being more accurate than the former.

2-4-1 Multi-rate EKF

In multi-rate multi-sensor systems, the sensor might give measurements at different sampling rates, and those data could come asynchronous. For example, in the real application of multi-rate multi-sensor systems, onboard odometry and a global camera is often applied, where the sampling frequency of the former will be higher than that of the latter.

For dealing with that property, the fusion technique of multi-sensor systems has been investigated in the literature [7], [29]. In literature[7], where the state vector is estimated with a neural network that fuses the outputs of multiple multi-rate Kalman filters, one filter for each sensor system. In the paper [29], the author established the discrete dynamic system model based on every single sensor and presented an asynchronous multi-rate multi-sensor state fusion estimation algorithm.

The main workflow of multi-rate multi-sensor usually can be described in two steps. First, the state is individually observed by each multi-rate sensor. Afterwards, the estimations obtained by various sensors are combined concurrently.

In paper [29], the author gives the state space model based on each sensor index i. The author also stated that by dividing the data, including the state and the measurements, etc., into data blocks, and by rewriting the state space model as presented in the paper, the multi-rate multi-sensor data fusion problem is transformed to be the multi-sensor data fusion with the same sampling rate formally. As a result, the problem is simplified.

Later the author proves that the estimation results are unbiased and optimal in terms of minimum covariance. It is also confirmed that the fused estimations are more accurate in comparison with the estimation results obtained from a Kalman filter using a single sensor and the accuracy of the fused estimate will decrease if any of the sensors' information is neglected.

2-5 Localization Architecture

When dealing with multiple sensor sources, it is important to carefully consider the method used to fuse the data obtained from each sensor. This is because the way in which the data is combined can significantly impact the overall performance of the system.

Different fusion methods can produce varying results, and some may be more effective than others depending on the quality and reliability of the fused data. The quality and reliability of data can be affected by a variety of factors, including the sensors' accuracy, precision, and noise characteristics.

For instance, state estimation problems are often accompanied by missing measurement conditions. The missing measurements conditions are likely to occur in multi-sensor systems when performing state estimation at different rates, it can also happen due to the low quality of the communication and flaw of the sensor, for example, when robots are underwater, the GNSS connection is weak or robots are too close for a global sensor to recognize.

Moreover, one sensor could become less reliable over time due to sensor drift. If we rely too heavily on a less reliable sensor when fusing data from multiple sensors, it can negatively impact the overall performance of the system, because of the introduced bias or uncertainty.

There are many different methods that can be used to combine multiple fused results from different sensors, such as cascading, ordered weighted averaging (OWA), Fuzzy logic and Neural networks.

The OWA is simple and intuitive, it can be effective in many applications, easy to implement, and computationally efficient. In the paper [30], an optimally distributed fusion Kalman filter is designed based on the optimal fusion criterion weighted by matrices with missing sensor measurements and packet dropouts. When facing missing sensor measurements or packet dropouts conditions, the problems are to be solved by applying a group of optimally weighted matrices that can minimize the mean square error.

In the paper, [31], The author designed the Multi-rate Kalman filters to estimate a target position at various sampling rates. Later, the Ordered Weighted Averaging (OWA) operator is utilized to integrate multi-rate Kalman filters and improve the estimation quality. In the OWA operator, the weighting factors are updated based on a real covariance matrix, which can be further determined depending on the $r_i(k)$ innovation matrix or residual matrix of each sensor. The $r_i(k)$ is the summation of the difference between expected values and measurements during a period of time.

Moreover, a fuzzy logic method can be used to combine multiple fused results from different sensors. In the paper [32], the authors propose a fuzzy logic-based fusion method for integrating multiple features extracted from different sensors including radar and acoustic signals, for target recognition. The proposed method uses fuzzy rules and operators to combine the features from each sensor, and the weights of each feature are determined using a fuzzy entropy-based method. The experimental results show that the fuzzy logic-based method outperforms other fusion methods such as principal component analysis and support vector machines. A similar example can be seen in paper [33], the proposed method used fuzzy rules to combine the data from each sensor, and the weights of each sensor were determined using a fuzzy entropy-based method.

The fuzzy logic based methods would require careful design of the fuzzy rules and operators, which may be sensitive to the choice of membership functions. The designing of fuzzy rules wouldn't be a good choice if the fuzzy rules are complex.

Another intelligent-based method, such as the neural network method is also applied. But with an intelligent-based method, it's not necessary to combine multiple fused results from different sensors, but directly fuse the raw data from multiple sensors. For example, in the paper [34] and [35], multiple layers of neural network based architectures are developed to fuse sensors' data for pose estimation.

2-6 Discussion

The major goal is to develop a localization architecture for Elisa-3 robots to better localize themselves in an indoor environment based on onboard odometry sensors' and the global Optitrack camera system measurement. Several facts can be stated: The mathematical models of the robot system and the measurements are known. The multi-sensor system has various sampling rates, and missing measurements are expected to occur.

Based on the above facts, multi-rate EKFs are first applied to generate fused results based on odometry data, camera measurement and system model. Next, since the fuzzy logic requires careful design of the fuzzy rules and the neural network method requires a large amount of training data, which would be unnecessary for this case, an OWA operator and cascade architecture would be implemented at the output of multi-rate EKFs to combine the generated information.

Chapter 3

System Description

In this chapter, the system description is given, the first section 3-1 illustrates the multi-sensor system on the robot. Then in the section 3-2, basic information about Optitrack System and possible bias is introduced. Section 3-3 gives the system state space model. Based on the above, lastly, section 3-4 gives the system block diagram as a full description.

3-1 Sensors of Elisa-3 Robot

Robots commonly rely on a variety of sensors to effectively perform localization tasks. These sensors encompass a range of technologies, including Lidar, radar, sonar, proximity sensors, and laser range finders. Additionally, robots can leverage global sensors like GNSS (Global Navigation Satellite System) and camera systems to enhance their localization capabilities. By integrating data from multiple sensors, robots can achieve more accurate and robust localization in diverse environments.

This thesis focuses on the localization of Elisa-3 robots, and the current state estimation of the robot is obtained through three distinct methods. The first approach involves utilizing the system dynamics model to estimate the robot's state. The second method relies on the measurements provided by internal sensors, such as odometry, which capture the robot's movement and position. The third alternative involves leveraging the Optitrack camera system to capture images of the robot and extract precise location information.

The sensors used in the localization of Elisa-3 robots often operate at different sampling rates. Figure 3-1 illustrates a multi-sensor diagram with two sensors operating at different sampling rates. The red arrow in the diagram represents a lower-rate sensor, characterized by a sampling period of 3, while the other sensor operates at a higher sampling rate. Moreover, in real-world scenarios, sensor readings are not always perfect and can be subject to various issues, including communication problems. Therefore, there is a possibility of data loss or inconsistency in sensor readings, further complicating the localization process.

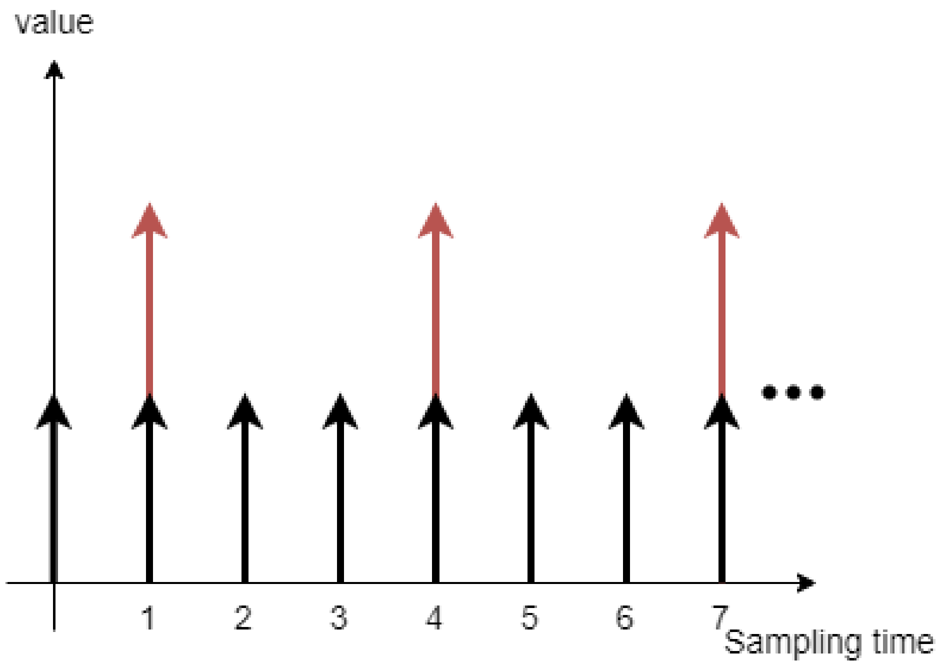

Figure 3-1: Multi sensor diagram

3-2 Optitrack System

The Network Embedded Robotics Lab of DCSC utilizes the OptiTrack motion camera system, which employs an infrared camera system to accurately track the location and orientation of objects. The system operates at a high rate of 120 Hz, providing precise data with millimetre-level accuracy. Figure 3-2 shows an example of 10 Elisa-3 robots placed in the field, and figure 3-4 displays a screenshot of the Optitrack system capturing the robot positions.

It is important to note that the reliability of the OptiTrack system depends on the distance between the objects being tracked. When multiple objects are too close to each other, the system may have difficulty identifying each of them and may merge them into a single object, resulting in some robots being untracked. In the example shown in Figure 3-3, three robots in the red box were placed close enough to each other to cause tracking issues. As a result, Object 8 and 9 were missing their position in the OptiTrack system, as shown in Figure 3-5.

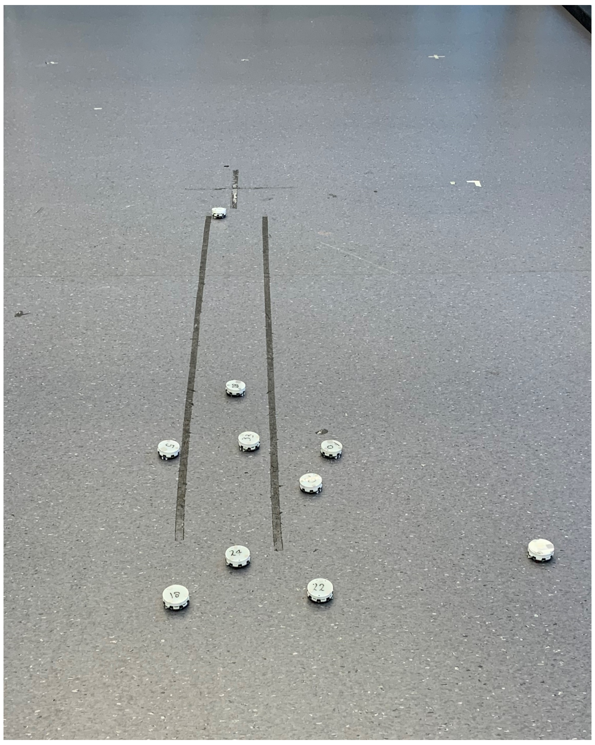
Figure 3-2: Robots Placement

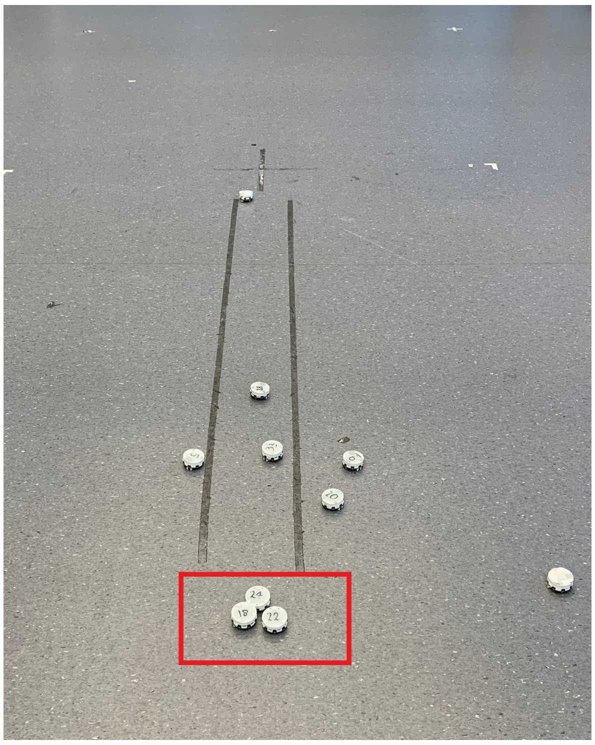
Figure 3-3: Robots Placement with Merging

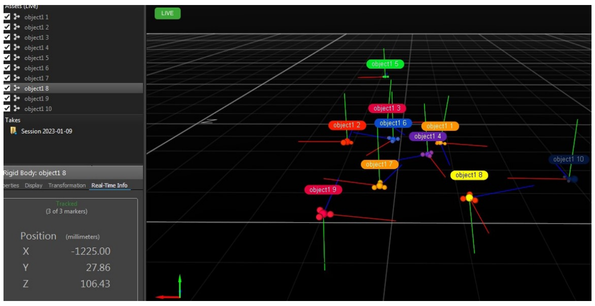
Position (millimeters)
X -1225.00
Y 27.86
Z 106.43

Figure 3-4: Robots Placement in Optitrack

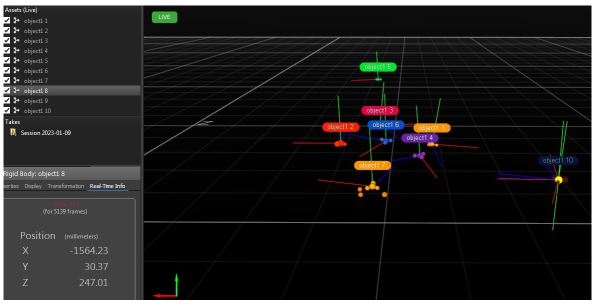
Position (millimeters)
X -1564.23
Y 30.37
Z 247.01

Figure 3-5: Robots Placement in Optitrack with Merging

Moreover, since the Optitrack system utilizes infrared reflecting, the tracking might become unstable or lost due to environmental light changes. When the object is standing still in the lab, the camera measurements have small noise, as shown in figure 3-6, where the variance is negligible (less than 1e-6), and the reason for the little rising could be the variance of environmental light. This shows that the camera has a relatively accurate position measurement when one object is being tracked. Then we can reach a statement that the main bias from the camera side is the merging condition and data missing.

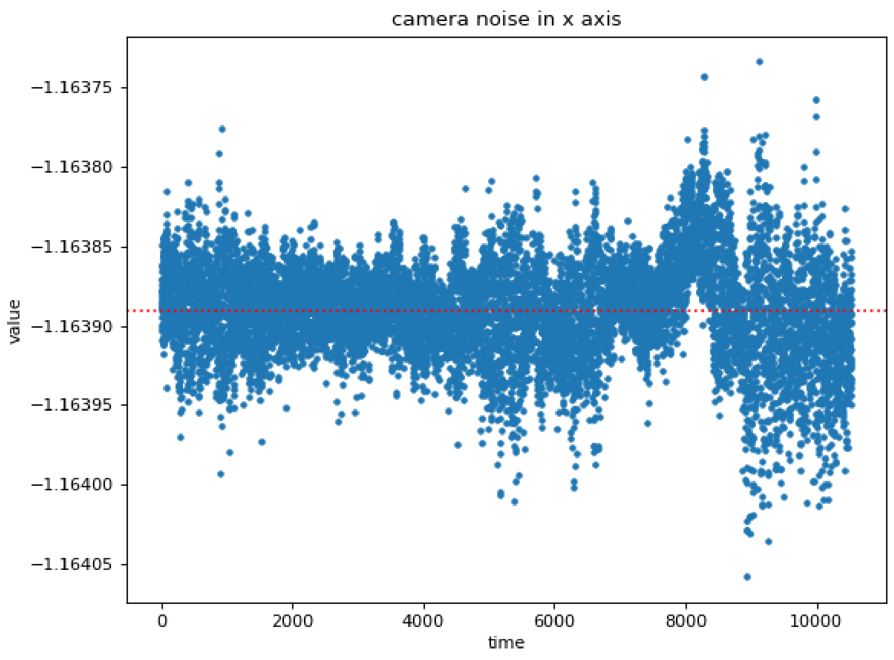

Figure 3-6: Camera Noise in x Axis

3-3 System Model

In this section, the system dynamics model and controller for the robot's moving are presented.

3-3-1 System Dynamics

For a two-dimension localization task on a unicycle-type robot(Elisa-3 robots), according to [36], the kinematic model of each robot can be defined by the following state space iteration equation:

$$
\begin{aligned}
x(k+1) &= x(k) + v_x(k)cos(\theta(k)) + w_1(k) \\
y(k+1) &= y(k) + v_y(k)sin(\theta(k)) + w_2(k) \\
\theta(k+1) &= \theta(k) + \omega(k) + w_3(k)
\end{aligned}
\tag{3-1}
$$

Where $X(k) = [x(k), y(k), \theta(k)]^T$ is defined as the states vector, it describes the global position $x(k), y(k)$ and orientation vector $\theta(k)$ of the robot at time k and $k \in \{1, 2, ..., N\}$.

The control input at time k is denoted as $u(k) = [v_x(k), v_y(k), \omega(k)]^T$, where $v_x(k)$ and $v_y(k)$ represent the linear velocities in the x and y directions, respectively, and $\omega(k)$ represents the angular velocity.

The noise $w_i(k)$ that occurs in the state transition is assumed to be a zero-mean white Gaussian noise with a variance denoted as $Q_i(k)$. The specific value of the variance can be determined through multiple experiments to accurately characterize the noise.

With the state vector and the control input vector, suppose there are N available sensors, the state space model of the robot can be rewritten as follows:

$$\begin{aligned} X(k+1) &= f(X(k), u(k), w(k)) \\ Z_i(k+1) &= \gamma_i(k) h_i(X(k), v(k)), i = 1, ..., N \end{aligned} \tag{3-2}$$

where $f(X(k), u(k), w(k))$ is the state transition function, and states can be observed through different sensors, $h_i(X(k), v(k))$ is used as a measurement function for ith sensor. The variable $\gamma_i(k)$ represents the possibility of data drop for the ith sensor at time k. Both the state transition function and the measurement function can be nonlinear.

The kinematic model of a unicycle robot provides an ideal representation of its motion. However, in real-world scenarios, the robot may experience various situations that cause the actual position to deviate from the model's output. Factors such as wheel slipping or getting stuck can introduce uncertainties and errors in the robot's motion, resulting in deviations between the expected and actual positions.

Also, to account for Optitrack System's imperfection and camera noise, perturbations $w(k)$ and $v(k)$ are introduced in the state transition and measurement, respectively. These perturbations are assumed to follow a zero-mean white Gaussian distribution and are uncorrelated with each other.

3-3-2 Robot go-to-goal Controller

These robots need to decide the velocity and heading angle at the next step to reach their destination, basing their current position and heading. Therefore, they are assumed to perform a simple go-to-goal task with a P controller.

The controller follows the equation below:

$$\begin{aligned} e(k) &= [X - x(k), Y - y(k)] \\ V(k) &= \|K_{P1} e(k)\| \\ \phi(k) &= \arctan\left(X - x(k), Y - y(k)\right) \\ \omega(k) &= K_{P2} \arctan(sin(\phi(k) - \theta(k)), cos(\phi(k) - \theta(k))) \end{aligned} \tag{3-3}$$

where the parameter K_P was chosen as $K_P = [K_{P1}, K_{P2}] = [-0.001, -0.001, 0.001]$, with that the parameter, the robot will move in a moderate velocity and a smooth turning during the simulation. And X and Y are the endpoints of one agent.

The control input can be expressed as follows, where the $X(k)$ is the state vector in time k including $x(k)$, $y(k)$ and $\theta(k)$.

$$u(k) = [v(k), \omega(k)] = f_1(X, Y, X(k)) \tag{3-4}$$

3-4 System Model Block Diagram

With the above description, we can express the localization system behaviour using a block diagram as shown in figure 3-7.

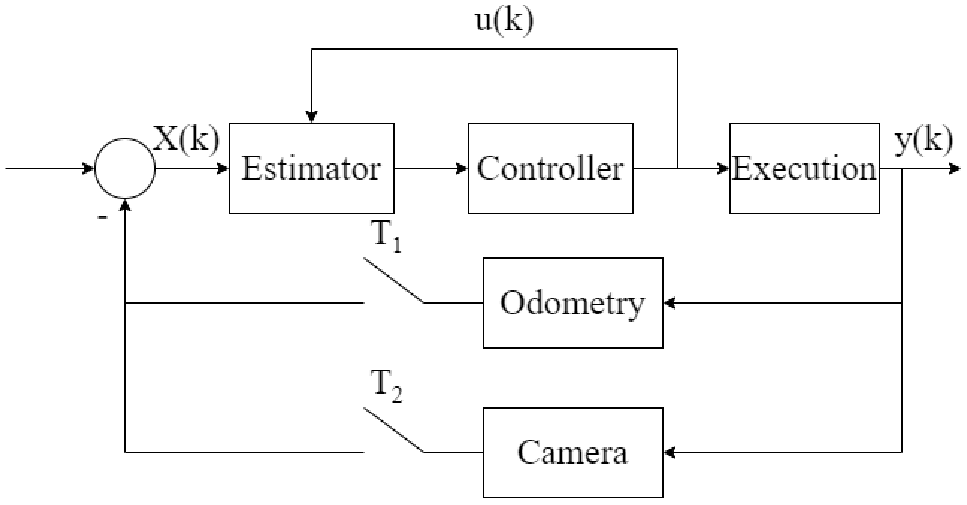

Figure 3-7: System Block Diagram with 2 Sensors

The robot is equipped with a pre-set goal point. To navigate towards this goal point, the robot determines its desired velocity and heading $u(k)$ based on equation 3-3. By applying this control input, the robot takes a step forward in its motion to reach a new location that brings it closer to the goal point.

The robot's new location is captured using two distinct sensors, namely odometry and camera, which operate at different sampling rates represented as T_1 and T_2, respectively.

Another alternative is to assume that the system dynamics follow the same model as the odometry, resulting in a single measurement from the camera with a sampling rate of T_1. In this case, the camera measurement is used as the primary source of information for estimating the robot's state, and the odometry is not considered a separate measurement.

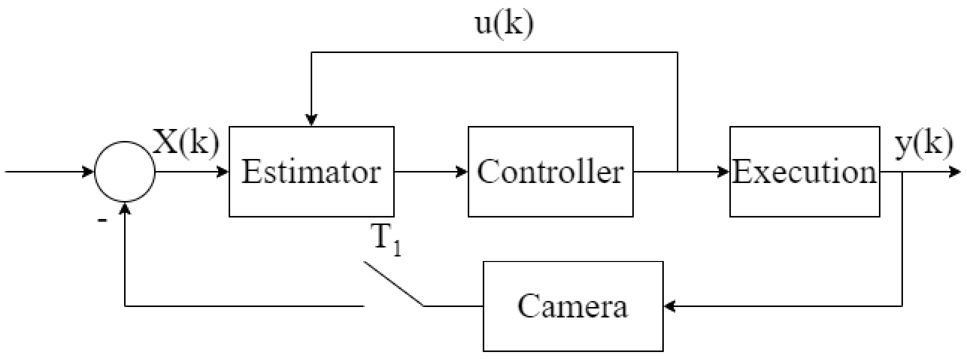

Figure 3-8: System Block Diagram with 1 Sensor

The measurements obtained from these sensors are subsequently fused using a data fusion method to generate a relatively accurate estimation of the robot's state. With this state estimation, the robot is capable of generating a new control input to sustain its motion towards the goal point. A new destination will be assigned to one robot if it is close enough to the original destination.

Chapter 4

Sensor Data Fusion Method

This chapter covers the sensor data fusion techniques. Section 4-1 introduces the concept of single-rate EKF, highlighting its significance in the realm of state estimation. Section 4-2 gives detailed information on the purpose of multi-rate EKF and presents the relevant equations and algorithm for multi-rate EKF. Lastly, section 4-3 illustrates different architectures of state estimation fusion. Furthermore, this section offers a brief evaluation of these fusion schemes

4-1 Single Rate Extended Kalman Filter

In real-world scenarios, the odometry of a robot can suffer from drift, leading to inaccuracies in localization. To address this issue, a global camera system is employed to provide additional information for correction purposes. However, it is important to note that both the odometry and camera measurements are not perfect and cannot be blindly trusted. Moreover, adding more sensors to the system would increase complexity. Hence, the application of sensor fusion techniques becomes crucial in order to combine the available sensor data and improve the overall localization accuracy.

The Kalman filter is a mathematical framework that offers an effective way to combine information from multiple sources, taking into account noise and inaccuracies, to estimate the true state of a system during transitions. It provides a recursive computational approach that iteratively updates the state estimate based on new measurements and the system's dynamic model.

The Kalman filter has been widely used in various fields, including robotics, tracking, and navigation, due to its ability to handle uncertain and noisy measurements. It has been extensively studied and applied in research papers such as [37], [38], and [39].

In this thesis, the state transition is a nonlinear function, as mentioned in equation 3-1, where the regular Kalman Filter is not suitable to solve. As a result, the extended Kalman Filter known as EKF [40] is used in this thesis. The EKF is well-suited for nonlinear system models and provides an effective means of estimating the state variables in such cases.

Generally, the EKF algorithm can be divided into two parts, the prediction part and the update part, based on [41]. In the prediction part, the algorithm predicts the next estimation and covariance via the system dynamics, later in the correction part, the predicted estimation and covariance are modified by integrating the sensor's measurement.

The calculation of the extended Kalman Filter is listed in the algorithm 1.

Algorithm 1 Single rate extended Kalman filter

Input: $\hat{x}(k+1 \mid k)$, $z(k+1)$

Output: $\hat{x}(k+1 \mid k+1)$

Prediction part

predicted state value

$$\hat{x}(k+1 \mid k) = f(X(k), u(k), w(k)) \tag{4-1}$$

predicted covariance of the state estimation

$$\boldsymbol{P}(k+1 \mid k) = \boldsymbol{F}(k+1)\boldsymbol{P}(k \mid k)\boldsymbol{F}(k+1)^\top + \boldsymbol{Q}(k+1) \tag{4-2}$$

Correction part

calculate the measurement residual between estimation and measurement $\boldsymbol{z}(k+1)$

$$\tilde{y}(k+1) = z(k+1) - h\left(\hat{x}(k+1 \mid k)\right) \tag{4-3}$$

computes the residual covariance

$$\boldsymbol{S}(k+1) = \boldsymbol{H}(k+1)\boldsymbol{P}(k+1 \mid k)\boldsymbol{H}(k+1)^\top + \boldsymbol{R}(k+1) \tag{4-4}$$

calculates the near-optimal Kalman gain

$$\boldsymbol{K}(k+1) = \boldsymbol{P}(k+1 \mid k)\boldsymbol{H}(k+1)^\top \boldsymbol{S}(k+1)^{-1} \tag{4-5}$$

updates the state estimation for the next step using the current estimation and measurements.

$$\hat{x}(k+1 \mid k+1) = \hat{x}(k+1 \mid k) + \boldsymbol{K}(k+1)\tilde{y}(k+1) \tag{4-6}$$

In this thesis, the state transition prediction function 4-1 can be further extended as equation 4-7 based on the system dynamics, where Δk is the time step length and $w(k)$ is the zero-mean white noise.

$$X(k+1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} + \begin{bmatrix} \cos\theta(k) * \Delta k & 0 \\ \sin\theta(k) * \Delta k & 0 \\ 0 & \Delta k \end{bmatrix} \begin{bmatrix} v(k) \\ \omega(k) \end{bmatrix} + \begin{bmatrix} w_1(k) \\ w_2(k) \\ w_3(k) \end{bmatrix} \tag{4-7}$$

In the predicted covariance equation 4-2, the $\boldsymbol{P}(k \mid k)$ represents the accuracy of the state estimation. This matrix has variances on the diagonal, and covariance on the off-diagonal position, the $\boldsymbol{P}$ matrix is initialized as some guess value. $\boldsymbol{F}(k+1)$ and its transpose $\boldsymbol{F}(k+1)^\top$ are both equivalent to A.

The Q matrix is the action uncertainty matrix, which is equivalence with the state transition noise in the state space model 3-2. Intuitively, a smaller Q value in the diagonal implies the real state has a larger difference compared to the estimated state value, which will cause the Kalman filter to "trust" state estimation more than the camera measurement.

The $\boldsymbol{R}(k+1)$ matrix in the equation 4-4 is the sensor measurement noise covariance matrix. As we assumed above, the measurement noise is assumed as a zero mean white noise and is uncorrelated to each other.

In the 4-6, the $\hat{x}(k+1 \mid k+1)$ is the final output of the extended Kalman filter algorithm, this value can be used in the P controller to generate new control input $u(k+1)$.

4-2 Multi-rate Kalman Filter Method

Multi-sensor fusion is a powerful technique, which elicits significant information from multiple sensors to acquire an optimal or sub-optimal state estimation [42], [8], [43], [44]. The multi-rate EKF was introduced to solve situations where sensors operate at different sampling rates. This approach allows for the fusion of measurements obtained from sensors with varying update rates.

In a traditional EKF, the sensor measurements are processed and fused at a single common update rate, which would typically be limited by the slower sensor. The multi-rate EKF allows measurements processed and fused at different rates, the estimation updates can occur more frequently based on the faster sensor's update rate, which causes a more up-to-date estimation.

In the multi-rate EKF, the covariance prediction for the slower sensor is different compared to the faster sensor in the single-rate EKF. In the case of the slower sensor, the time step between measurements is longer, resulting in a larger covariance prediction, because a longer time interval between measurements implies greater uncertainty in the system's evolution.

4-2-1 State Space Model for Slower Sensors

Before giving the algorithm of multi-rate EKF, the state space model need to modify. To express a model for ith sensor with a lower sampling rate, the state space model of the unicycle robot is extended to the steps of M augmentation in time step k. The augmentation can be expressed in the following iterative way, as shown in equation 4-8.

$$\begin{aligned} X(k+2) &= A(X(k+1)) + Bu(k+1) + w(k+1) \\ &= A(A(X(k)) + Bu(k) + w(k)) + Bu(k+1) + w(k+1) \\ &= A^2(X(k)) + \begin{bmatrix} AB & B \end{bmatrix} \begin{bmatrix} u(k) \\ u(k+1) \end{bmatrix} + \begin{bmatrix} A & I \end{bmatrix} \begin{bmatrix} w(k) \\ w(k+1) \end{bmatrix} \end{aligned} \tag{4-8}$$

where the M is the sampling period of ith sensor, the ith sensor will have measurement at time $k = 0, M, 2M, \cdots$.

We can further proceed with the above expression in an iterative way [8] and rewrite it in a compact form, then we have:

$$X(k+M) = A^M(X(k)) + B_M B u_M(k) + B_M w_M(k) \tag{4-9}$$

Where the notation A^M represents the matrix A raised to the power of M. And the B_M is the parameter matrix and $w_M(k)$ is called the noise matrix in one data block as shown in figure 3-1.

$$\begin{aligned} B_M &= \begin{bmatrix} A^{M-1} & \dots & I \end{bmatrix} \\ w_M(k) &= \begin{bmatrix} w(k) & \dots & w(k+M) \end{bmatrix}^T \end{aligned} \tag{4-10}$$

Then due to the assumption that the noise in the state transition is assumed as a zero mean white noise, the covariance matrix $Q_M(k)$ of the block noise $w_M(k)$ is driven as follows:

$$\begin{aligned} E\left\{w_M(k) w_M^T(m)\right\} &= \boldsymbol{Q}_M(k)\delta(k+m) \\ \boldsymbol{Q}_M(k) &= \begin{bmatrix} \boldsymbol{Q}(k+1) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \boldsymbol{Q}(k+M) \end{bmatrix} \end{aligned} \tag{4-11}$$

The term $\delta(k+m)$ represents a delta function, which is equal to 1 at the sampling time $k+M$ and zero otherwise.

4-2-2 Multi-rate EKF Algorithm

After we obtain the augmentation state space model, then the multi-rate Kalman filtering algorithm of ith sensor with a lower sampling rate can be implemented by substituting the covariance matrix $Q_M(k)$, parameter matrix $B_M(k)$ and A^M into the single-rate EKF, as presented in the algorithm 1. This process is outlined in Algorithm 2.

Algorithm 2 Multi-rate extended Kalman filter

Input: $\hat{x}(k+M \mid k)$, $z(k+M)$
Output: $\hat{x}(k+M \mid k+M)$

Prediction part
predicted state value

$$\hat{x}(k+M \mid k) = f(X(k), u(k), w(k)) \tag{4-12}$$

predicted covariance of the state estimation

$$\boldsymbol{P}(k+M \mid k) = \boldsymbol{F}(k+M)^M \boldsymbol{P}(k \mid k) \boldsymbol{F}(k+M)^{M^\top} + B_M \boldsymbol{Q}_M(k) B_M^T \tag{4-13}$$

Correction part
calculate the measurement residual between estimation and measurement $\boldsymbol{z}(k+1)$

$$\tilde{y}(k+M) = z(k+M) - h\left(\hat{x}(k+M \mid k)\right) \tag{4-14}$$

computes the residual covariance

$$\boldsymbol{S}(k+M) = \boldsymbol{H}_i \boldsymbol{P}(k+M \mid k) \boldsymbol{H}_i^\top + \boldsymbol{R}_i(k+M) \tag{4-15}$$

calculates the near-optimal Kalman gain

$$\boldsymbol{K}(k+M) = \boldsymbol{P}(k+M \mid k) \boldsymbol{H}_i^\top \boldsymbol{S}(k+M)^{-1} \tag{4-16}$$

updates the state estimation for the next step using the current estimation and measurements.

$$\hat{x}(k+M \mid k+M) = \hat{x}(k+M \mid k) + \boldsymbol{K}(k+M) \tilde{y}(k+M) \tag{4-17}$$

4-2-3 Dealing with Missing Measurements

In addition, it is possible for the ith sensor to encounter a missing measurement at time k, which is represented by a stochastic process $\gamma_i(k) \in 0, 1$. When $\gamma_i(k) = 0$, it indicates that sensor i does not provide a measurement at that particular time step.

In such a scenario, when a missing measurement occurs for a particular sensor, only the estimations made by other sensors can be considered reliable. As a result, the correction part of the Kalman filter algorithm will not proceed, and the Kalman gain $\boldsymbol{K}(k+M)$ associated with that sensor will be set to zero.

As a result, when a measurement is missing, the multi-rate Kalman filter output of that sensor will be obtained as follows:

$$\begin{aligned}
\hat{x}(k+M \mid k+M) &= \hat{x}(k+M \mid k) = f(X(k), u(k), w(k)) \\
\boldsymbol{P}(k+M \mid k+M) &= \boldsymbol{P}(k+M \mid k) \\
&= \boldsymbol{F}(k+M)^M \boldsymbol{P}(k \mid k) \boldsymbol{F}(k+M)^{M^\top} + B_M \boldsymbol{Q}_M(k) B_M^T
\end{aligned} \tag{4-18}$$

4-3 State Estimation Fusion Architectures

When working with multiple sensor sources, the choice of data fusion architecture becomes crucial. The manner in which data from each sensor is combined can have a profound impact on the overall performance of the system.

Through the literature review, the cascade style architecture and Ordered Weighted Averaging (OWA) architecture is found suitable and easily implemented. Other architectures, such as fuzzy logic or neural networks, require additional considerations. For instance, fuzzy logic architecture demands the careful design of fuzzy rules, while neural network approaches necessitate substantial amounts of training data, which may not be necessary in this particular case. Hence, this section focuses on discussing the cascade and OWA style architectures, which offer effective and practical solutions for multi-sensor data fusion.

Based on the above discussion, we have two sensors, including odometry and camera, and we will have methods to fuse the multiple sensors' data and construct the localization architecture.

In the observation function (equation 3-2) we assign $i = 1$ for the odometry and $i = 2$ for the camera.

We can express the Kalman filter estimation in the following equation:

$$\hat{x_1}(k+1 \mid k+1) = g_1(\hat{X}(k), h_1(X(k))), \theta_1(k) = 0 \tag{4-19}$$

$$\hat{x_2}(k+1 \mid k+1) = g_2(\hat{X}(k), h_2(X(k))), \theta_2(k) = 0 \tag{4-20}$$

where $\hat{x_1}$ and $\hat{x_2}$ is obtained by odometry and camera respectively.

4-3-1 Single-rate Extended Kalman Filter Method with Cascade Style Architecture

In this scenario, two single-rate EKF (Extended Kalman Filter) algorithms are employed. Both the odometry and system dynamics have the same sampling rate, allowing for the fusion of initial estimation with odometry data, resulting in an intermediate estimation. The workflow of this process is depicted in Figure 4-1.

If the camera information is available at the current time step k, then the final estimation is produced by fusing the camera information and intermediate estimation. However, in cases where the camera information is not available, the intermediate estimation serves as the final output of the fusion module.

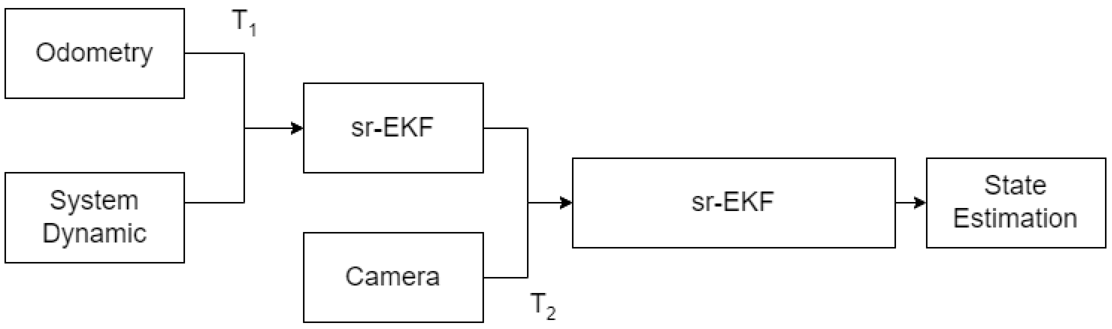

Figure 4-1: Single-rate EKF Localization Workflow

4-3-2 Single-rate Extended Kalman Filter Method with OWA Style Architecture

In papers, [8] and [45], Kalman filter outputs combining different sensors value are integrated together in an OWA fashion. In paper [8], the author elaborates on the benefit of using an OWA integration rather than a normal Kalman filter. The workflow is shown in figure 4-2.

At the time k, the weight of the OWA architecture for ith sensors $W_{i,k}$ is determined following 4-21.

$$\begin{aligned} r_i(k) &= y_i(k) - H_i(\hat{(}x(k \mid k)) \\ \hat{C}_i^{-1}(k) &= \frac{1}{l_w}\sum_{i=1}^{l_w} r_i(k+1-i)r_i(k+1-i)^T \\ W_{i,k} &= \hat{C}_i^{-1}(k \mid k)(\sum_{j=1}^{N} \hat{C}_j^{-1}(k \mid k))^{-1} \end{aligned} \tag{4-21}$$

The $r_i(k)$ is the innovation matrix for ith sensor, it is introduced as a criterion for predicting the nonlinearities and uncertainties caused by unmodeled dynamics, uncertainty, and missing measurements in the system.

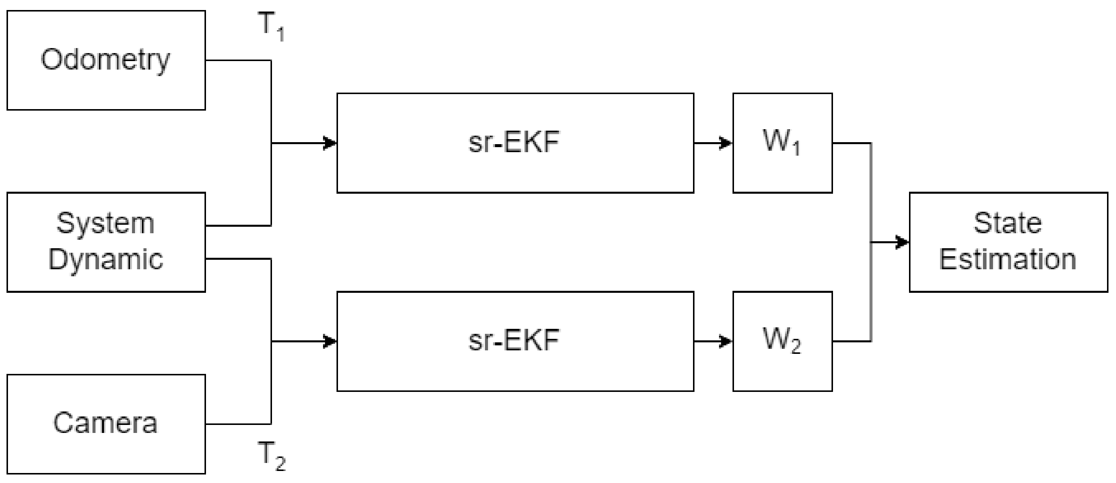

Figure 4-2: Single-rate EKF with OWA Localization Workflow

And the $\hat{C}_i(k)$ implies the real state estimation covariance during the mr-EKF fusing process, it can be computed based on the innovation matrix. Then the l_w is the length of sliding windows, which could be the sampling period of sensor i.

If the Optitrack system measurement is present at the current time, the state estimation combing different fused information can be expressed in the following way.

$$\hat{x}(k \mid k) = \sum_{i=1}^{N} W_{i,k}\hat{x}(k \mid k) \tag{4-22}$$

4-3-3 Multi-rate Extended Kalman Filter Method with Cascade Style Architecture

Another architecture that unitizes the Multi-rate Extended Kalman filter method shares the same idea with the previous case, the workflow is shown in figure 4-3. The only difference is the fusing between camera information and intermediate estimation: In the case of using an mr-EKF, the sensor measurements would be processed at their respective rates, resulting in two separate Kalman filters. One filter would be used for the fast sensor and the other for the slow sensor, each with different time steps. When the camera information is ready, the output of these filters would be fused together obeying cascading style.

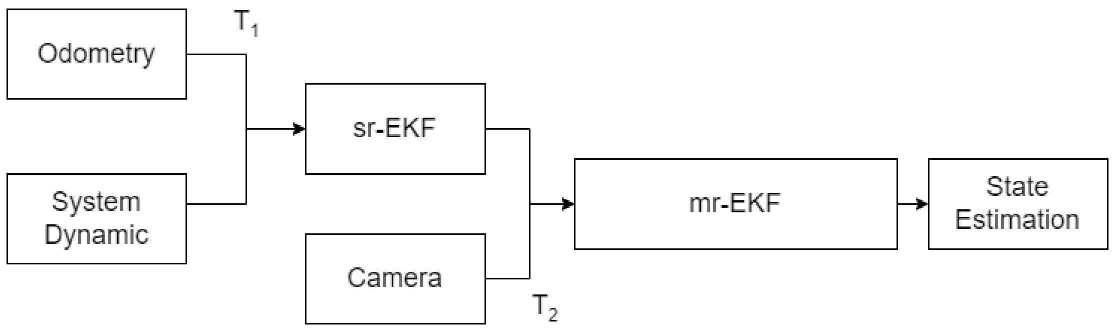

Figure 4-3: Multi-rate EKF Localization Workflow

4-3-4 Multi-rate Extended Kalman Filter Method with OWA Style Architecture

The case where the mr-EKF is applied in an OWA-style architecture is similar to the previous case, the workflow is presented in figure 4-4.

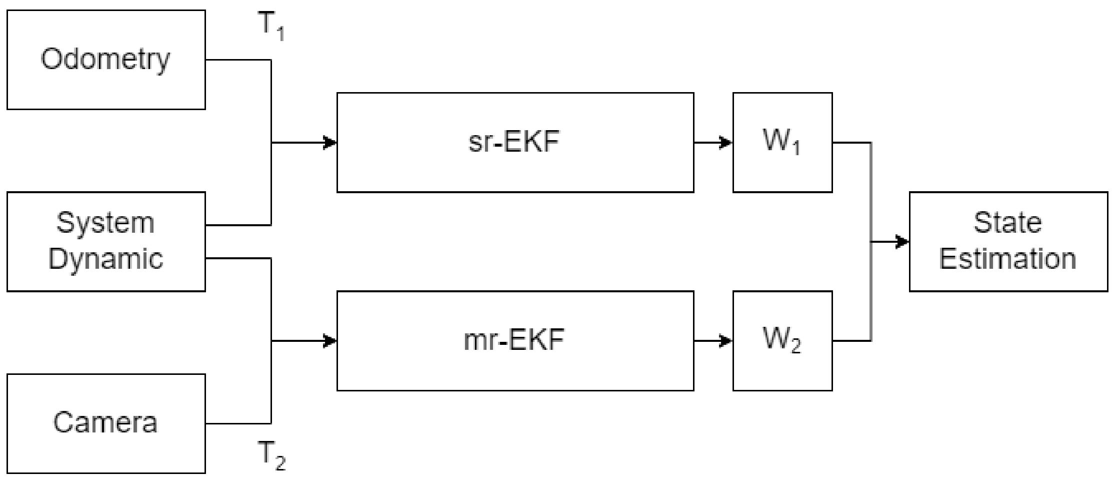

Figure 4-4: Multi-rate EKF with OWA Localization Workflow

4-3-5 Methods Evaluation

The mr-EKF may outperform the sr-EKF. If the measurement rates of the sensors differ significantly, using an mr-EKF may be advantageous because it allows the system to update the state estimate at a higher rate with the fast sensor measurements, while incorporating the slower measurements from the other sensor as they become available. In contrast, an sr-EKF would update the state estimate at the slower rate of the second sensor, potentially leading to less accurate results.

The OWA-style architecture might outperform the cascade-style one. Because the odometry has a drift, the effect of the drift may accumulate over time and affect the accuracy of the state estimate, with the proposed OWA style, a time-varying weighted average method can be a better choice than a cascade method if one sensor is known to be less accurate over time, as it allows the weights to be adjusted to account for the changing reliability of the sensors.

However, there may be some cases where a cascade method is still preferred because the cascade method is simpler to implement and requires fewer computational resources than the OWA style. If in the real implementation, the odometry can be reset frequently in order to eliminate its drift, these two architectures won't have too much difference.

Chapter 5

Simulation

The upcoming chapter focuses on simulations conducted to evaluate the performance of different data fusion methods before applying them to real robots. Section 5-1 introduces the basic setup of the simulation, and also describes how the simulation mimics perturbations in the system. Section 5-2 discusses the simulation's detailed parameter settings and explains the choices made. The overall simulation workflow is given in this section. The simulation results are presented in section 5-3, and conclusions are drawn in section 5-4.

5-1 Simulation Setup

In this section, basic information about the simulation is provided. Several setups and assumptions are made for further simulation to mimic the imperfection that happens both in the state transition process and state measurements.

5-1-1 Basic Setup

The simulation is implemented using Python and Pygame. The dimensions of the experiment playground are set to 1080 pixels in width and 640 pixels in height. The moving robots are represented by red triangles and circles, with a diameter of 15 pixels. The unique destination points of each robot are depicted in green, with a diameter of 8 pixels. Figure 5-1 presents a screenshot of the simulation, offering an overview of the setup.

In the simulation, a group of ten robots is released and positioned according to the specific locations outlined in Table 5-1. These initial positions are chosen to ensure that the robots are positioned far apart from each other. Moreover, a set of destination points is initialized

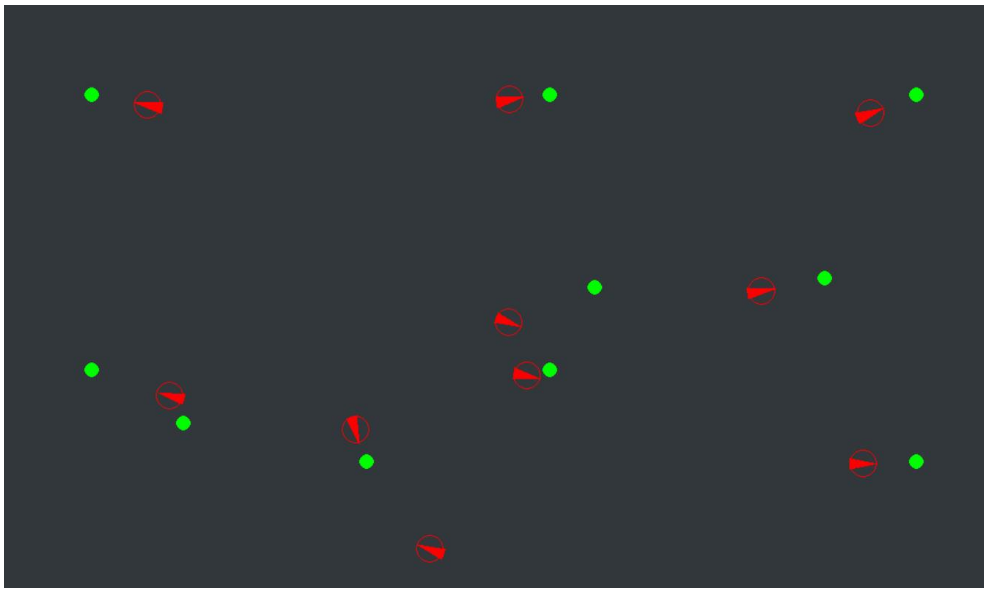

Figure 5-1: Simulation Overview with 10 Robots

Robot	1	2	3	4	5	6	7	8	9	10
x	100	100	100	200	200	300	400	500	700	900
y	100	200	400	50	300	500	300	400	100	400

Table 5-1: Simulation Position Setup

based on the specifications in Table 5-2. The heading of each robot is set to 0 degrees, indicating that they are initially oriented towards the right edge of the experiment field.

During the simulation, as a robot approaches its assigned endpoint and reaches a distance of less than 30 pixels, a new endpoint will be generated randomly and assigned to the robots. The simulation itself will continue until the number of Pygame frames exceeds 10,000 frames.

We are assuming the Odometry of each robot is working at a rate T_1, and robots are receiving measurements from the Optitrack system with a certain sampling period T_2, where $T_2 = 5T_1 = 5(\text{frames})$.

5-1-2 Perturbation

As described in the previous section 3-3, the kinematic model has a noise term. This perturbation is represented by a set of zero mean Gaussian white noise $w(k)$ in the 3-2, their covariance was assumed as 0.01, 0.01 and 0.0005.

Therefore, in the simulation, a random number satisfying the above assumption was added to the state vector. Then the Odometry on one robot would record that biased state vector as the odometry measurement.

destination	1	2	3	4	5	6	7	8	9	10
x	600	600	900	400	600	1000	1000	600	100	100
y	400	100	300	500	400	500	100	600	100	400

Table 5-2: Simulation destination Setup

Besides the Odometry, another source of information is the camera.

The measurement perturbation can happen in two places, noise and merge conditions. As described in figure 3-6, while the robot is standing still, the position readings from the camera have turbulence due to the noise on the Optitrack system, as a result in the sensing part of the simulation, a random number v following Gaussian distribution was added to the camera measurement, where $v \sim \mathcal{N}(0,\, 0.005)$.

In the simulation, to simulate the merging condition, the below assumptions were made. If the distance between the moving robot and other agents is small enough, e.g. less than 50 pixels, the camera will record the midpoint of all nearby robots' locations as the position measurement. Meanwhile, the camera has a dropping rate γ to discard some measurement readings, and we set $\gamma = 0.05$.

5-2 Estimation Method

In this section, the parameter selection for EKF is discussed. Since the robot can't directly get its camera measurement, methods for identifying correct camera measurements and handling outlier data are presented. The overview of the simulation flow of each robot is provided at the end.

5-2-1 Parameter Selection

Since the robot has two sources of information, two Kalman filters are required. The main parameter in both sr-EKF and mr-EKF are the predicted covariance $\boldsymbol{P}(k \mid k)$, action uncertainty matrix $Q(k)$ and sensor measurement noise covariance $R(k)$ in the equation 4-2 and 4-4.

For both Kalman filters, as shown in the equation 5-1, the predicted covariance matrix $\boldsymbol{P}(k \mid k)$ is initialized as some guessed value, the matrix $\boldsymbol{F}(k)$ is initialized as state transition matrix A, and it remains constant along the run time.

$$\boldsymbol{P}(k \mid k) = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} ,\boldsymbol{F}(k) = \boldsymbol{F}(k)^\top = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5-1}$$

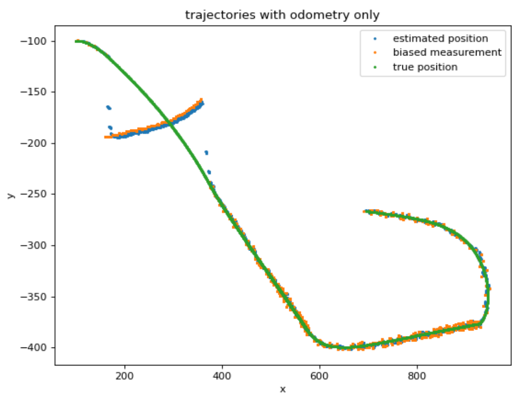

Figure 5-2: Trajectories with Q = 1

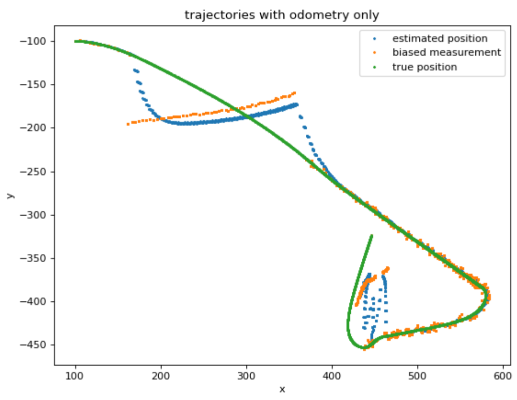

Figure 5-3: Trajectories with Q = 0.05

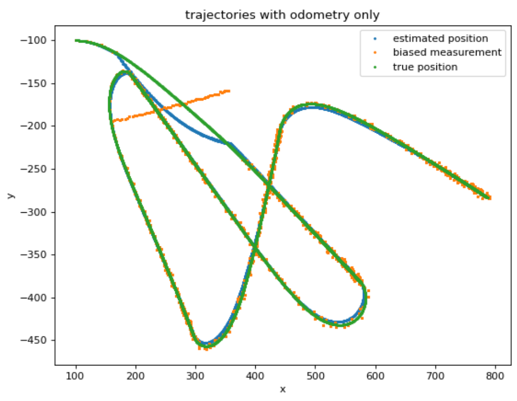

Figure 5-4: Trajectories with Q = 0.0005

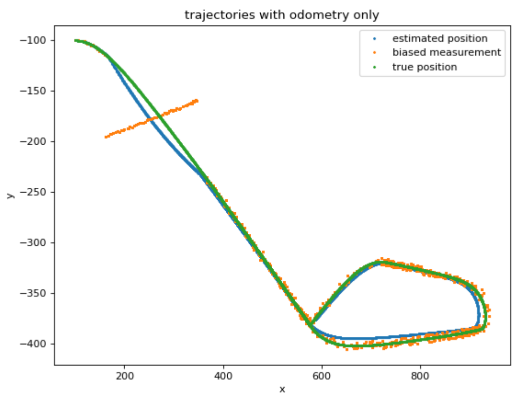

Figure 5-5: Trajectories with Q = 0.00005

The action uncertainty matrix $Q(k)$ has an impact on the behaviour of the Kalman filter. Some tests were conducted to test the effect of different Q matrices.

Three robots were released in these tests. For simplicity, only one trajectory was shown here. The green dots represent the real position, the orange dots stand for the camera measurement, and the blue dots are the estimated position. Only a single-rate Kalman filter was applied in these tests.

From the figure 5-2 to 5-5, as the value in Q becomes smaller, the blue dots (estimation) are getting closer to the green dots(real position), and the biased camera measurement has a smaller impact on the estimation. Therefore, larger Q will cause the robot to trust the camera more than its own estimation, so the estimation will be misled when the camera information is wrong and thus have a large difference compared to the true position.

As a result, for the Odometry, since the Odometry would have drift, and the drift grows larger as time goes by, we won't want the robot "trust" the Odometry too much, so the noise matrix was initialized as the identity matrix. And the value in the action uncertainty matrix is the same as the covariance value that happened in the state transition, which was assumed as 0.01, 0.01 and 0.0005.

Then on the odometry side, the action uncertainty matrix and the noise covariance matrix were initialized as below:

$$\boldsymbol{R}_1(k) = \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & 1.0 & 0 \\ 0 & 0 & 1.0 \end{bmatrix}, Q_1(K) = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.005 \end{bmatrix} \tag{5-2}$$

For the camera, as discussed in the previous subsection, although the noise while robots are standing still is negligible, the camera measurement could have a large deviation when merging or missing condition occurs. Therefore, the noise covariance on the camera side wouldn't be too small and was assumed as $v(k) = [0.3, 0.3, 1.0]$. The Q matrix remains the same as the one with Odometry.

On the camera side, the action uncertainty matrix and the noise covariance matrix were initialized as below:

$$\boldsymbol{R}_2(k) = \begin{bmatrix} 0.3 & 0 & 0 \\ 0 & 0.3 & 0 \\ 0 & 0 & 1.0 \end{bmatrix}, Q_2(K) = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.005 \end{bmatrix} \tag{5-3}$$

5-2-2 Identify the Position from the Camera Outputs

The Optitrack System can track objects and sense their positions in the real application, despite the merging condition. However, this system can only give a list of all measurements (note that the number of records could be less than the number of robots because of the merging condition), meaning that robot can't identify which reading is the right measurement for itself.

In that case, each robot needs to find out which record is the correct position record and then take the specific records as its measurement.

Since all robots know their own initial position, and we want the state estimation on the last time step to be close enough to the ground true position, the robot would use the reading which is closest to its last estimation as the camera measurement.

We assign $H(k) = \{(x_0, y_0)_k, \cdots, (x_i, y_i)_k\}, i \in \{0, 1, \cdots, n\}$ as the camera output, where n is the total number of robots when merging happens, the number of output is less than the number of robots. We also assign $\hat{X}(k) = [\hat{x}(k), \hat{y}(k)]$ as the estimation coordination on the current time step, and $X(k) = [x(k), y(k)]$ as the assigned camera measurement at the current time step k.

Then the measurement $X(k)$ can be determined in the following way:

Algorithm 3 Identify position knowing initial position

```
Input: $\hat{x}(k)$, $H(k)$
Output: $X(k)$
  for $(x_i, y_i)_K$ in $H(k)$ do
      compute distance between $\hat{X}(k)$ and $(x_i, y_i)_k$
      find the minimal distance and corresponding camera measurement $(x_j, y_j)_k$
  end for
  $X(k) \leftarrow (x_j, y_j)_k$
```

To validate the above merging condition simulation and position assignment, the comparison between estimation (only using camera measurement) and true position was plotted in figure 5-6 and 5-7.

The above content indicates that when the robot knows its initial position, the method can help the robot to find the correct measurement among all the possible options given by the camera during the run time.

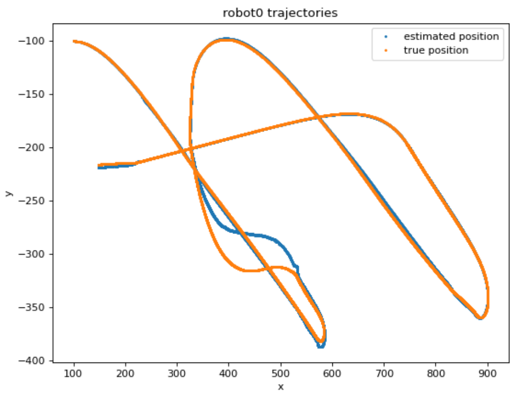

Figure 5-6: Trajectories of Robot 0

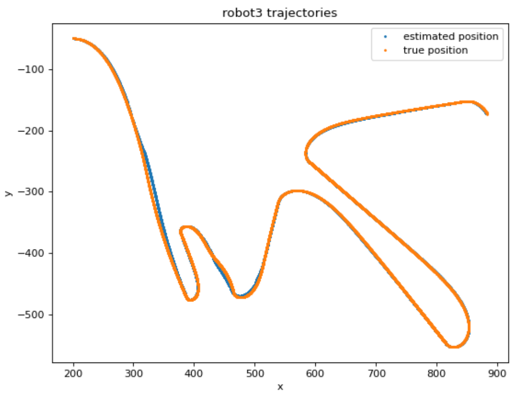

Figure 5-7: Trajectories of Robot 3

5-2-3 Dealing with Deviated Measurement

Moreover, the camera will lose some of the robots' positions even if no other robots are near them. This might happen when the connection quality is low or affected by natural light. In this case, all the possible positions given by the Optitrack system are far from the true position and estimation. Assigning a wrong position will cause the robot loses its position because the robot will make a wrong estimation based on a wrong measurement, and then use the wrong estimation to identify the next measurement.

A good way to avoid that is the following when all the possible positions are far enough, the robot will drop the camera information and localize itself only using the open loop estimation based on the fused estimation combining the kinematic model described in the equation 3-1 and odometry measurement.

5-2-4 Single Robot Simulation Flow

Based on the system block diagram 3-7, the single robot simulation process can be described in an iterative way using the following pseudo-code.

Algorithm 4 Single robots simulation workflow

```
Input: $X(0)$
Output: $\hat{X}(k)$, $H(k)$
  $k \leftarrow 1$ , initialize robots with $X(0)$
  while running do
      update $v(k), \omega(k)$ based on 3-4, and perturb the control input
      if $\theta(k) = 1$ then
          receive $H_k$, and identify position $h(X_k)$ from camera
          update state with estimation via equation 4-19
      else if $\theta(k) = 0$ then
          update state with estimation via equation 4-20
      end if
      if reaches destination then
          set a new destination
      end if
      $k \leftarrow k + 1$
  end while
  log the data
```

5-3 Simulation Results

Based on the above simulation setup and above pseudo code, several experiments with ten robots (as shown in the figure 5-1) were conducted. The simulation proceeded in a Monte Carlo style, and for simplicity, only the trajectory of some robots (mainly the No.2 robot) in the last rounds was shown.

5-3-1 Odometry only

The case where the estimation only depends on Odometry was tested first, the result is shown in figure 5-8.

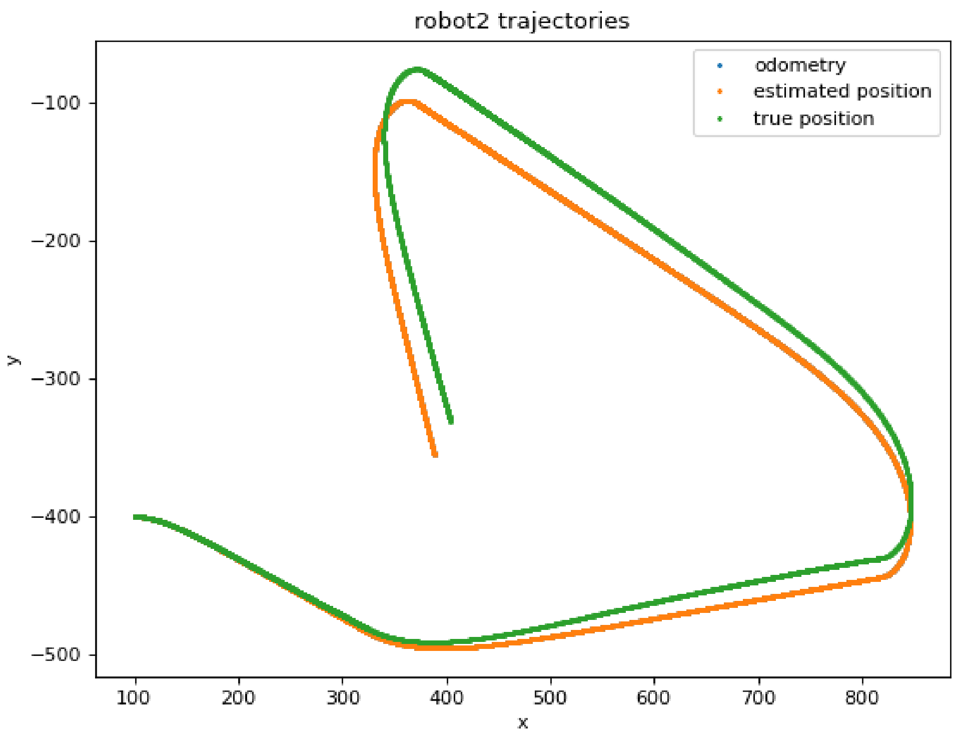

Figure 5-8: Trajectories of robot No.3 only with Odometry

The No.2 robot was initialized at the bottom left part of the picture. The result shows that if the robot only depends on the biased Odometry, the position estimation has a large difference from the true trajectory. Since the open loop estimation uses an iterative way, the error accumulated over time goes. Therefore, additional sensors are needed to correct the biased estimation.

5-3-2 Cascade sr-EKF

If the robot uses the cascade style, as shown in figure 4-3 the experimental result is shown in figure 5-9.

Since all the destination points were randomly generated, the robots swarm might become too cowardly or become sparse in some rounds. As a result, in figure 5-9, the trajectory was strongly affected by biased camera measurements.

As evaluated before, the result of the cascade style might become unpleasant because the estimation was first corrected by the Odometry and then by the camera. In that case, if the Odometry has a large drift, the first estimation will be biased, and this will degrade the accuracy of the combined estimation. For example, even though the camera measurement

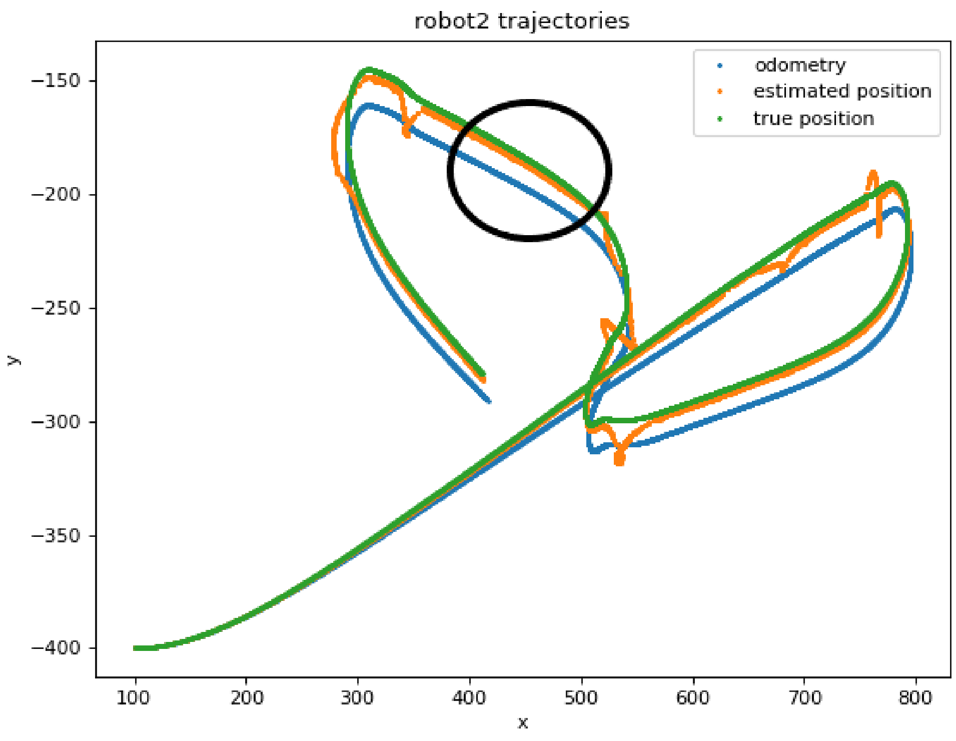

Figure 5-9: Trajectories of robot No.2 with cascade sr-EKF

is quite accurate in the black circle, the final estimation is deviated due to the biased Odometry.

5-3-3 OWA sr-EKF

If the robot uses the OWA style, as shown in figure 4-4, the experiment result is shown in figure 5-10.

The small rising on the orange curve is caused by the merging condition. With the OWA style, the final performance seems to be better than using a cascade style, since it uses a weight to evaluate the reliability of these two sources. From the picture 5-10, the estimation can track the true position well, even with some disturbance.

5-3-4 Cascade mr-EKF

If the robot uses the mr-EKF in a cascade style, the experiment result is shown in figure 5-11.

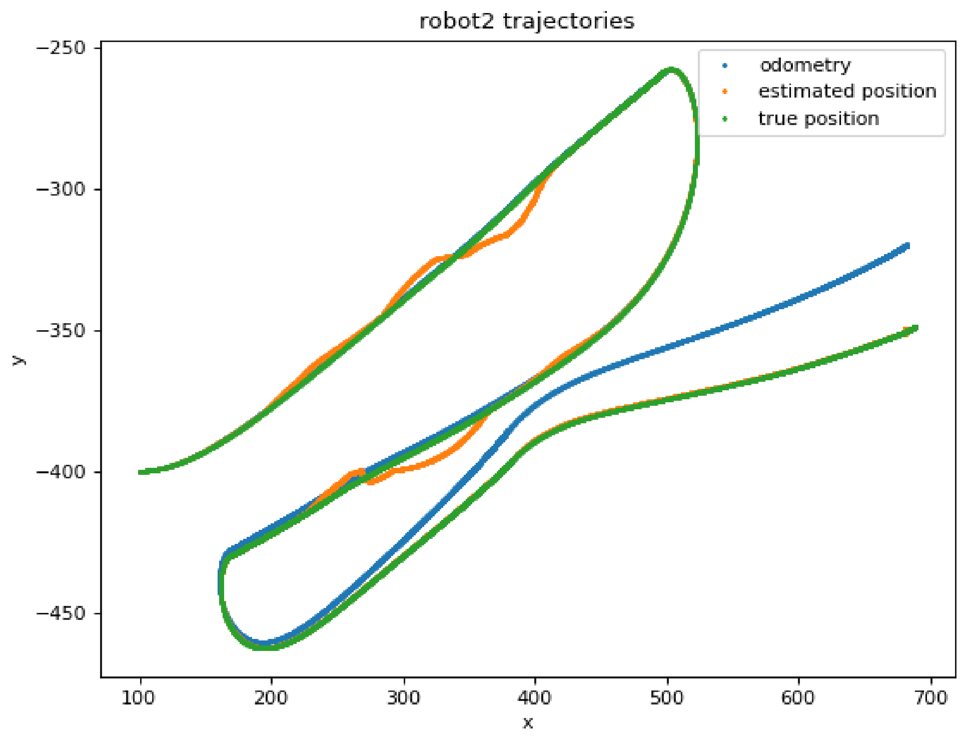

Figure 5-10: Trajectories of robot No.2 with OWA sr-EKF

Because the No.2 robot occurred small biased Odometry and quite an accurate camera in this round, the result from another robot is shown, the no. 6 robot was initialized at the top left of the field.

From the figure 5-11, despite the fact that the final estimation can follow the true position when both sensors give reliable information, the performance is not well when the camera is accurate while the Odometry has large bias.

As highlighted in the black circle, the Odometry (blue) has a large bias compared to the ground true position (green). This could drag the estimation away from the true position, and applying mr-EKF couldn't eliminate that effect very well.

5-3-5 OWA mr-EKF

If the robot uses the mr-EKF in an OWA style, the experimental result is shown in the figure 5-12.

Under this architecture, the estimation result is relatively close to the true position in this round. For the merging condition (orange raising), the estimation won't be affected too much by it.

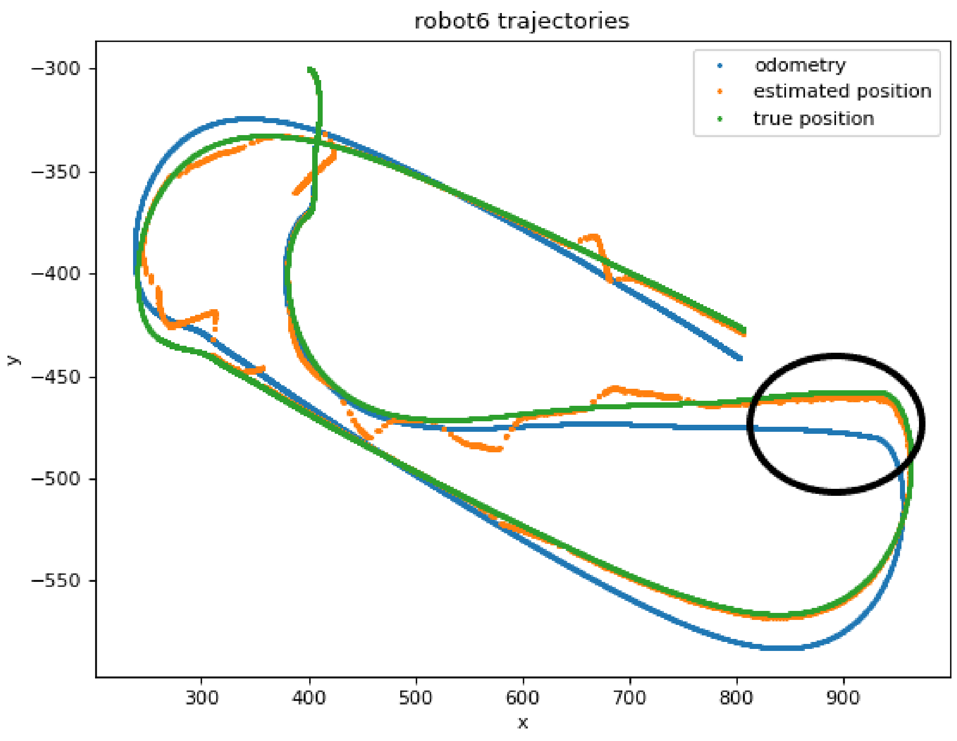

Figure 5-11: Trajectories of robot No.6 with cascade mr-EKF

Also, for the drifted Odometry, the estimation could also follow the right location. This could give a raw statement that with the OWA multi-rate EKF, the localization system can overcome this disturbance and track the true position better.

More examples can be viewed in figure 5-13, the estimation is close to the real points, and most of the bias happened when the camera is biased, and if we restrict the camera data discard rule which is the robot only accepts those camera measurements close enough to its state estimation, the effect of wrong camera measurement will be reduced, and total localization accuracy will increase.

5-3-6 Mean Square Error Comparison and Discussion

Mean Square Error Comparison

To check which method is reliable, the mean square error(MSE), as shown in the equation 5-4, is often used as a metric to indicate performance improvement. It can show how well the above localization architecture can improve localization performance.

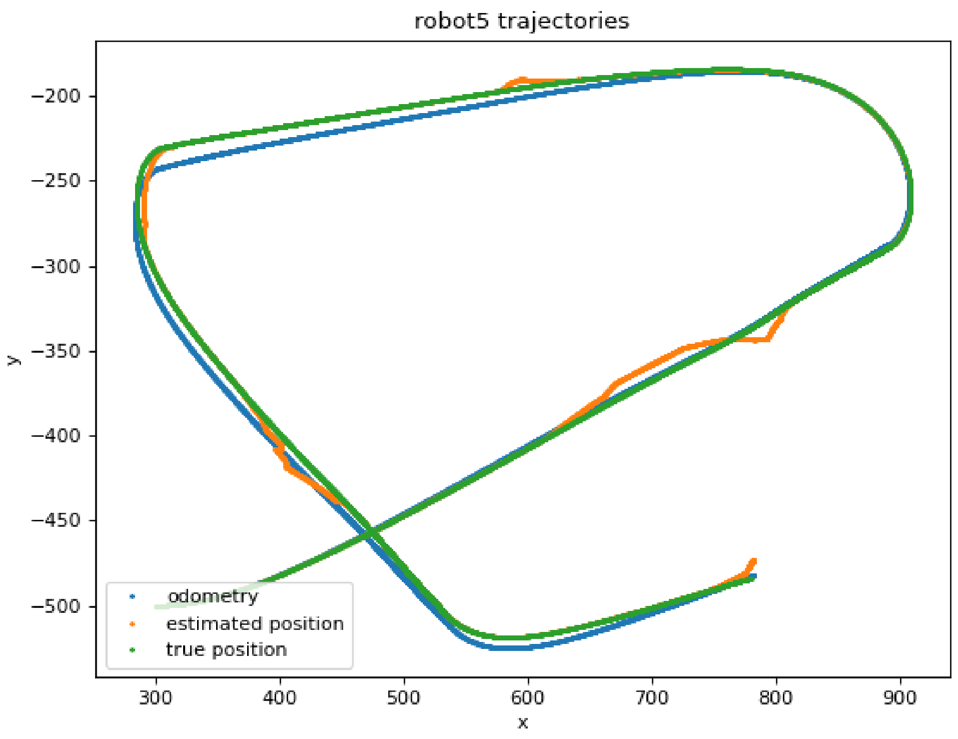

Figure 5-12: Trajectories of robot No.5 with OWA mr-EKF

For each round, the average MSE was computed under different methods and architectures, the result is shown in the table 5-3.

$$\mathrm{MSE} = \frac{1}{l_w}\sum_{i}^{l_w} = 1(\hat{x}_i - x_i)^2 \tag{5-4}$$

round	1	2	3	4	5	6	7	8	9	10
odometry	82.134	80.727	60.76	65.333	81.47	81.442	73.525	94.816	64.463	75.465
cascade sr-EKF	27.255	39.504	30.509	28.273	35.525	36.163	37.831	30.045	32.721	37.009
OWA sr-EKF	33.052	28.716	27.68	24.549	23.245	23.259	24.675	37.025	27.911	19.106
cascade mr-EKF	45.329	28.137	45.025	46.085	40.93	52.13	58.572	33.651	41.407	41.865
OWA mr-EKF	18.962	23.018	23.366	22.92	21.854	22.77	24.331	26.366	13.951	21.723

Table 5-3: MSE Result Comparison

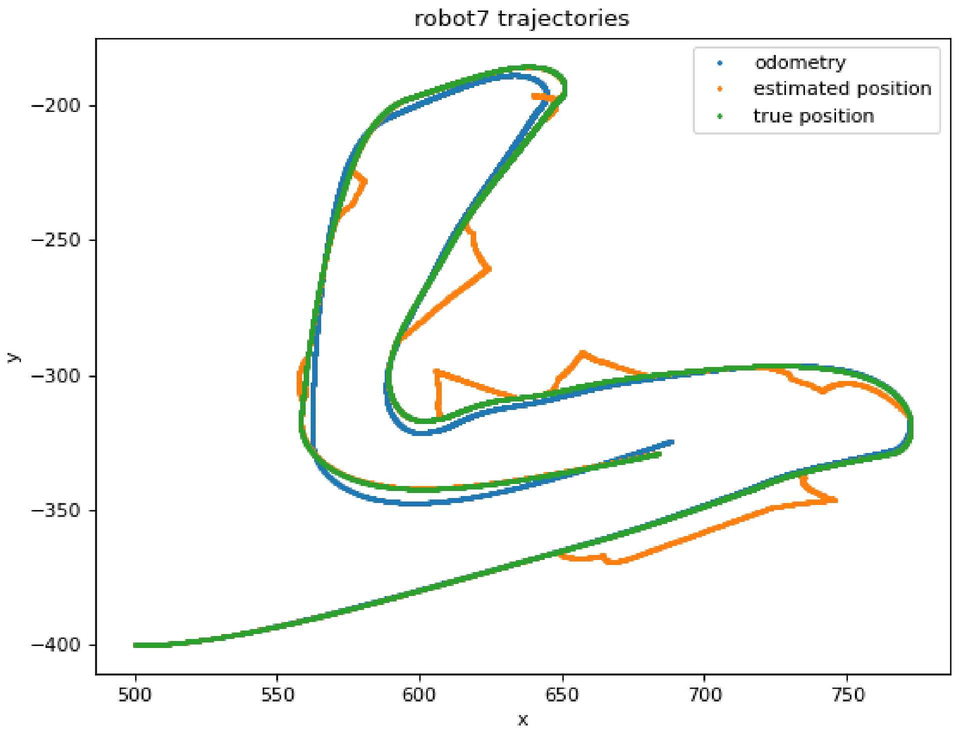

Figure 5-13: Trajectories of robot No.7 with OWA mr-EKF

Then for each method, the total average MSE was computed, and the result is shown in the table 5-4.

	odometry	cascade sr-EKF	OWA sr-EKF	cascade mr-EKF	OWA mr-EKF
MSE	76.013	33.484	26.921	43.313	21.926

Table 5-4: Average MSE Result Comparison

5-4 Result Discussion and Simulation Conclusion

Result Discussion

From the table 5-3, we can see a rough trend that the error between estimation and true position could be reduced with the data fusion method. Moreover, the MSE result with the OWA style seems smaller than the result with the cascade style, showing the OWA style outperforms the cascade style.

In each row, data are relatively close to each other, but some biased data also exist. For example, in the last row, with the OWA multi-rate EKF, in the first round, the MSE is larger than the other. This is because of the random processing noise and random movement, when lots of robots are gathering, the only reliable source is Odometry, and the Odometry is drifted. In that case, outliers might show up.

From the table 5-4, we can say that using the data fusion method does improve localization performance. The MSE result with the OWA style is smaller than using other methods, showing that using the OWA style is better than the cascade style.

Surprisingly, the multi-rate method doesn't improve the performance greatly compared to the improvement caused by using OWA style architecture. The reason for it could be that the camera information is quite nice most of the time except for merging and data-dropping conditions, as a result, the measurement could correct the estimation even using the simple version of the Kalman filter.

The case where the right camera measurement may not correct the estimation will mainly happen when the original estimation is already biased or "corrupted" by the biased odometry information. Since the Q and R matrix will remain the same during the run time, the correction ability of one Kalman filter won't change a lot, and as the drift on the Odometry becomes larger, the final estimation will deviate further than the true location if we still take the result produced by Odometry into account too much.

While using the OWA style architecture, according to the equation 4-21, the weight will change during the run time. If one sensor has a larger deviation than the estimation during a period of time, its weight will decrease. By doing so, the effect caused by increasingly biased Odometry can be eliminated a bit.

Simulation Conclusion

Some statements and conclusions can be given from the simulation result shown in the above content.

1. Compared to using Odometry only, using the data fusion method could improve the localization performance by at least 50%.
2. Using multi-rate Extended Kalman Filter seems not to improve the localization much.
3. The cascade style architecture will work well if the Odometry has a small drift.
4. The OWA style architecture outperforms the cascade style one. The simulation results show that it can increase the localization accuracy by around 34%. Therefore, it can be applied to a real platform.

Chapter 6

Experimental Results

This chapter will cover the experiment setup, the result of the localization and the discussion of the experimental discussion. Firstly, section 6-1 provides a detailed description of the necessary steps and procedures before applying the localization algorithm to the robots. Secondly, the comparison of experimental results under different parameters selection is discussed in the section 6-2. Then the localization system is applied to a control task, in section 6-3, the introduction of the control task is provided, and the experimental results are also presented. Later, summaries and conclusions are listed in section 6-4.

6-1 Experimental Setup

This section provides a brief introduction to the Elisa-3 robots. Additionally, the report discusses several improvements made to reduce sensor noise before generating estimations. These improvements include the addition of low-pass filters, the incorporation of more sensor data, and the implementation of hardware upgrades to enhance the capture quality of the Optitrack camera.

6-1-1 Elisa-3 robot

We utilized Elisa-3 robots (shown in Figure 6-1 and 6-2) as our platform for verifying localization performance, robots' details can be found at https://www.gctronic.com/doc/index.php/Elisa-3. These robots are equipped with various sensors, including IR emitters for the global camera, IR proximity sensors for obstacle avoidance, encoder odometry, and a 3-axis accelerometer. However, they do not possess a global orientation measure using a compass or gyroscope. Instead, angle estimation is achieved through the use of encoder odometry.

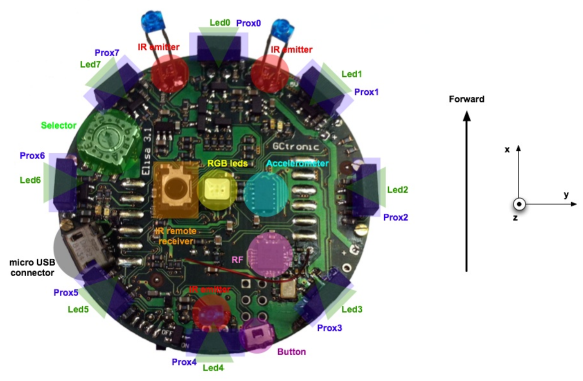

Figure 6-1: Elisa-3 Robot Hardware

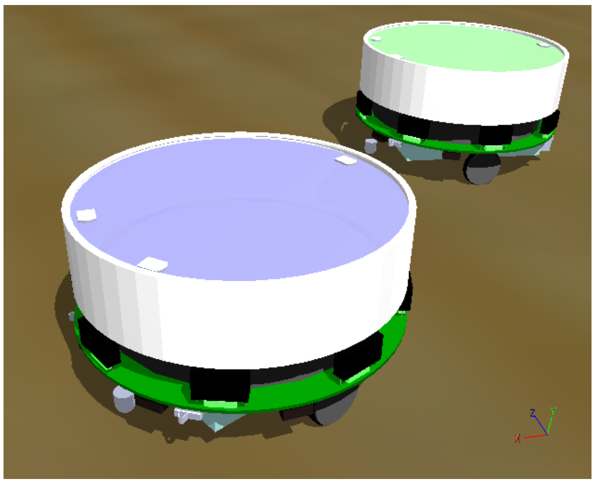

Figure 6-2: Elisa-3 Robot in Webot

The swarm of Elisa-3 robots can be organized via the ROS, the schema between robots and ROS is shown in figure 6-3. Each robot is controlled by the Elisa-3 ROS node, which organizes the robots into a swarm. The robots communicate wirelessly with the ROS node through the radio module (nRF24L01+), transferring data to and from the ROS node. The swarm shares information, such as odometry data, with the controller via ROS topics.

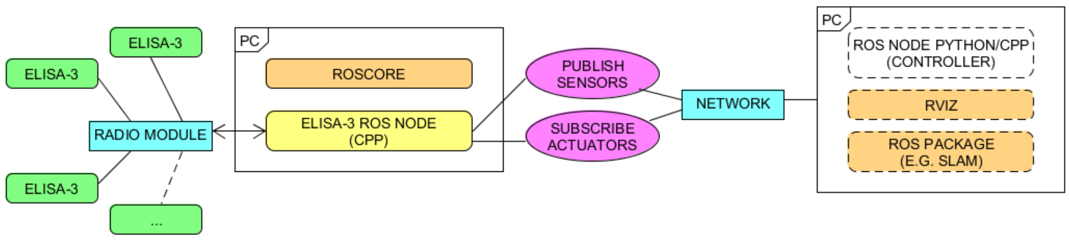

Figure 6-3: Elisa-3 Robots ROS Schema

6-1-2 Data Polishing

During the experiments, it was discovered that the onboard odometry encoder was not always precise. The possible reasons for this and their effects on the sensors' measurements are as follows:

- Communication problem

  Due to communication issues, time intervals between each odometry reading were found to be not roughly equal and have a large variation. The radio module shown in Figure 6-3 was unstable at times, leading to the odometry data freezing and becoming stuck while the robot was still moving. For example, in Figure 6-5, the blue trajectory shows that the gap between each odometry sample is different. At certain points, the robot was unable to update the newest odometry reading via the radio module, and the PC continued to receive outdated measurements. This discontinuity and outdated measurement can mislead the later position estimation with the camera measurement.

- Wheel slipping

  The odometry is based on an encoder that counts and sums up the velocity measurements stored in the analogue-to-digital converter, which is then converted into distance. However, the encoder odometry ignores factors such as wheel slipping, resulting in even more biased odometry than the actual situation, in addition to accumulated errors.

The odometry has many imperfections, however, an accurate onboard sensor reading is still needed especially when the camera data is not available. As a result, to compensate for the imperfection and have smoothed data, applying a fusion subsystem considering the accelerometer's data and incorporating a low-pass filter is necessary.

Fusion with Accelerometer's data

To overcome the imperfections of odometry mentioned above, it is necessary to include another sensor source [46]. Elisa-3 robots are equipped with a 3-axis accelerometer (Freescale MMA7455L), allowing for the use of additional sensors. But the robot can only move in one direction(the x-axis direction) and there is no gyroscope available. As a result, the only suitable measurement is the acceleration on the x-axis a_x. Similar to the approach taken in [47], a fusion subsystem between the accelerometer and encoder is shown in Figure 6-4.

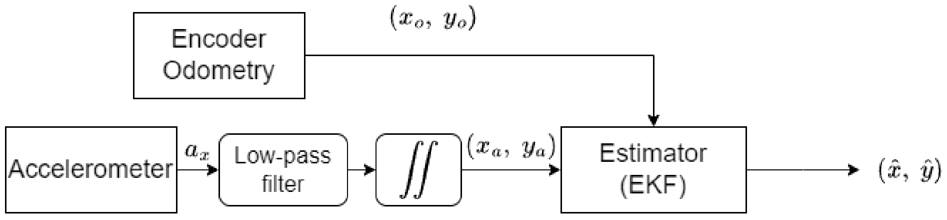

Figure 6-4: Sensor Fusion System (Accelerometer and Odometry)

The raw acceleration data a_x was first filtered and smoothed by a low-pass Butterworth filter. Then after the twice integration process for the smoothed acceleration data, the position (x_a, y_a) can be obtained. After the Kalman filter estimator is applied to the accelerometer and encoder information, the new estimated position $(\hat{x}, \hat{y})$ is obtained as the calibrated odometry measurement for the robot.

Figure 6-5 compares the trajectories obtained from the odometry reading (x_o, y_o), the accelerometer reading (x_a, y_a), and the calibrated odometry measurement $(\hat{x}, \hat{y})$ obtained through the fusion subsystem. This experiment lasted for 40 seconds. As shown in the figure, the odometry reading (blue trajectory) differs significantly from the accelerometer reading (green trajectory), while the calibrated odometry measurement $(\hat{x}, \hat{y})$ takes into account information from both sensors, making it suitable for fusion with the Optitrack system.

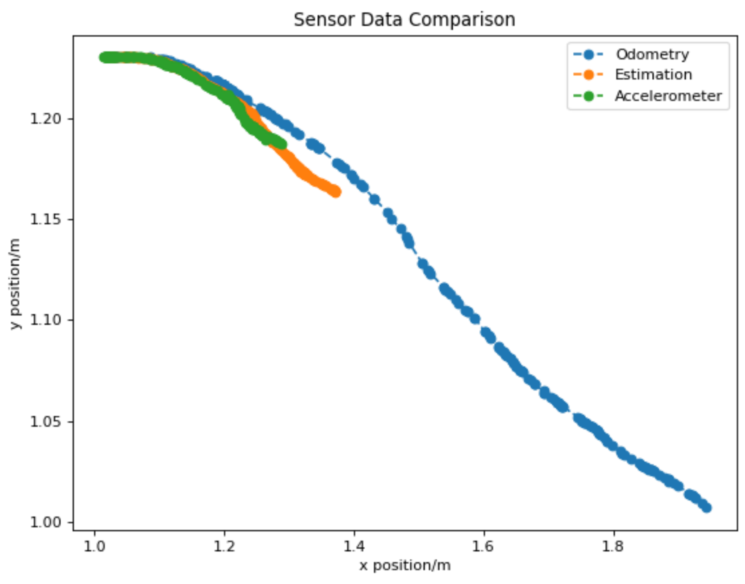

Figure 6-5: Fusion Subsystem Result(Accelerometer and Odometry)

Low-pass Filters

The equation of the first-order Butterworth low-pass filter in figure 6-4 is defined in equation 6-1, where the x stands for the raw measurement, and y is for the filtered result. While designing the low-pass filter, the sampling time and cutoff frequency are assigned as $f_s = 1000$ Hz and $f_c = 2.5$ Hz respectively.

$$16.895y_i = (x_i + x_{i-1}) + 14.895y_{i-1} \tag{6-1}$$

The standard deviation of acceleration data a_x distribution decreases from 0.636 to 0.445 after being smoothed by the filter, as shown in figure 6-6.

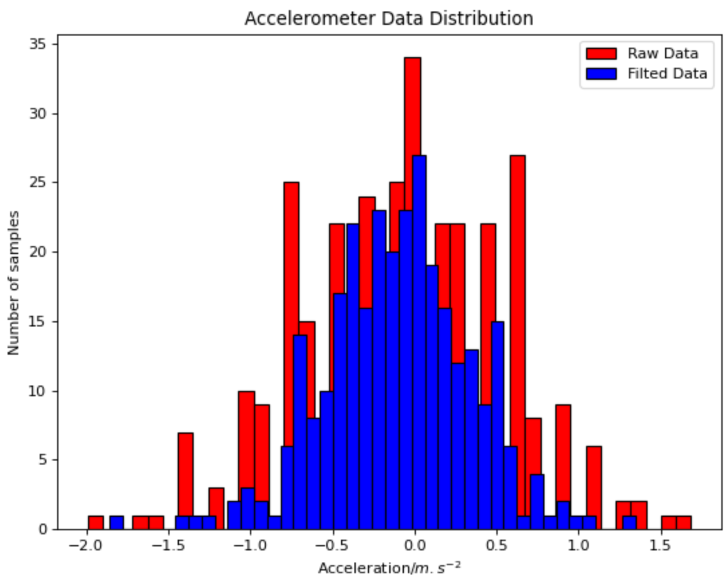

Figure 6-6: Accelerometer Data Distribution

6-1-3 Hardware Improvement

The signal from the IR emitters on the Elisa-3 robots was too weak to be captured by the Optitrack camera. To address this issue, stickers were placed on the robot's cap, as shown as the left robot in figure 6-7. However, due to the cap's small size, it was difficult to place separate stickers in a manner that did not interfere with each other that reduce the camera's ability to identify them.

Moreover, the Optitrack requires at least three markers to form a rigid body for tracking purposes, but since the robot is not always parallel to the ground and misdirects the reflection, some markers can become occluded. While adding more markers could help, the small size of the cap limits the number of markers that can be added.

To overcome this issue, the cap was enlarged, allowing for the placement of multiple stickers, as demonstrated by the right robot in figure 6-7. Additionally, the color of the cap was changed to black to reduce the reflection which could have otherwise decreased the camera's sensing ability.

6-1-4 Noise Model

After the hardware modification and refinement, the noise characteristics of each sensor source should be renewed. The noise matrices for the EKF estimator are basically derived from empirical measurements of sensor noise. The noise measurement data and distribution can be found in appendix A-2.

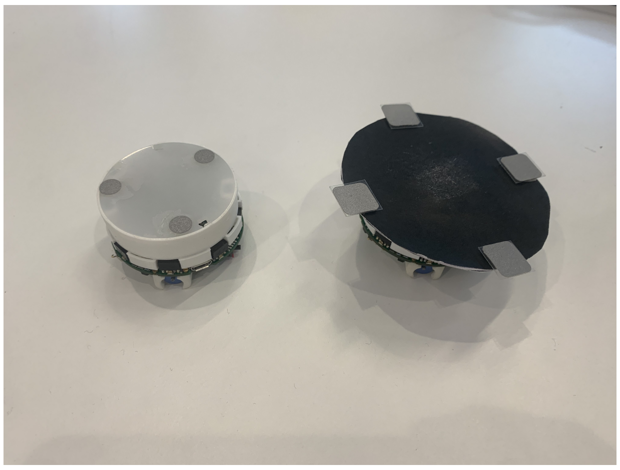

Figure 6-7: Hardware Improvement

The noise matrices are defined in equation 6-2, where $\mathbf{R}_1^n$, $\mathbf{R}_2^n$, $\mathbf{R}_3^n$ are noise matrices of odometry, accelerometer and camera.

$$\begin{aligned}
\mathbf{R}_1^n &= \operatorname{diag}\Big(1\left[\,\mathrm{m}^2\right], \quad 1\left[\,\mathrm{m}^2\right], \quad 1\left[\mathrm{rad}^2\right]\Big) \\
\mathbf{R}_2^n &= \operatorname{diag}\Big(0.448\left[\,\mathrm{m}^2\right], \quad 0.448\left[\,\mathrm{m}^2\right], \quad 0.448\left[\mathrm{rad}^2\right]\Big) \\
\mathbf{R}_3^n &= \operatorname{diag}\Big(0.001\left[\,\mathrm{m}^2\right], \quad 0.001\left[\,\mathrm{m}^2\right], \quad 0.001\left[\mathrm{rad}^2\right]\Big)
\end{aligned} \tag{6-2}$$

6-1-5 Odometry Resetting

In the real-world experiment, the odometry is not only observed as biased but also affected by the noncontinuous data. When camera information is unavailable, localization relies on onboard sensors, and a noncontinuous measurement can lead to incorrect position estimation. Moreover, since the robot selects the closest camera measurement based on its current estimation, an erroneous estimation can prevent the robot from finding camera data, resulting in untracked behavior. Consequently, in the implementation, the odometry is periodically reset to enhance localization performance.

6-2 Experiments Results

In this section, several experiments were conducted to test the performance of the localization ability and the influence of different parameter selections. In these experiments, robots are randomly moving inside the lab field, and the base sampling time T_s is set as 50 ms. The red cross stands for starting point, the blue cross stands for ending point. The results without resetting the odometry are presented first, followed by the results with resetting the odometry.

6-2-1 Result without Resetting

OWA Style VS Cascade Style

In this experiment, the performance of two localization architectures, namely OWA style and Cascade Style, is compared. The architectures are illustrated in figure 4-3 and 4-4 respectively. The sampling time for all sensors in both architectures is set to $T_s = 50$ ms. Since the sampling time is the same for all sensors, the Kalman estimator used in this subsection is single-rate EKF.

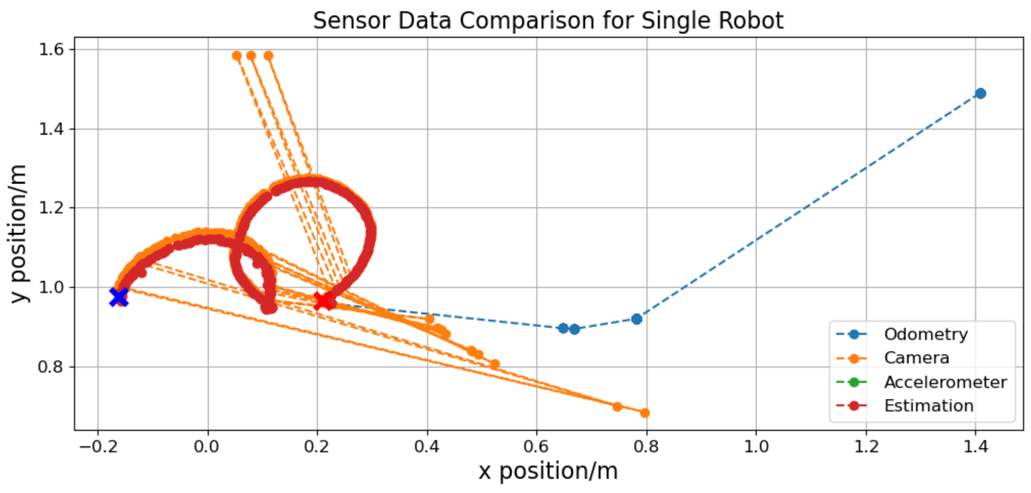

Figure 6-8: Robots Tracking with Cascade Style and High Sample Frequency

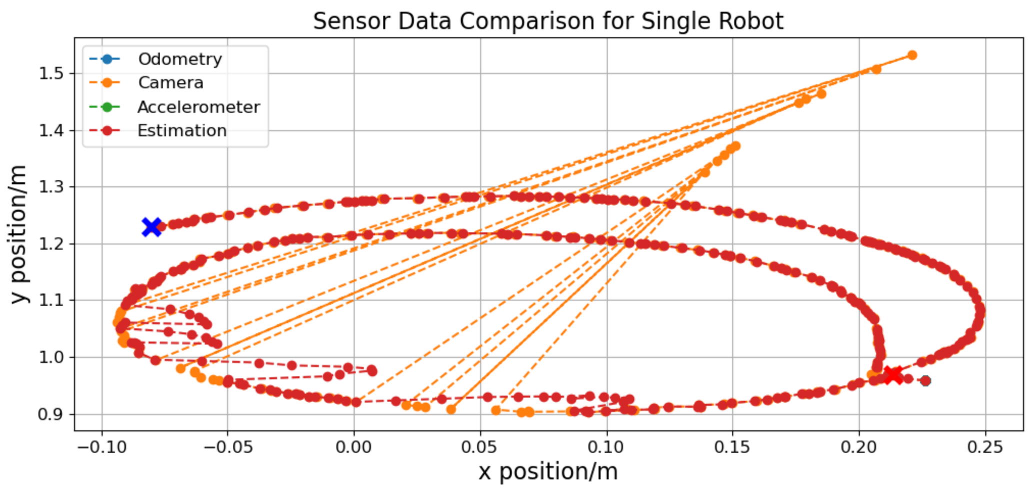

Figure 6-9: Robots Tracking with OWA Style and High Sample Frequency

The tracking results for both tracking architectures are presented in figure 6-8 and figure 6-9. In both experiments, the orange line represents the camera measurements, which exhibit significant fluctuations. When the camera readings become too abnormal, the robot replaces them with its own estimation to fuse with other sources and generate the fused estimation. And the blue line represents the odometry measurements, it shows that the communication quality is bad. In figure 6-8, the odometry data has a large gap between each sample, and the accelerometer's data is absent. In figure 6-9, both onboard information is lost.

In the cascade style, the tracking system is able to compensate for the fluctuations on the camera side and generate a smoother trajectory. By referring to the noise matrices equation 6-2, we can observe that the estimator relies less on the odometry. Consequently, when the camera information is absent, the estimation is less affected by the biased odometry. This leads to a more reliable estimation in the absence of camera data.

In the OWA style, the final estimation is obtained by weighted ordering, the weight variation is shown in figure 6-10, since the accelerometer is offline in this experiment, the weight variation won't be shown. The weight assigned to the camera side decreases when the camera information is deemed untrustworthy. As a result, biased odometry can dominate the estimation process, leading to fluctuations in the robot's trajectory.

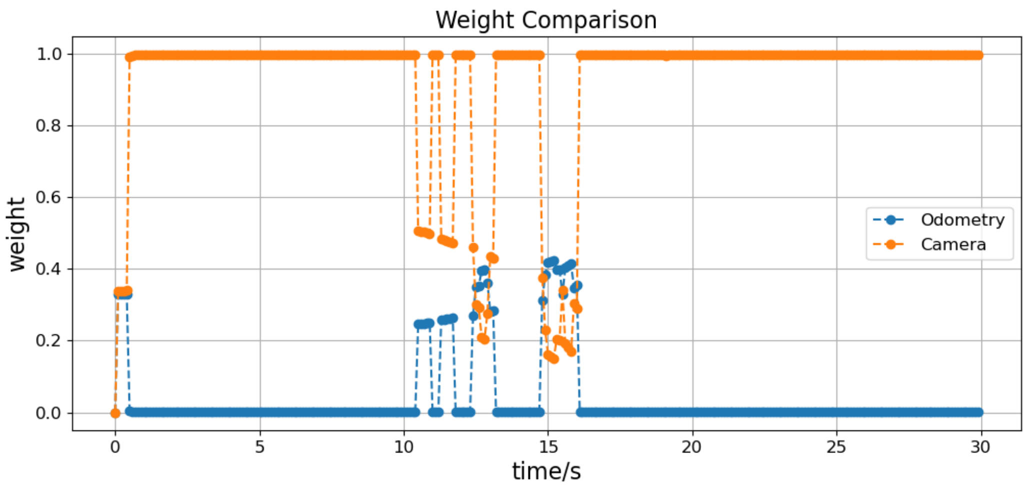

Figure 6-10: Weight Variation with OWA Style and High Sample Frequency

Single-rate EKF VS Multi-rate EKF

In this experiment, the performance of different versions of the Extended Kalman Filter (EKF), Single-rate EKF and Multi-rate EKF, is compared under different architectures. The sampling time for the onboard sensors in both architectures is set to $T_s = 50$ ms. Additionally, the camera has a slower sampling time of $3T_s = 150$ ms.

The performance of the two different algorithms is initially tested using a cascade style configuration, as illustrated in figures 6-11 and 6-12. During the experiments, communication problems occurred, leading to the loss or noncontinuous nature of the onboard information.

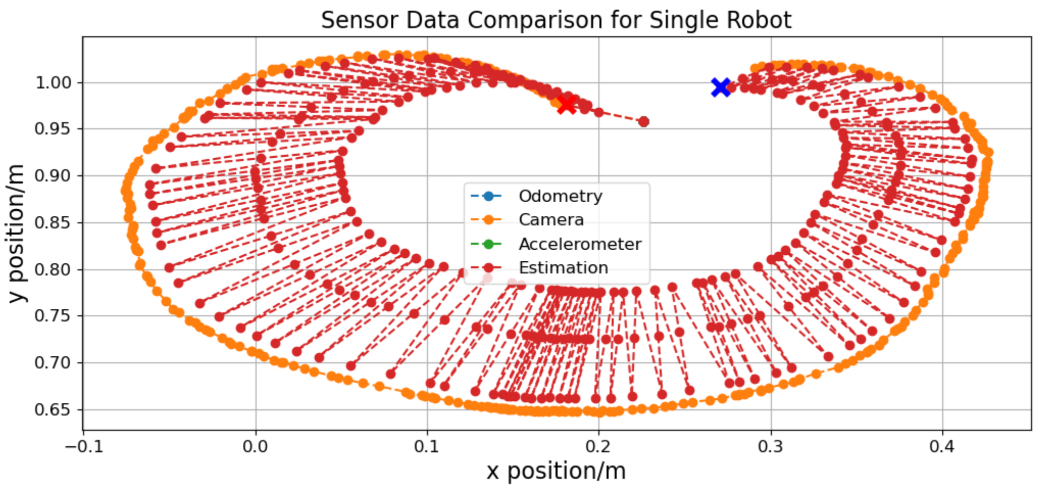

Figure 6-11: Robots Tracking with Cascade Style, Low Sample Frequency and sr-EKF

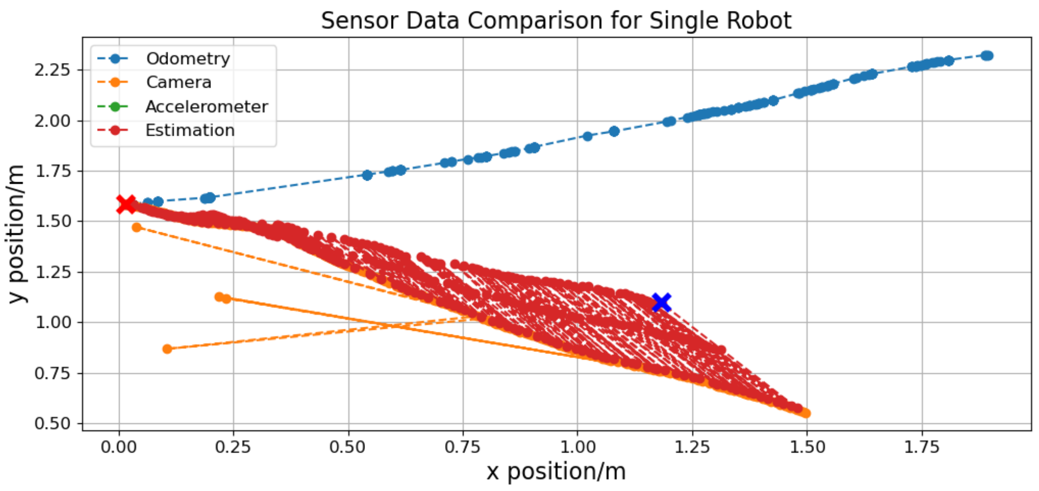

Figure 6-12: Robots Tracking with Cascade Style, Low Sample Frequency and mr-EKF

These images demonstrate that without frequent corrections from the camera, the estimation relies heavily on the camera readings when they are available. However, in cases where the camera reading is missing, the biased odometry becomes dominant, resulting in a jagged and unacceptable trajectory.

Figure 6-12 illustrates that although the use of mr-EKF brings the estimation closer to the camera position when camera readings are available, it does not significantly improve the overall performance. While the MSE between camera measurements and estimations reduces from 13.4e-5 to 8.2e-5 with mr-EKF. When the camera readings are absent, the estimation heavily relies on biased odometry, leading to large inaccuracies in the estimation.

Then the performance of two different algorithms is tested on OWA style, as shown in figure 6-13 and figure 6-14.

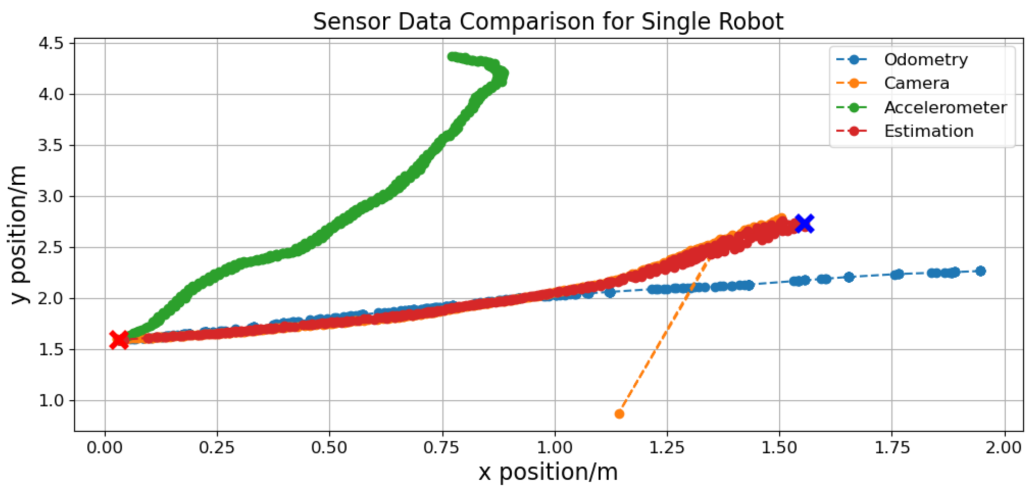

Figure 6-13: Robots Tracking with OWA Style, Low Sample Frequency and sr-EKF

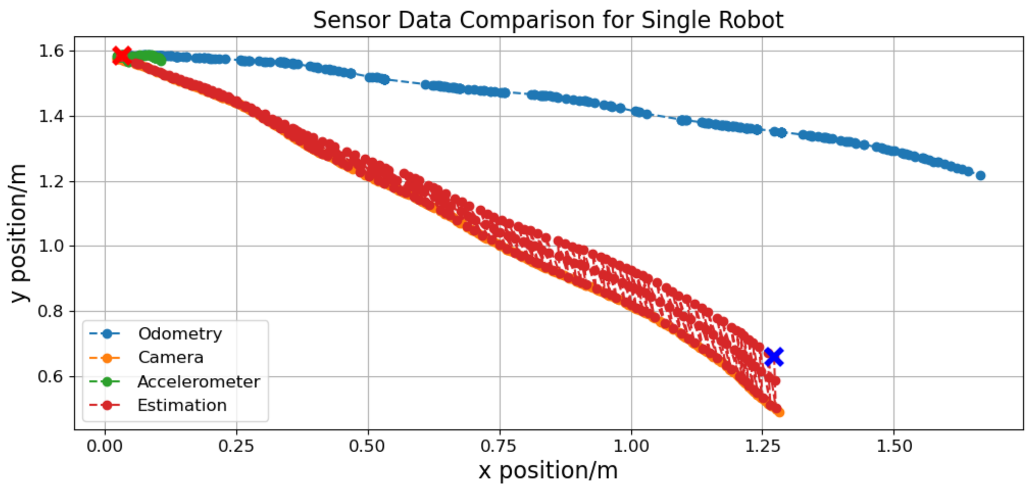

Figure 6-14: Robots Tracking with OWA Style, Low Sample Frequency and mr-EKF

In this experiment, similar to the case in the cascade style, reducing the sampling frequency of the camera has an impact on the smoothness of the overall trajectory. The slower rate of camera samples leads to a higher reliance on odometry, which can introduce errors and result in lower accuracy in the final position estimation. However, it is worth noting that in this experiment, the odometry readings are relatively unbiased compared to previous cases, which leads to a smaller fluctuation in the estimated trajectory.

When utilizing the mr-EKF algorithm, it does not perform correction when there is missing camera information. Consequently, the final estimation, as depicted in figure 6-14, is inaccurate and exhibits significant fluctuations. Such fluctuations are considered unacceptable in real control tasks where precise and stable estimations are crucial.

Furthermore, in scenarios where the camera information is consistently biased over an extended period, as illustrated in figure 6-9, the accuracy of the localization is significantly compromised. This is primarily due to the absence of accurate sensor data during that specific time instance, which further exacerbates the localization error.

The presence of exacerbated localization error can result in significant deviations from the true position estimation. In situations where the camera measurement is not explicitly assigned to each robot, the robots are required to determine their own position based on their current knowledge. However, due to the deviated estimation, the robots may struggle to accurately identify the correct camera measurement when it becomes available, e.g. in figure A-7, the orange line represents the camera measurement, the biased estimation leads the robot picked the wrong camera measurement. This mismatch can result in mistracking, leading to inaccuracies in the overall localization process.

The aforementioned results clearly demonstrate that relying solely on a single reliable source of information can lead to a degradation in the quality of estimation when combined with low-accuracy sources of information. The presence of these additional sources introduces noise, biases, or missing data, which negatively impact the overall estimation accuracy.

6-2-2 Result with Resetting

The results from the previous experiments highlight the significant impact of noncontinuous odometry on overall tracking accuracy. As observed in figure 6-9 and figure 6-11, when the odometry information remains unchanged and doesn't compensate for the absence of camera data, the tracking accuracy is compromised. Consequently, regularly resetting the odometry can be an effective approach to enhance overall accuracy, as the case in figure 6-13, particularly in situations where the camera information is biased by noise or unavailable.

Based on that, experiments were conducted under the following conditions: both sensors were reset at a rate of $T = 3T_S = 150$ ms.

OWA Style VS Cascade Style

Firstly, the performance of OWA style and Cascade style localization architectures is compared. And all the sensors have the highest sampling rate as $T_s = 50$ ms. The tracking trajectories are shown in figure 6-15 and 6-16.

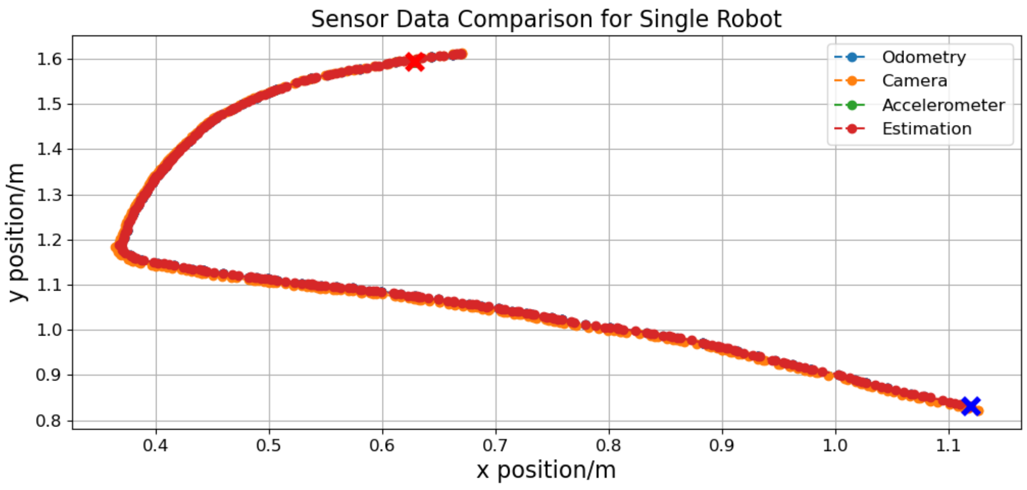

Figure 6-15: Robots Tracking with Cascade Style, High Sample Frequency and Reset

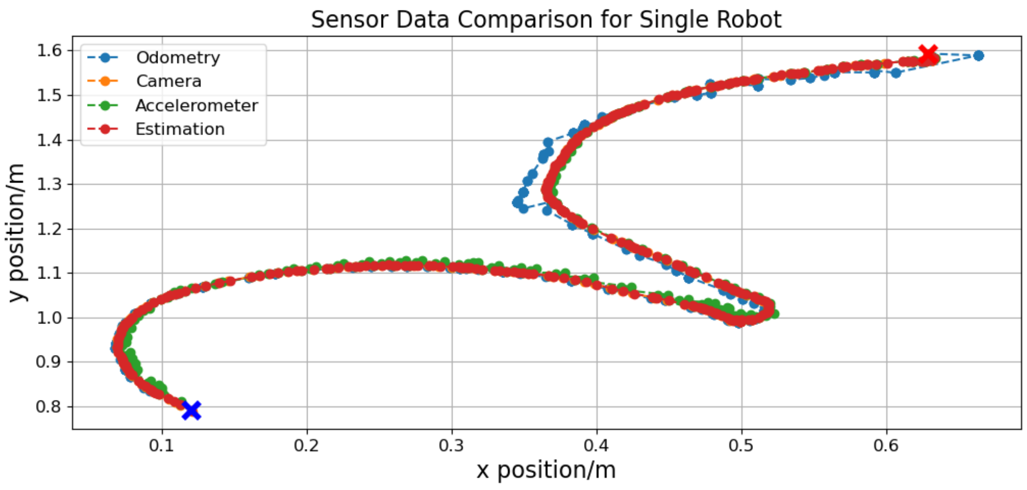

Figure 6-16: Robots Tracking with OWA Style, High Sample Frequency and Reset

The tracking trajectories depicted in Figure 6-15 and Figure 6-16 clearly demonstrate that regular sensor resetting, combined with either the Cascade style or OWA style localization architectures, leads to accurate estimations of the robots' positions, which is similar to the case without frequently resetting. Especially in the case shown in 6-16, the estimator successfully overcomes the variance present in the odometry and accelerometer measurements and produces a smooth tracking trajectory,

Single-rate EKF VS Multi-rate EKF

Later, the performance of different versions of the Extended Kalman Filter (EKF), Single-rate EKF and Multi-rate EKF, is compared under different architectures.

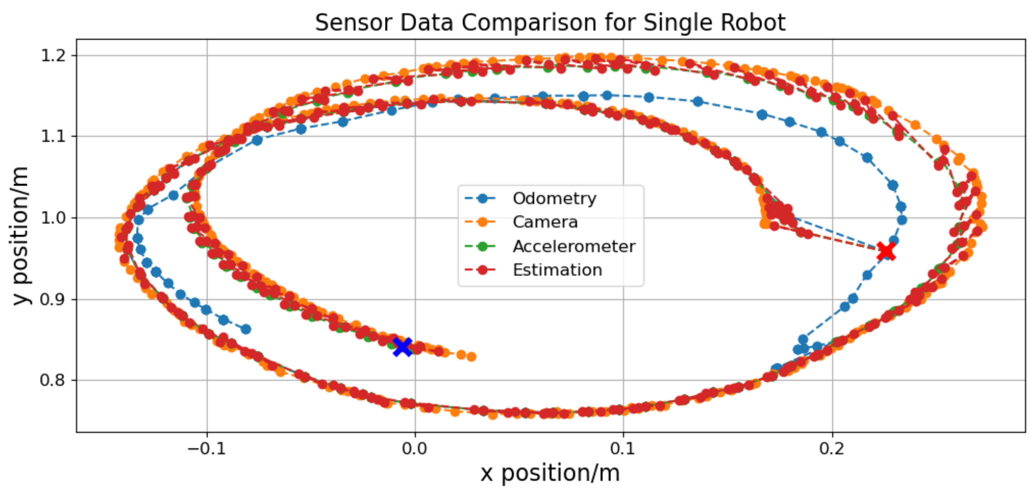

Figure 6-17: Robots Tracking with Cascade Style, Low Sample Frequency, sr-EKF and Reset

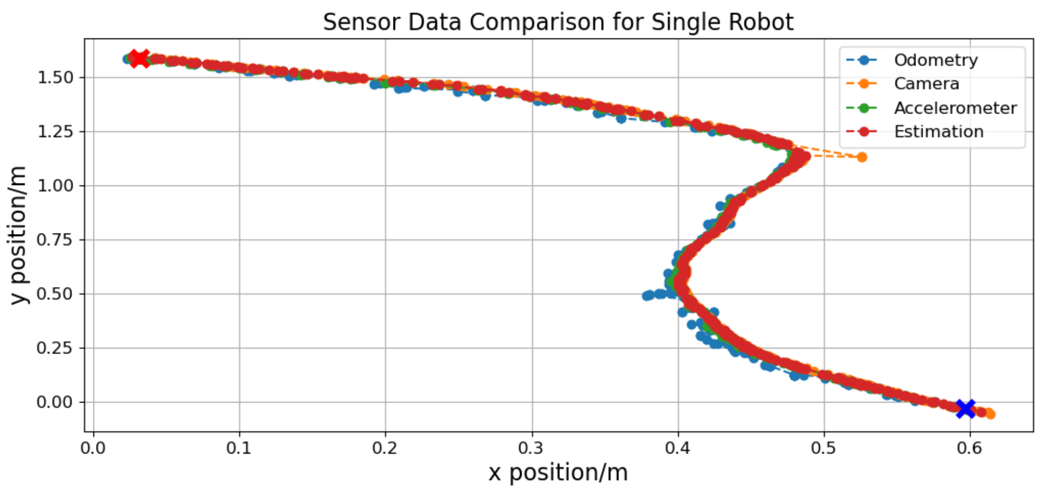

Figure 6-18: Robots Tracking with Cascade Style, Low Sample Frequency, mr-EKF and Reset

In both Figure 6-17 and Figure 6-18, the comparison between the sr-EKF and the mr-EKF under the Cascade style localization architecture is presented. In both cases, it can be observed that the estimation performance improves when the reset operation is applied, compared to the scenario without resetting.

However, since the reset function won't be always reliable, the odometry might be wrongly reset, e.g. in Figure 6-17, it can be observed that the blue line, representing the estimation with odometry reset, deviates from the other lines. As a result, the estimation still exhibits fluctuations, and applying the cascade style in a low sensor sampling rate scenario can be risky and result in unsatisfactory output.

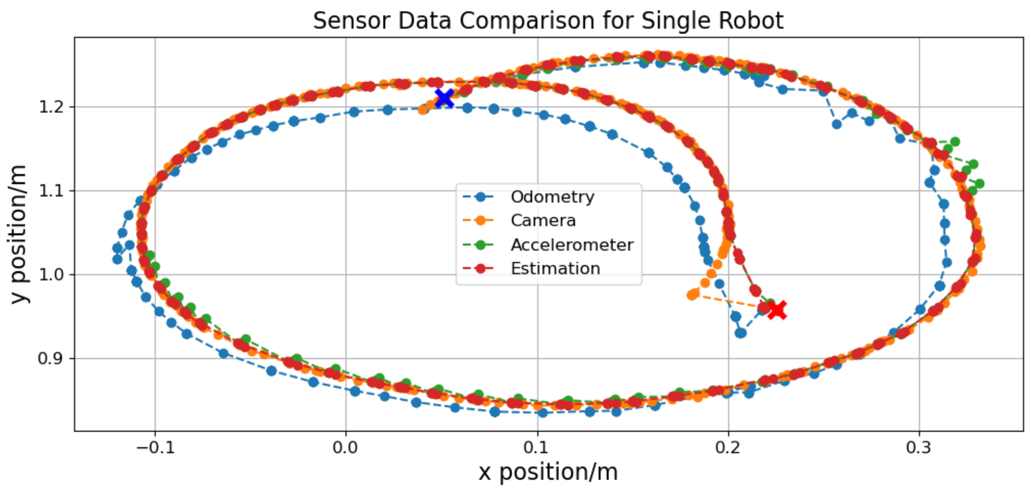

Figure 6-19: Robots Tracking with OWA Style, Low Sample Frequency, sr-EKF and Reset

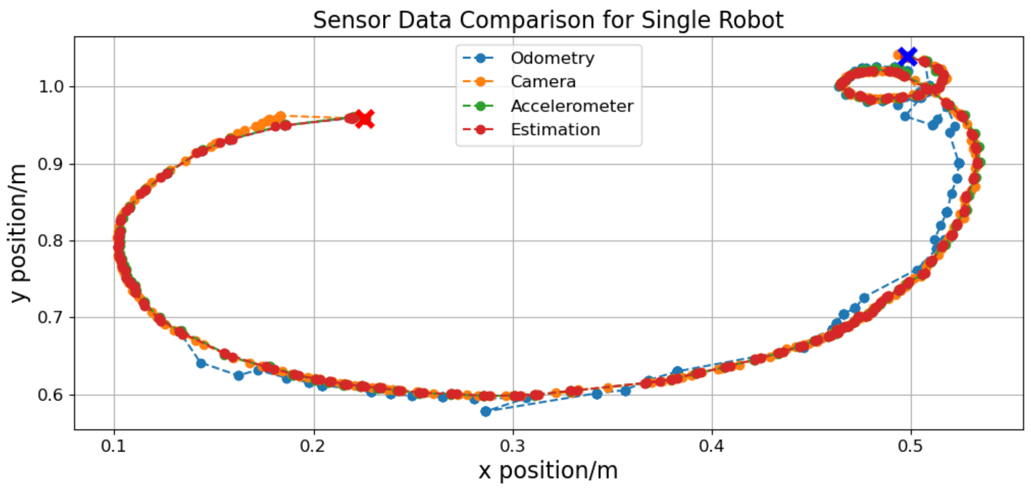

Figure 6-20: Robots Tracking with OWA Style, Low Sample Frequency, mr-EKF and Reset

Figure 6-19 and 6-20 illustrates the comparison between sr-EKF and mr-EKF under OWA style. Figures show the estimation result is satisfying because the estimator doesn't get affected by the noise from the odometry and accelerometer, the final estimation is suitable for real-world control tasks. It is worth noting that the application of mr-EKF does not significantly improve the overall performance.

6-2-3 Experiments with 8 Robots

Below is an example of tracking performance while robots were randomly moving. Figure 6-21 shows the effeteness of the localization system.

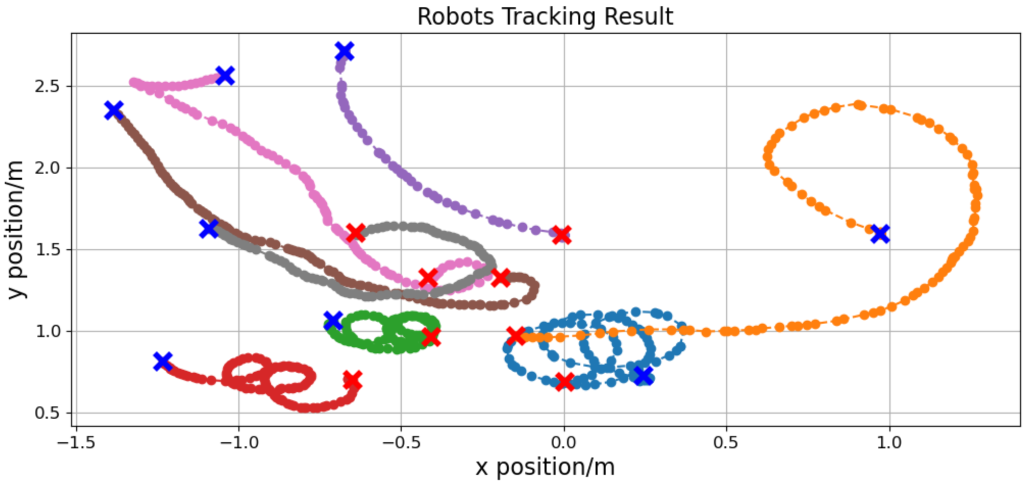

Figure 6-21: Experiments with 8 Robots

6-3 Experiment: Applied in A Control Task

In this section, a brief introduction to the control task is given, providing context and background information on the specific task that the robots are required to perform. Following that, in Section 6-3-2, the tracking results obtained from the experiment and discussion are presented.

6-3-1 Introduction of the Control Task

The recommended localization algorithm from 6-4 is applied in comparing different consensus algorithms in the context of consensus problems in a robot swarm.

Consensus is a fundamental problem in the field of robotics, particularly in the context of robot swarms. It involves achieving an agreement among a group of autonomous robots on a certain value or behaviour, even in the presence of uncertainties and communication limitations.

Consensus algorithms play a crucial role in enabling effective coordination and decision-making within a swarm, allowing the robots to work together toward a common objective. These algorithms determine how information is shared and combined among the robots, influencing the convergence speed, accuracy, and robustness of the consensus process. In recent years, there has been a growing interest in exploring and comparing different consensus algorithms to understand their strengths, weaknesses, and suitability for various swarm applications [48].

In this control task, the focus lies on achieving consensus in a robot swarm using a localization algorithm. Localization is a critical component for enabling successful consensus. Accurate localization allows each robot to have an awareness of its own position as well as the positions of its neighbouring robots. By integrating the localization algorithm with the consensus algorithm, the swarm can converge to a consistent and agreed-upon state, enabling cooperative behaviour and collective decision-making.

The comparison experiment was conducted by [49], where several consensus algorithms on a robot swarm are implemented and tested. To enhance the accuracy and reliability of the experiment, the localization algorithm was applied to provide accurate position information for the robots. The experiments aimed to evaluate the performance of the consensus algorithms under different formation scenarios, such as aligned or rectangular formations.

6-3-2 Results Discussion

In the following experiments, the tracking algorithm is the single-rate Extended Kalman Filter with OWA architecture and frequently resetting.

The base sampling time T_s was set to 10 ms. The sampling time of all sensors was configured to be the same as the base sampling time T_s, and the collected data was stored at a rate of $5T_s$. Furthermore, the reset rate was set as $50T_s$ to periodically reset sensor values.

Experiment without Localization

An experiment was conducted without utilizing localization, where three robots were released. The resulting trajectory is depicted in figure 6-22. In this experiment, each robot solely relied on its onboard odometry for estimating its position. When all robots aligned parallel to the y-axis, the swarm came to a halt.

This picture depicts a "successful" test scenario where the robots perceive themselves as aligned based on the information from their onboard odometry (represented by the blue cross and the trajectory labelled as "Odometry"). As a result of the consensus reached among the robots, they came to a halt and ceased further movement.

However, in reality, their true position is totally different from the odometry, as indicated by the red cross and trajectory labelled as "real". This implies that without the localization algorithm, the swarm will have a large possibility fail to form a consensus.

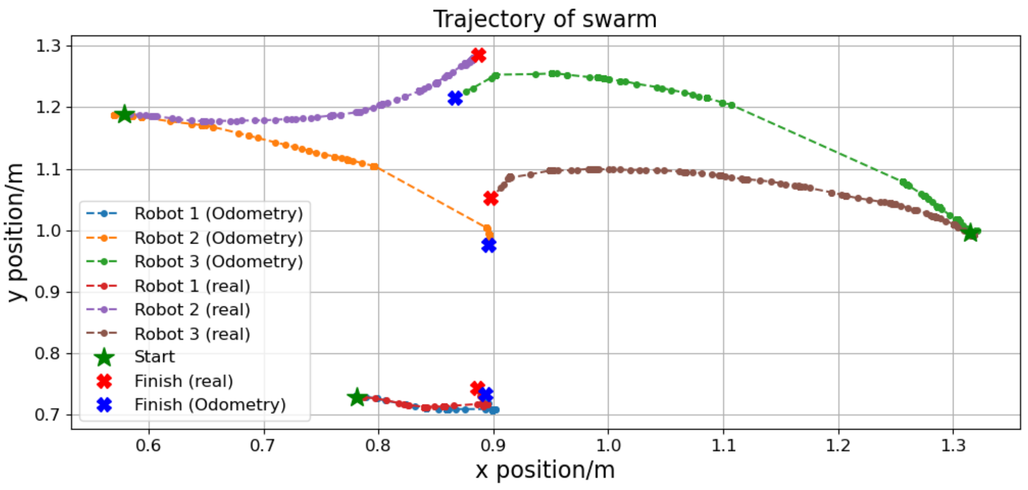

Figure 6-22: Robots Trajectory without Localization

However, in reality, the actual position of the robots was significantly different from the estimated positions based on the odometry, as indicated by the red cross and trajectory labelled as "real." This observation highlights the limitation of relying solely on odometry for position estimation. Without the localization algorithm, the swarm faces a high probability of failing to achieve consensus, as the true positions of the robots deviate considerably from their estimated positions.

Aligned Formation

By utilizing the localization algorithm, we conducted an experiment in which four robots were released to form an aligned formation. The trajectories of each robot's movement were recorded and analyzed. Figure 6-23 provides an overview of the trajectories and more detailed information on each robot's trajectory can be found in Appendix A-4.

The trajectories clearly demonstrate the effectiveness of the localization algorithm in providing accurate position estimations for the robots. By utilizing the algorithm and periodically resetting the odometry with the appropriate values, the robots are able to achieve more accurate position tracking.

Rectangular Formation

A series of tests on a swarm of robots starting from a rectangular formation were conducted. Three consensus algorithms were implemented and tested: the Event-triggered Control Algorithm (ETC), the Periodic Event-triggered Control Algorithm (PETC), and the Phase Algorithm (PA). The differences between each algorithm won't be discussed, and only the experimental result with ETC is presented, More experimental results with different consensus algorithms can be found in A-6 and A-7.

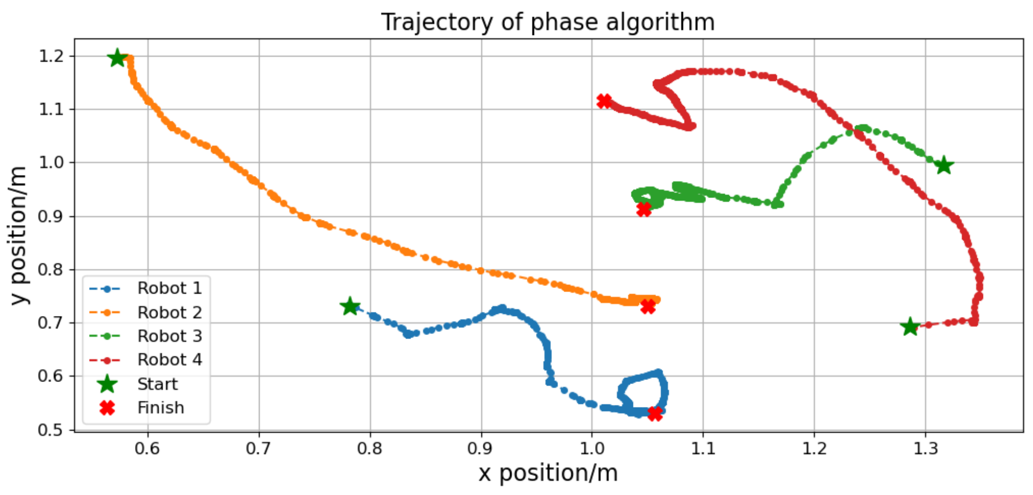

Figure 6-23: Robots Trajectory Aligned Formation

Figure 6-24 provides an overview of the trajectories with ETC, each robot's trajectory can be found in Appendix A-5.

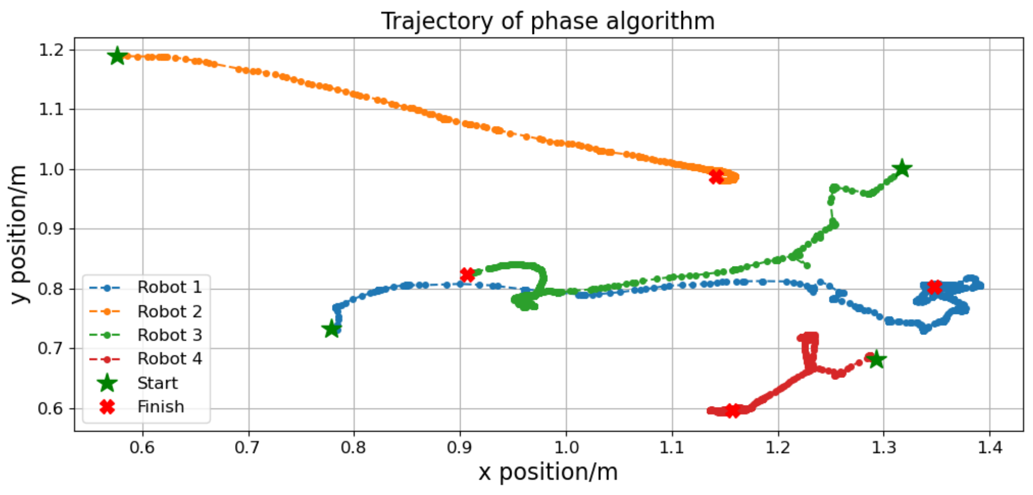

Figure 6-24: Robots Trajectory Rectangular Formation with ETC

6-4 Discussion and Conclusion

From the simulation result of section 5-4, we can reach a conclusion that the OWA style architecture is generally better than the cascade style architecture, and applying mr-EKF, the total performance doesn't greatly get improved.

In the real-world experiment, the odometry is even worse than the case in the simulation, as a result, a similar conclusion can be reached, which is summarized as follows:

1. Samples the measurement at a higher rate can improve the total tracking performance.
2. Using multi-rate Extended Kalman Filter seems not to improve the localization much.
3. By resetting the onboard sensors regularly, the drift on odometry and accelerometer can be eliminated and yield better tracking results.
4. Even with resetting the onboard sensors, using the OWA style often produces a smoother trajectory than using the Cascade style.

A recommended setup would be to apply the OWA style with a high sampling rate for all sensors and frequent resetting. More plots are shown in the A-3.

Based on this recommended setup and experimental result from the application on the control task, we can conclude that the localization algorithm can eliminate the effect of the biased odometry such as figure A-28 and A-32. The biased odometry as coloured in blue now has little impact on overall accuracy.

When the camera connection is stable and there are fewer camera flaws, the tracking results show satisfactory performance. However, as the camera flaws increase, the accuracy of the camera-based estimation deteriorates. This can be observed in figure A-27, where the orange line deviates significantly from the actual trajectory during periods with camera flaws.

During the periods when the camera is not available, the system relies solely on onboard information, such as odometry, to estimate the robot's position, the estimation during these times tends to have larger variations compared to the case where camera information is relatively accurate.

Chapter 7

Conclusion and Future Work

In this chapter, the results of this thesis, future work and possible improvements are presented. The sections 7-1 summarizes the thesis content and conclusion. Later in the section 7-2 focus on several avenues for future improvements.

7-1 Conclusion

The goal of this master's thesis was to develop a specialized localization architecture for Elisa-3 robots, focusing on enhancing their localization capabilities in indoor environments. The achieved tracking results demonstrate the success of this objective.

The developed localization architecture effectively addresses the challenges of accurate position estimation by integrating data from multiple sensors, including measurements from onboard odometry and the global camera. By leveraging the strengths of each sensor and compensating for their limitations, the algorithm provides reliable localization for the robots.

Throughout the experiments, including random motion tracking and utilization of the estimated positions for control purposes, the tracking results showcase the effectiveness of the localization architecture. Even in situations where the camera connection was compromised or temporarily unavailable, the algorithm relied on onboard sensor data to maintain accurate position estimation and guided the robots toward achieving the desired formations.

In conclusion, the localization architecture was successfully implemented for Elisa-3 robots in indoor environments. The achieved tracking results demonstrate the effectiveness of the algorithm and its potential for various control tasks.

7-2 Future Work

While the developed localization architecture has demonstrated satisfying results in enhancing the localization capabilities of Elisa-3 robots in indoor environments, there are several areas that can reduce the localization performance, therefore, potential improvement and future development could be explored.

7-2-1 Hardware Improvement

The Elisa-3 robots, although effective in their functionality, face certain hardware challenges that can significantly impact their overall performance. Two key challenges are communication issues and instability in maintaining a stable standing position.

Communication problems directly affect the accuracy of the odometry readings, which are crucial for precise localization. Inadequate communication can introduce noise and errors in the sensor data, leading to less reliable position estimations. And instability in maintaining a stable standing position can disrupt the camera tracking process and give a wrong measurement.

Moreover, it is worth noting that with the existing coarse modification, mentioned in 6-7, robots' obstacle avoidance capabilities can be hindered which might lead to collisions and robots sticking together.

In summary, overcoming the hardware limitations is crucial for enhancing the localization capabilities of the Elisa-3 robot swarm. By exploring potential repairs or upgrades and considering the impact on obstacle avoidance, it is possible to improve the reliability and accuracy of the localization algorithm and enable more robust performance in various control tasks.

7-2-2 Software Improvement

Apart from the possible hardware improvement, the ROS schema, as depicted in figure 6-3 where the communication between the controller and the swarm relies on the ROS publish/subscribe mechanism, can introduce latency and delay in data transmission and cause a real-time issue.

In a control task that requires precise coordination and synchronization among the robots, any delay in communication can lead to suboptimal performance and reduced effectiveness of the localization algorithm. According to the data presented in Table 7-1, it is evident that as the size of the swarm increases, the average communication time exhibits a nonlinear growth pattern. Additionally, the variance of communication time also increases, resulting in a higher occurrence of time jitters.

number of robots	4	6	8
average computation time/s	0.144	0.333	0.707
variance of computation time	0.009	0.037	0.120
average communication time/s	0.399	0.832	1.713
variance of communication time	0.036	0.153	0.340

Table 7-1: Introduced Time Delay by Size of Swarm

To address the real-time issue and mitigate delays in communication, a potential solution is to modify the existing schema, as depicted in Figure 6-3. One proposed modification involves moving the controller part to the ROS node, as illustrated in Figure 7-1. This modification aims to minimize the reliance on network communication and improve the overall real-time performance of the system.

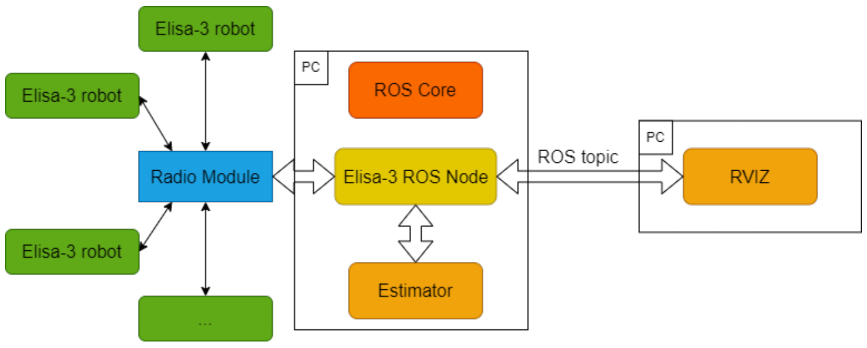

Figure 7-1: Modified Elisa-3 Robots ROS Schema

Integrating the controller and estimator into the ROS node offers significant advantages in reducing communication latency and minimizing its impact on the system. This integration enables more immediate and synchronized data exchange between the robots, as the controller and estimator can directly interact within the local ROS node.

Appendix A

Measurement Data

In this chapter, the detailed measurement data will be given. In section A-1, the MSE of each robot in each round will be procided. Later, the section A-2 will give the distribution of the sensor's data and its covariance. Since the overview of the tracking trajectory is given in the main latter, each robot's trajectory will be given in the following section A-3 and section A-4.

A-1 Simulation Result

MSE of Each Robot when only odometry is applied is shown in the table A-1.

	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
1	110.050	105.554	141.118	32.559	30.773	98.822	100.010	41.232	75.179	86.039
2	60.274	53.572	53.675	43.248	67.426	14.380	117.232	207.489	148.910	41.059
3	28.232	100.471	71.409	76.173	81.952	23.322	60.195	60.811	70.785	34.246
4	48.981	118.178	25.605	64.414	35.001	62.093	95.062	89.367	71.184	43.448
5	129.314	78.263	21.456	106.428	49.459	40.137	129.123	171.837	24.527	64.159
6	50.350	125.793	81.184	154.055	50.222	115.303	106.159	54.736	33.660	42.960
7	79.552	57.794	98.374	91.055	110.261	57.478	62.856	125.523	38.044	14.312
8	80.653	73.519	90.375	46.728	186.615	118.069	78.582	65.373	149.761	58.489
9	74.852	58.738	91.089	61.636	67.737	72.544	30.001	60.752	77.641	49.643
10	38.476	38.006	149.397	143.415	35.623	89.811	109.642	37.928	37.939	74.410

Table A-1: MSE of Each Robot with Odometry

MSE data with sr-EKF applied in Cascade style is shown in the table A-2.

	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
1	45.548	10.586	20.916	46.324	36.977	16.060	26.714	9.480	17.538	42.408
2	58.684	47.292	38.317	33.145	39.997	27.090	48.427	35.283	28.706	38.101
3	17.972	39.777	34.142	25.779	14.107	28.869	32.917	28.855	26.067	56.607
4	31.108	22.697	19.362	57.398	17.456	28.335	21.664	31.792	34.462	18.457
5	59.658	29.261	37.593	31.104	29.177	50.165	28.135	17.701	40.390	32.069
6	32.089	56.907	24.939	46.753	26.444	21.869	39.438	28.637	54.990	29.562
7	33.305	37.166	42.760	47.833	27.897	40.241	25.701	27.881	38.898	56.633
8	25.045	42.768	18.783	42.210	39.060	27.528	24.792	50.118	11.662	18.481
9	21.128	33.843	48.482	28.017	32.361	36.136	16.489	25.603	44.536	40.615
10	32.433	56.178	51.654	42.930	31.501	30.269	38.314	34.420	26.774	25.619

Table A-2: MSE of Each Robot with Cascade sr-EKF

MSE data with sr-EKF applied in OWA style is shown in the table A-3.

	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
1	26.075	6.270	80.586	31.300	23.128	55.127	15.738	23.833	10.742	57.720
2	44.509	44.855	12.013	28.674	24.510	28.087	16.789	54.059	25.091	8.577
3	13.897	65.290	9.333	5.094	22.238	48.258	12.930	22.817	57.819	19.119
4	11.569	27.429	24.206	24.158	24.593	17.299	21.397	31.480	32.767	30.592
5	14.933	17.688	10.913	49.887	33.689	30.137	13.827	24.859	11.108	25.405
6	30.775	55.976	12.238	7.592	18.237	17.212	18.457	23.166	41.157	7.785
7	37.182	12.565	31.870	32.357	20.278	10.679	17.259	20.024	50.672	13.860
8	28.295	110.284	58.252	42.420	44.938	35.238	19.775	3.522	3.367	24.161
9	27.292	21.626	55.883	11.226	24.415	22.445	25.146	22.279	11.699	57.103
10	13.245	42.352	12.668	2.784	7.361	10.578	34.698	38.672	10.748	17.950

Table A-3: MSE of Each Robot with OWA sr-EKF

MSE data with mr-EKF applied in cascade style is shown in the table A-4.

	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
1	39.041	69.232	44.124	25.097	36.913	48.017	89.748	46.460	28.325	26.333
2	36.039	33.204	53.533	8.374	31.404	22.435	27.131	27.272	9.171	32.802
3	41.614	26.466	24.625	38.432	23.765	98.220	58.385	54.830	28.966	54.942
4	40.698	14.854	36.290	37.748	39.014	107.620	34.948	64.829	68.913	15.940
5	10.180	25.116	35.696	45.129	41.542	35.317	29.329	25.893	151.775	9.326
6	43.910	190.963	30.619	31.889	18.911	25.993	24.467	31.704	40.911	81.936
7	42.710	35.829	46.133	11.969	131.499	26.746	150.453	19.598	55.638	65.141
8	51.998	20.394	18.149	26.354	40.434	31.745	8.654	29.152	34.976	74.652
9	22.567	64.052	34.969	29.738	14.854	46.194	17.896	95.991	45.914	41.890
10	52.555	27.795	17.798	28.864	27.699	79.945	53.392	58.859	32.373	39.369

Table A-4: MSE of Each Robot with Cascade mr-EKF

MSE data with mr-EKF applied in OWA style is shown in the table A-5.

	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
1	44.362	8.190	14.198	12.458	38.980	23.238	7.950	18.688	9.547	12.007
2	23.010	42.423	8.727	33.214	20.988	18.614	21.596	10.160	3.764	47.685
3	23.618	14.324	13.837	23.335	24.975	16.943	26.886	44.409	32.517	12.818
4	50.624	30.420	22.408	20.329	6.555	14.818	18.972	21.948	28.142	14.987
5	8.517	26.243	14.752	25.921	21.969	7.527	24.974	35.508	23.587	29.538
6	15.992	9.899	8.522	9.706	52.653	39.126	36.148	23.397	9.750	22.505
7	22.794	5.109	44.095	35.091	32.062	22.574	21.489	14.041	35.670	10.386
8	32.490	26.246	40.595	9.233	8.574	30.381	22.512	52.477	30.770	10.384
9	20.293	7.051	9.506	11.265	7.962	35.469	5.780	12.700	18.804	10.676
10	25.575	10.962	24.673	57.173	7.364	8.165	18.784	28.937	13.187	22.409

Table A-5: MSE of Each Robot with OWA mr-EKF

A-2 Noise Matrices

The following figures provide a visualization of the distribution of accelerometer data (a_x) and camera position measurements (x and y).

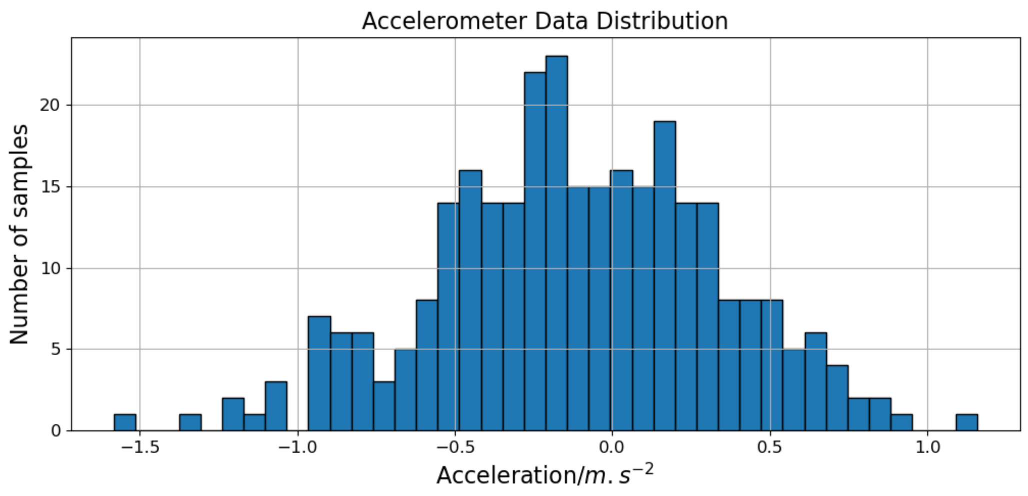

Figure A-1: Robot 2 Trajectory Aligned Formation

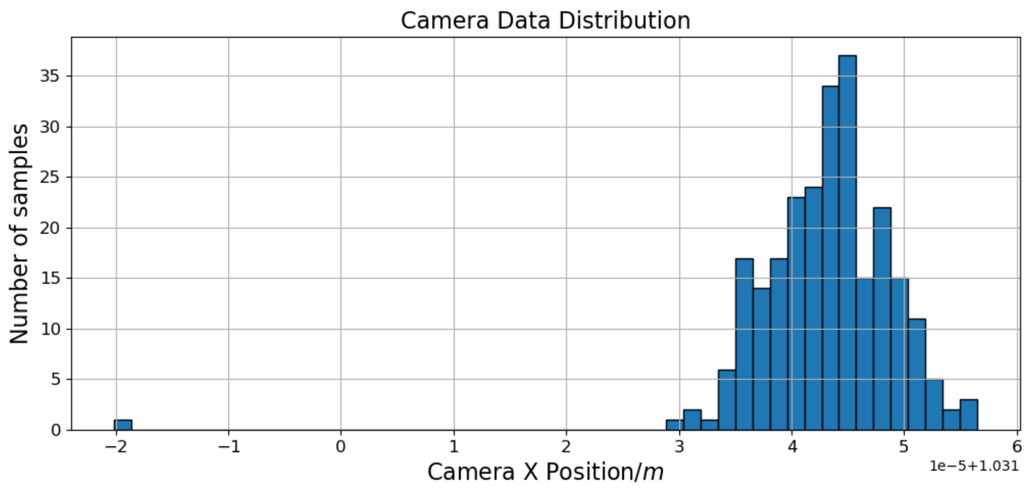

Figure A-2: Robot 2 Trajectory Aligned Formation

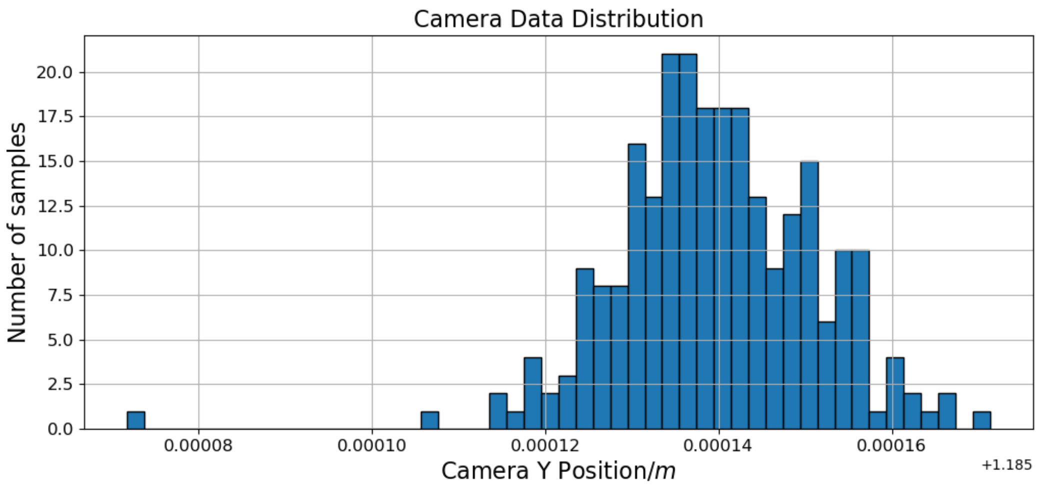

Figure A-3: Robot 2 Trajectory Aligned Formation

Based on the collected data, the accelerometer variance a_x was determined to be 0.448, while the camera measurement variances for x and y were found to be 6.402e-6 and 1.158e-6, respectively.

However, subsequent experiments revealed that the camera measurements exhibited higher levels of noise compared to when the robot remained stationary, due to the missing tracking and merge condition. As a result, during the implementation, the variances for x and y were adjusted to 0.001 to account for the increased noise in the camera measurements.

Regarding odometry, the variances were set to 1 due to communication issues and accumulated errors during encoding. These factors contributed to a higher level of uncertainty in the odometry readings.

A-3 Real Experiment

A-3-1 OWA Style VS Cascade Style, without Resetting

The plots provide an overview of the robots' trajectory under the Cascade style and high sampling rate, without resetting, considering different versions of the EKF. Back to main latter 6-2-1.

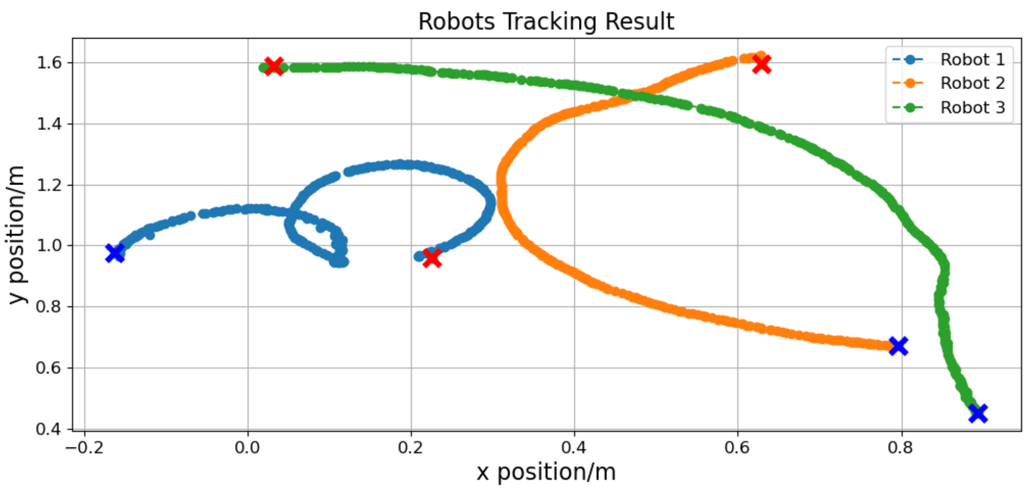

Figure A-4: Robots Trajectory Overview with sr-EKF, Cascade Style and High Sampling Rate

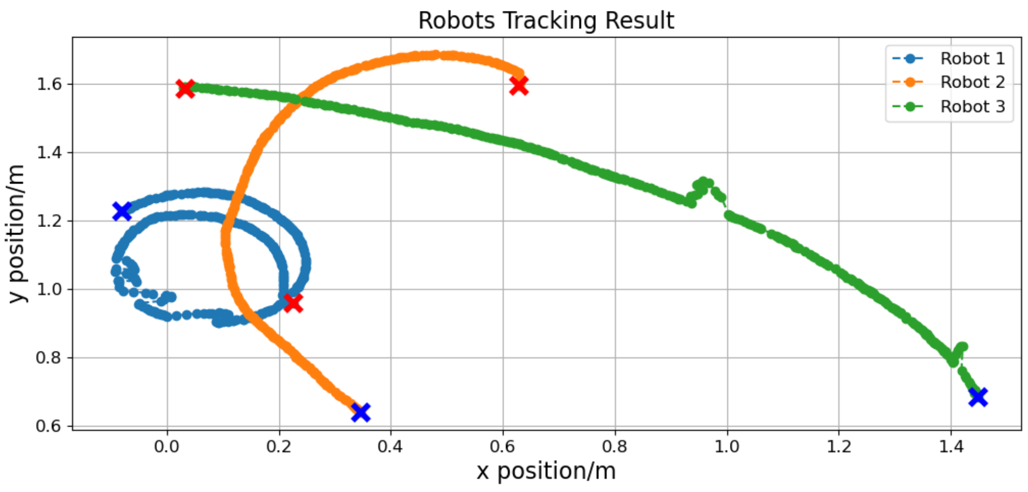

Figure A-5: Robots Trajectory Overview with sr-EKF, OWA Style and High Sampling Rate

A-3-2 Single-rate EKF VS Multi-rate EKF, without Resetting

The plots provide an overview of the robots' trajectory under the sr-EKF or mr-EKF and low sampling rate, without resetting, considering different versions of the architectures. Back to main latter 6-2-1.

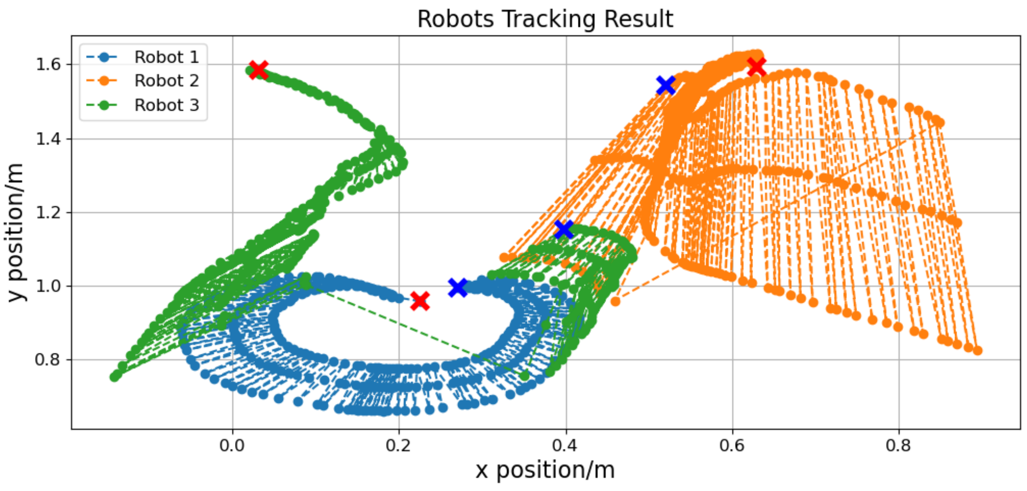

Figure A-6: Robots Trajectory Overview with sr-EKF, Cascade Style and Low Sampling Rate

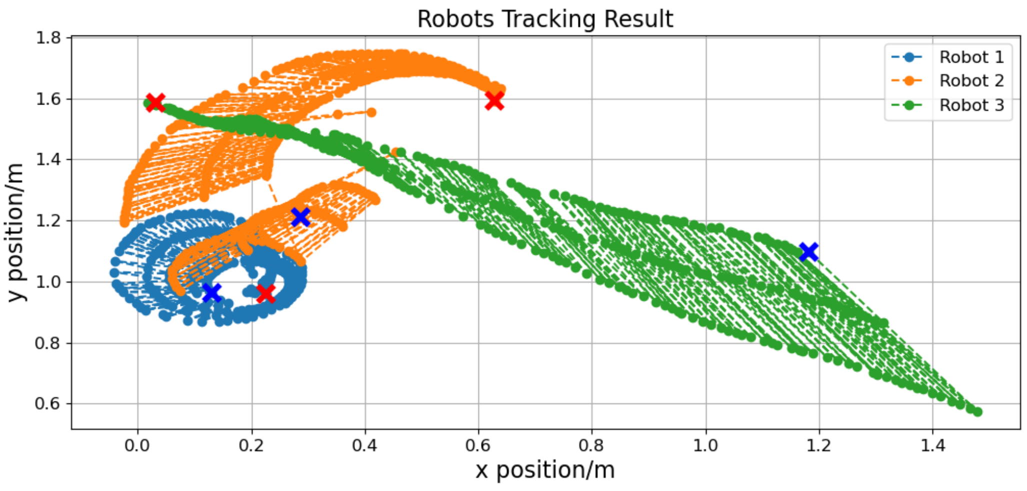

Figure A-7: Robots Trajectory Overview with mr-EKF, Cascade Style and Low Sampling Rate

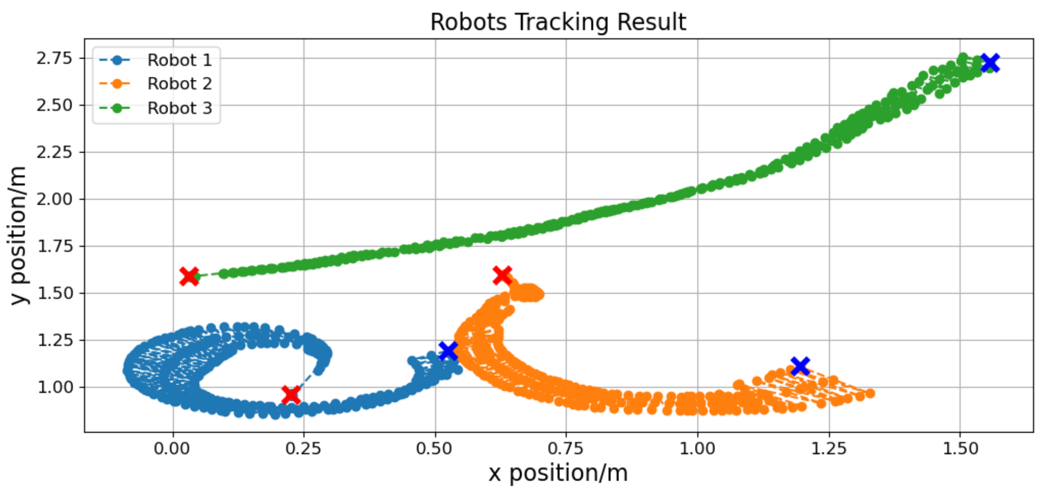

Figure A-8: Robots Trajectory Overview with sr-EKF, OWA Style and Low Sampling Rate

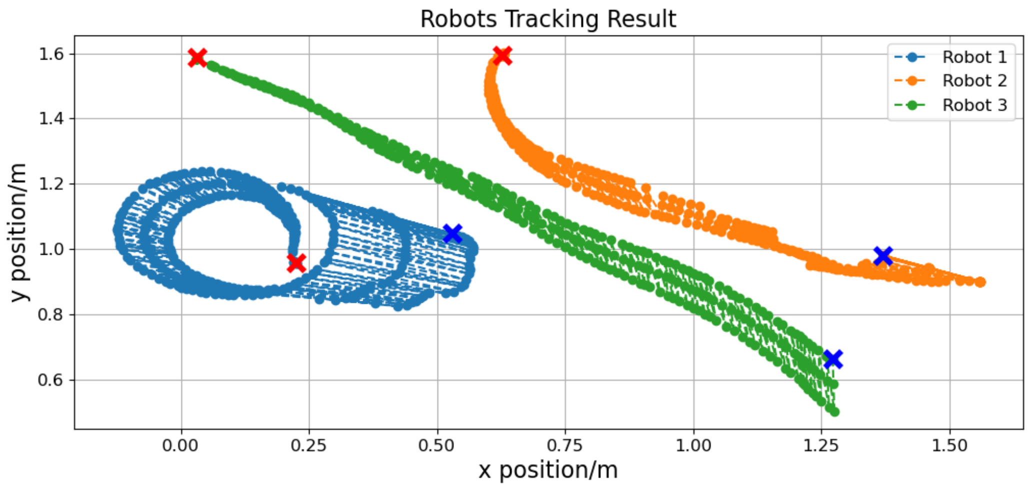

Figure A-9: Robots Trajectory Overview with mr-EKF, OWA Style and Low Sampling Rate

A-3-3 OWA Style VS Cascade Style, with Resetting

The plots provide an overview of the robots' trajectory under the Cascade style and high sampling rate, with resetting, considering different versions of the EKF. Back to main latter 6-2-2.

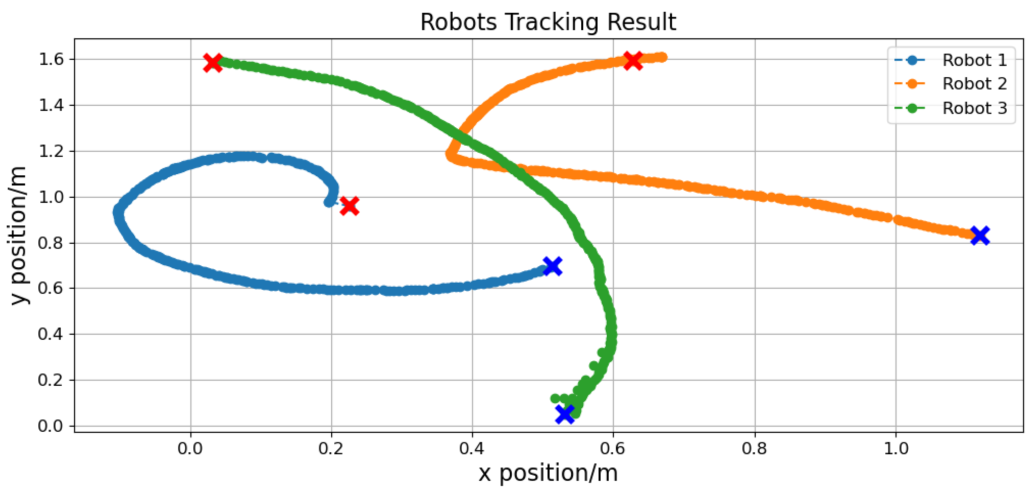

Figure A-10: Robots Trajectory Overview with Resetting, sr-EKF, Cascade Style and High Sampling Rate

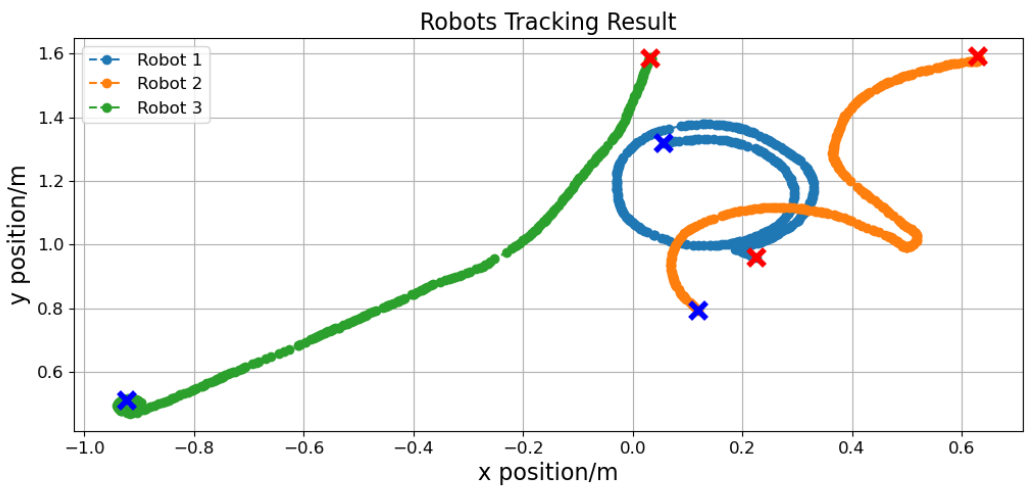

Figure A-11: Robots Trajectory Overview with Resetting, sr-EKF, OWA Style and High Sampling Rate

A-3-4 Single-rate EKF VS Multi-rate EKF, with Resetting

The plots provide an overview of the robots' trajectory under the sr-EKF or mr-EKF and low sampling rate, without resetting, considering different versions of the architectures. Back to main latter 6-2-2.

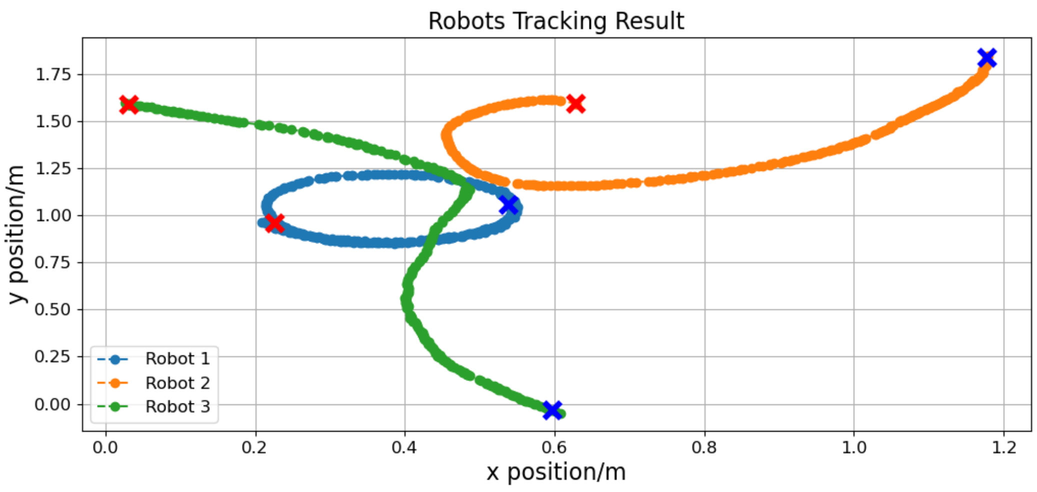

Figure A-12: Robots Trajectory Overview with Resetting, sr-EKF, Cascade Style and Low Sampling Rate

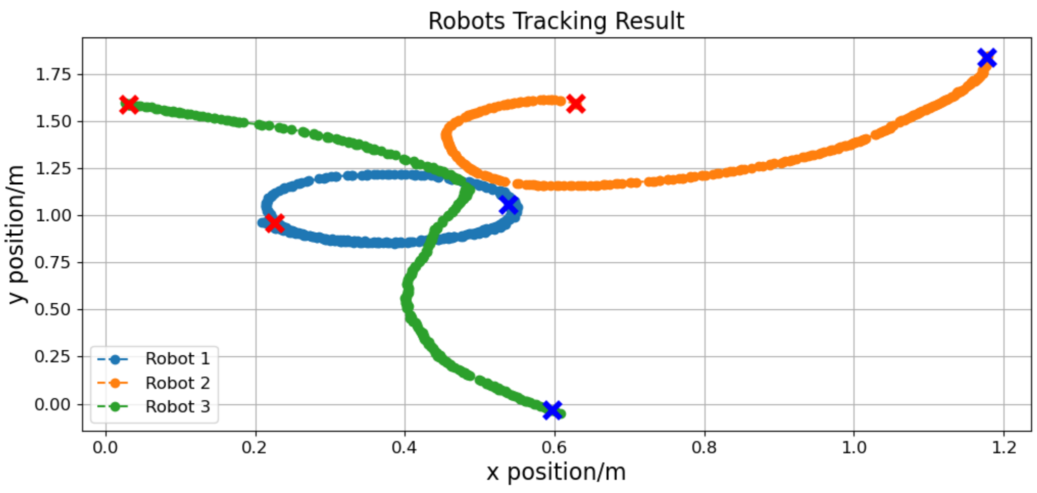

Figure A-13: Robots Trajectory Overview with Resetting, mr-EKF, Cascade Style and Low Sampling Rate

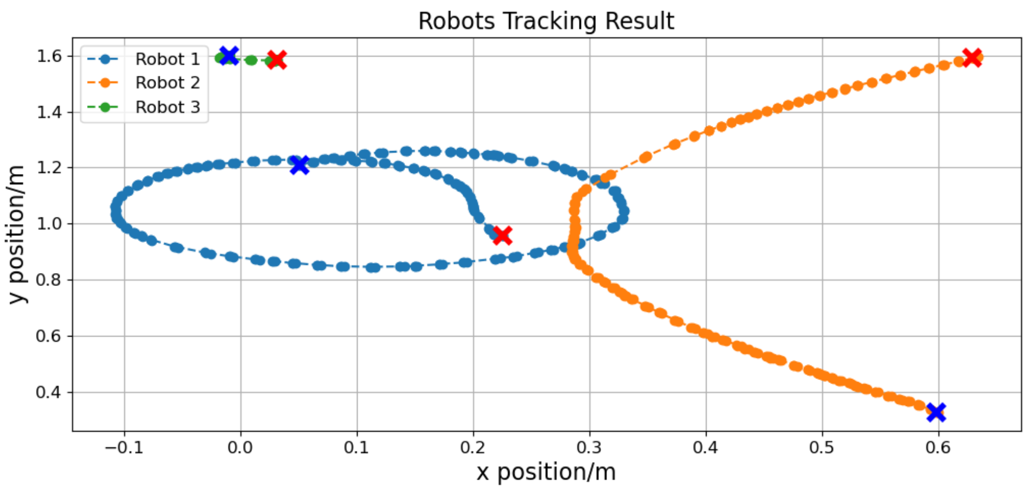

Figure A-14: Robots Trajectory Overview with Resetting, sr-EKF, OWA Style and Low Sampling Rate

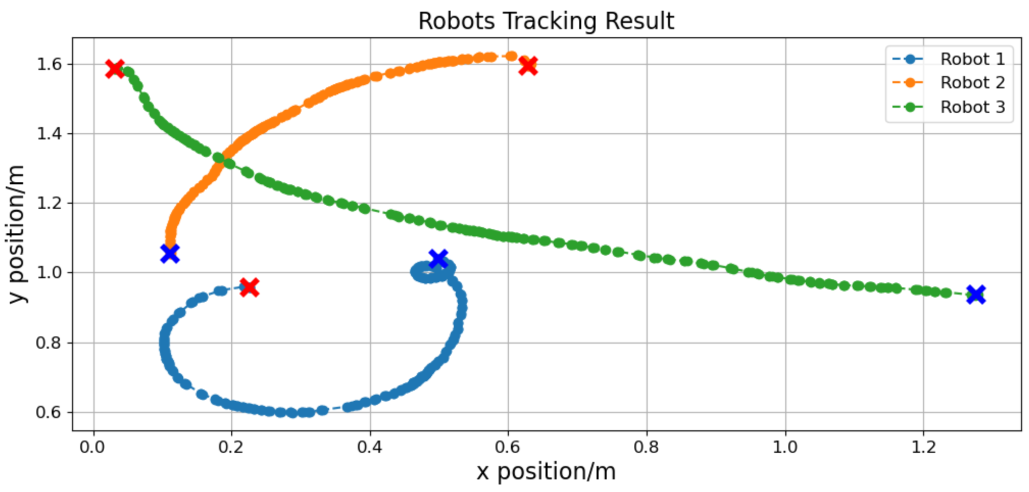

Figure A-15: Robots Trajectory Overview with Resetting, mr-EKF, OWA Style and Low Sampling Rate

A-4 Real Experiment, Aligned Formation

Back to main latter 6-3-2

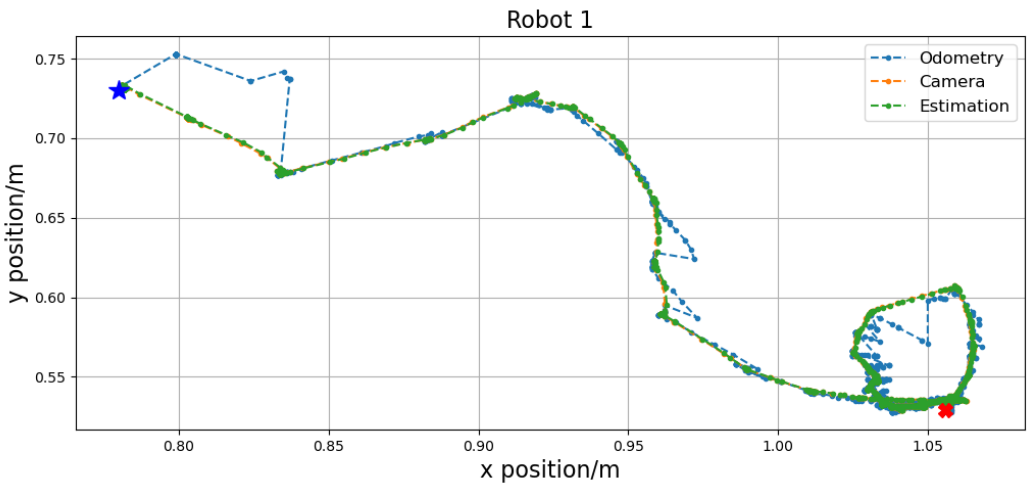

Figure A-16: Robot 1 Trajectory Aligned Formation

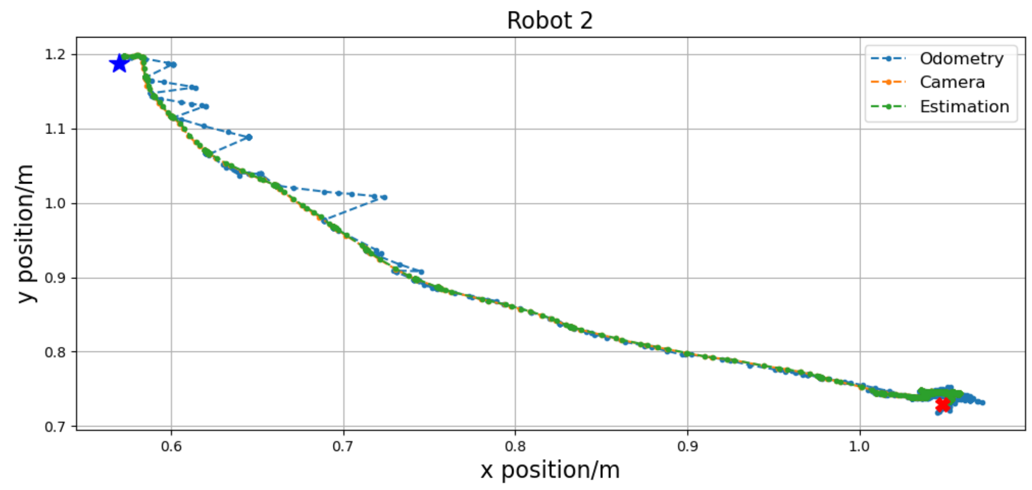

Figure A-17: Robot 2 Trajectory Aligned Formation

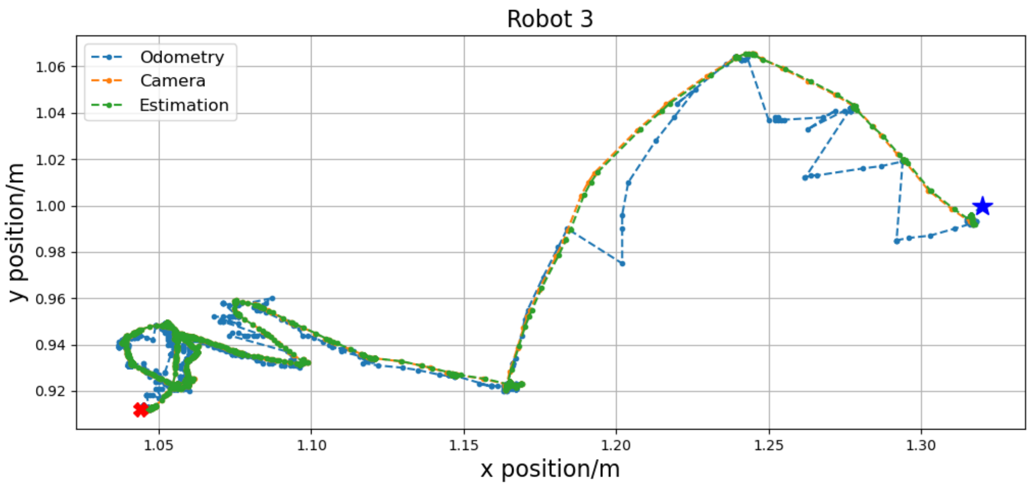

Figure A-18: Robot 3 Trajectory Aligned Formation

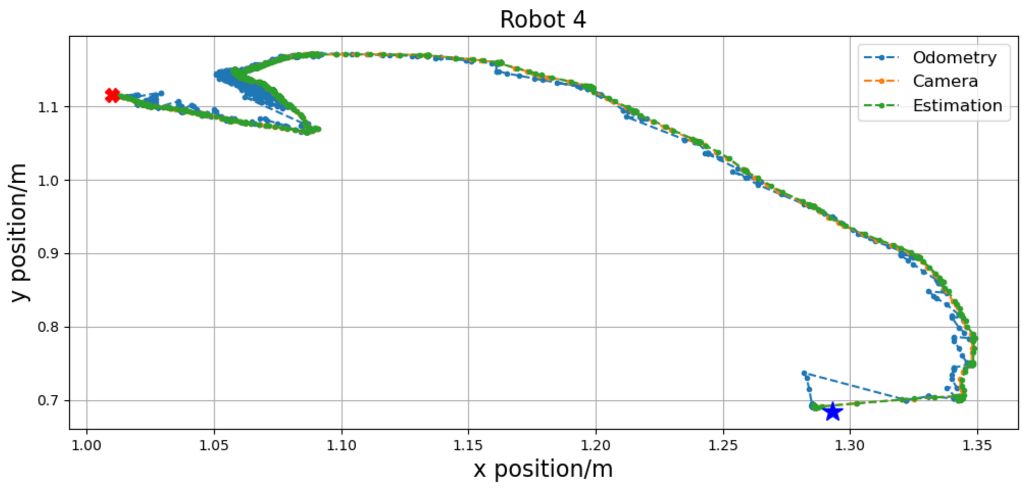

Figure A-19: Robot 4 Trajectory Aligned Formation

A-5 Real Experiment, Rectangular Formation with ETC

Back to main latter 6-3-2

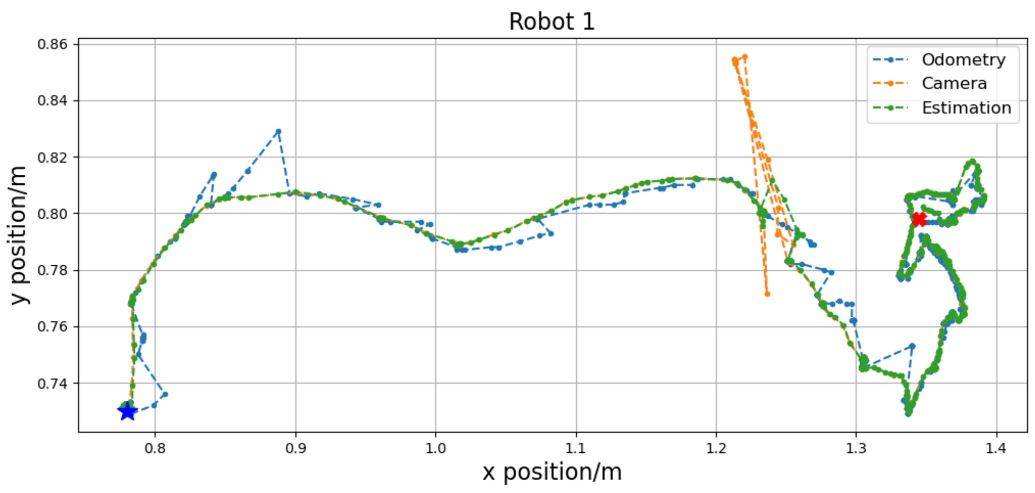

Figure A-20: Robot 1 Trajectory Rectangular Formation with ETC

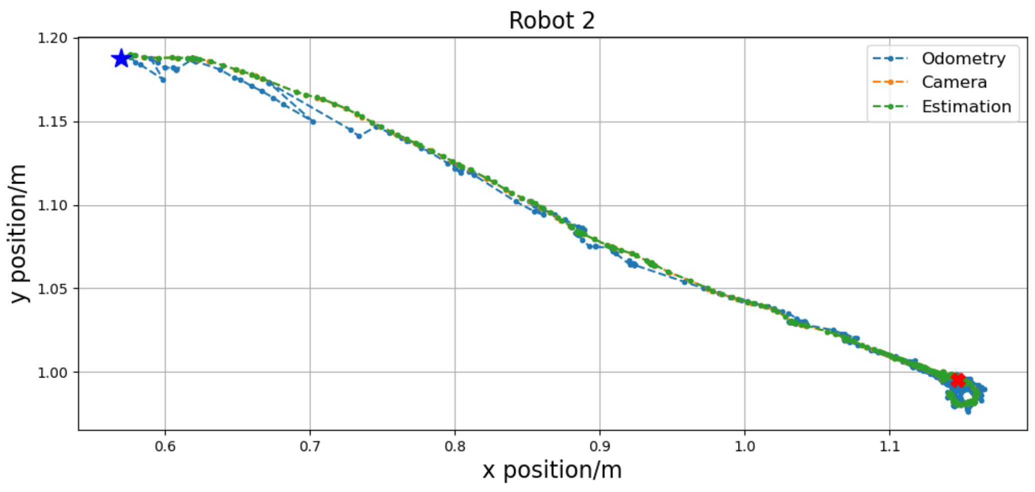

Figure A-21: Robot 2 Trajectory Rectangular Formation with ETC

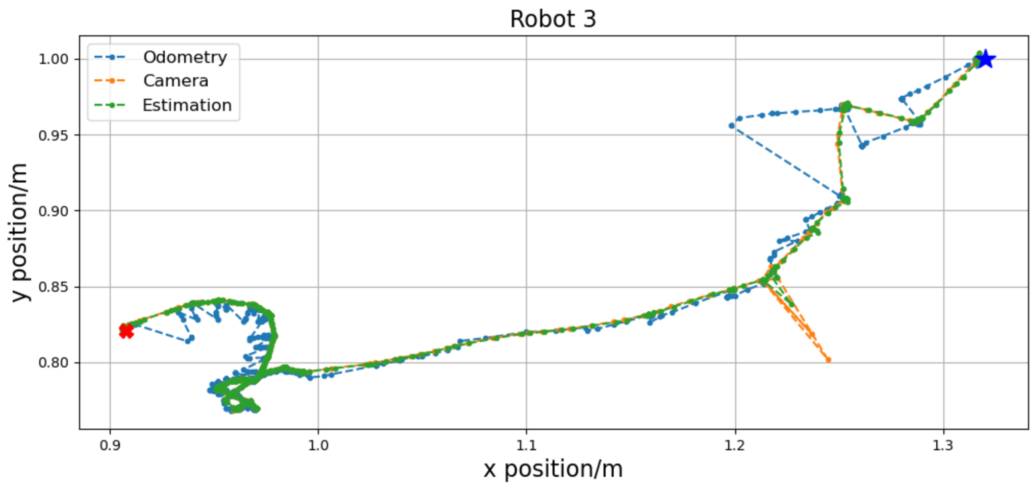

Figure A-22: Robot 3 Trajectory Rectangular Formation with ETC

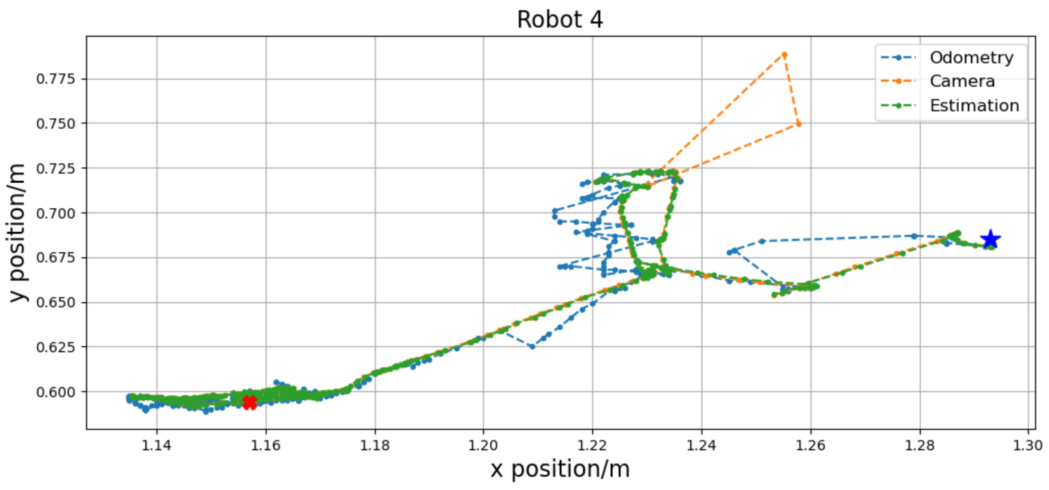

Figure A-23: Robot 4 Trajectory Rectangular Formation with ETC

A-6 Real Experiment, Rectangular Formation with PETC

Figure A-24 provides an overview of the trajectories under PETC, each robot's trajectory can be found in Appendix A-6.

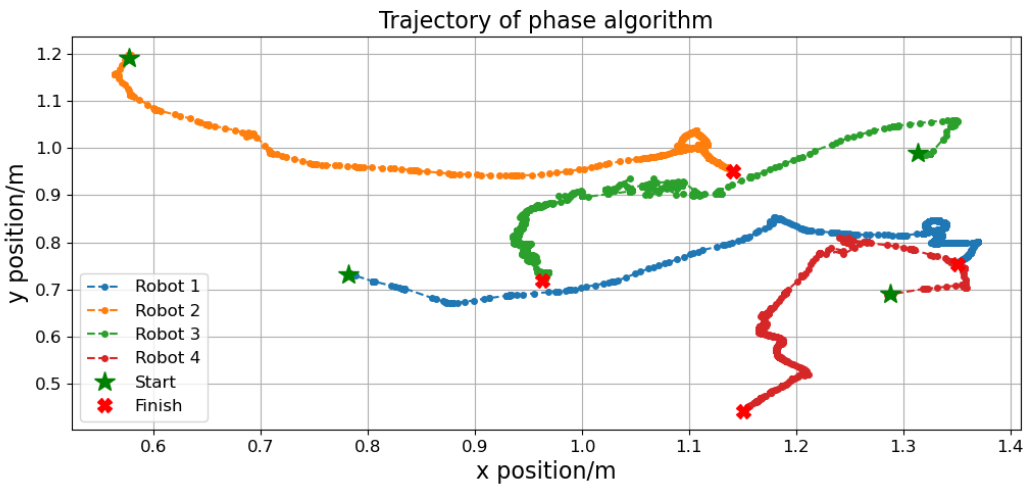

Figure A-24: Robots Trajectory Rectangular Formation with PETC

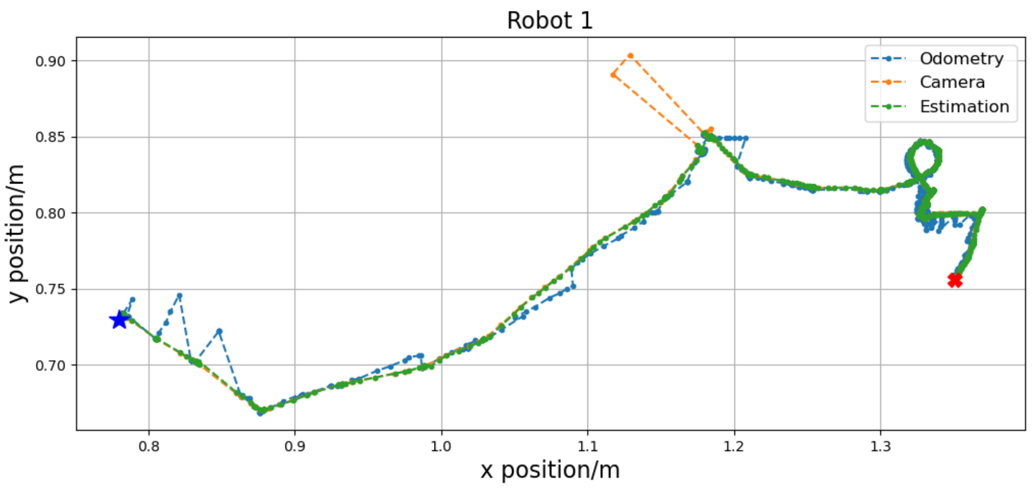

Figure A-25: Robot 1 Trajectory Rectangular Formation with PETC

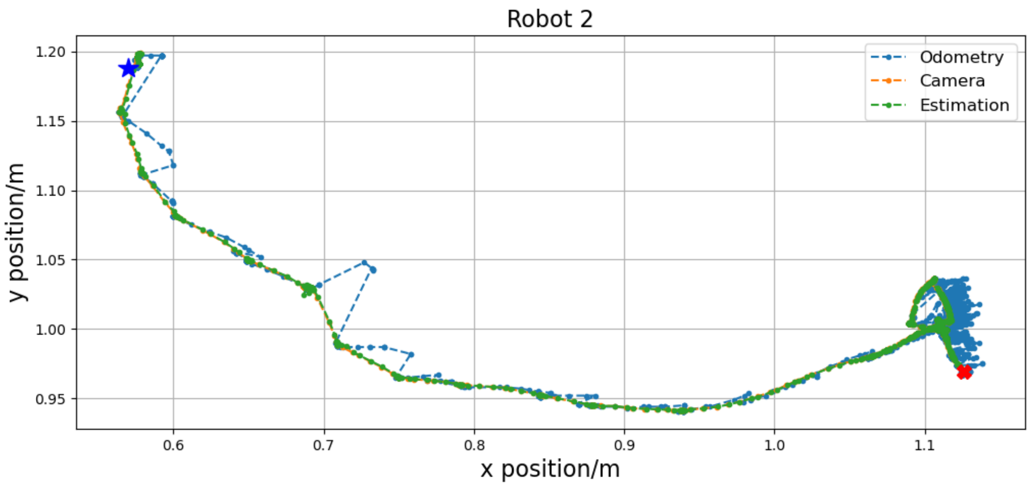

Figure A-26: Robot 2 Trajectory Rectangular Formation with PETC

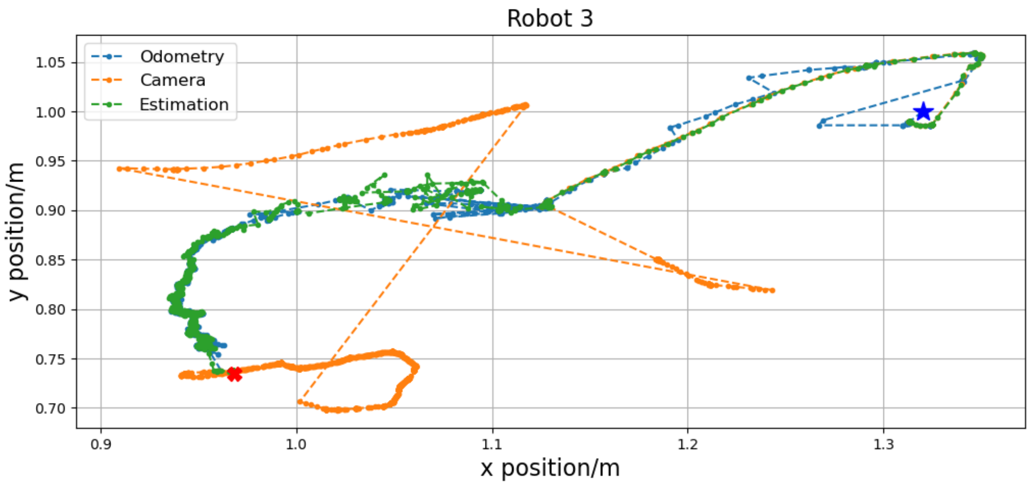

Figure A-27: Robot 3 Trajectory Rectangular Formation with PETC

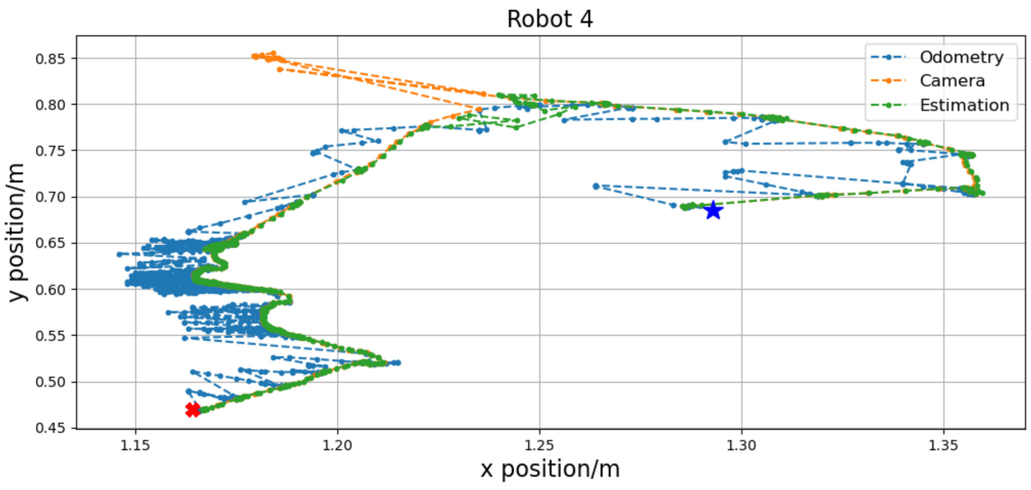

Figure A-28: Robot 4 Trajectory Rectangular Formation with PETC

A-7 Real Experiment, Rectangular Formation with PA

Figure A-29 gives an overview of the trajectories under PA, each robot's trajectory can be found in Appendix A-7.

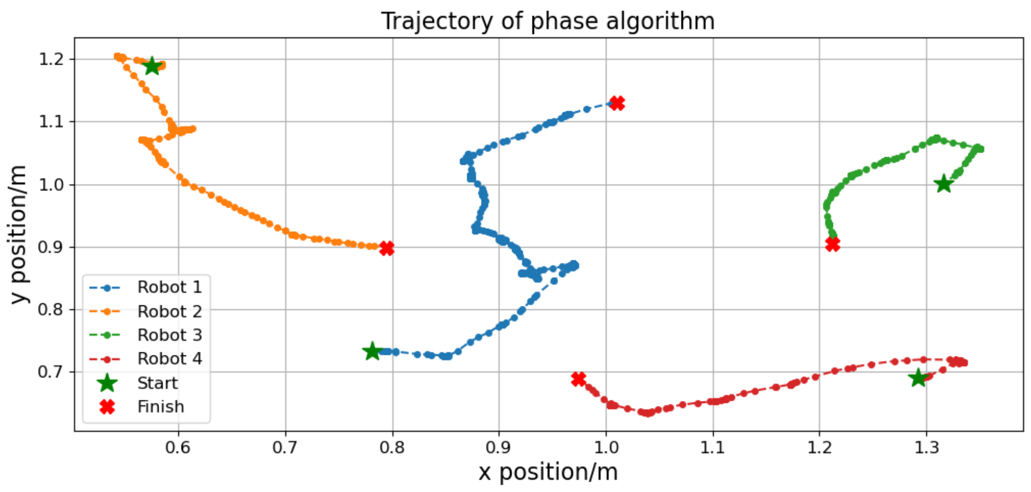

Figure A-29: Robots Trajectory Rectangular Formation with PA

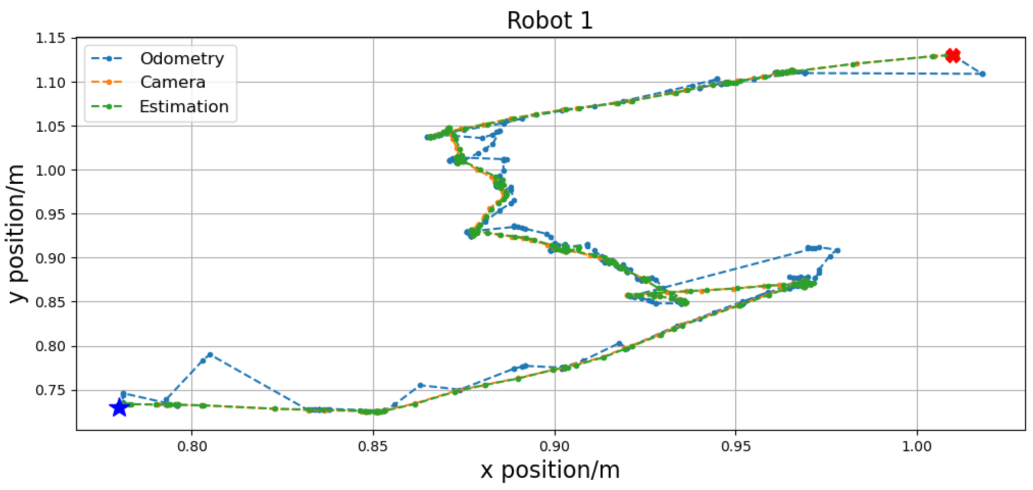

Figure A-30: Robot 1 Trajectory Rectangular Formation with PA

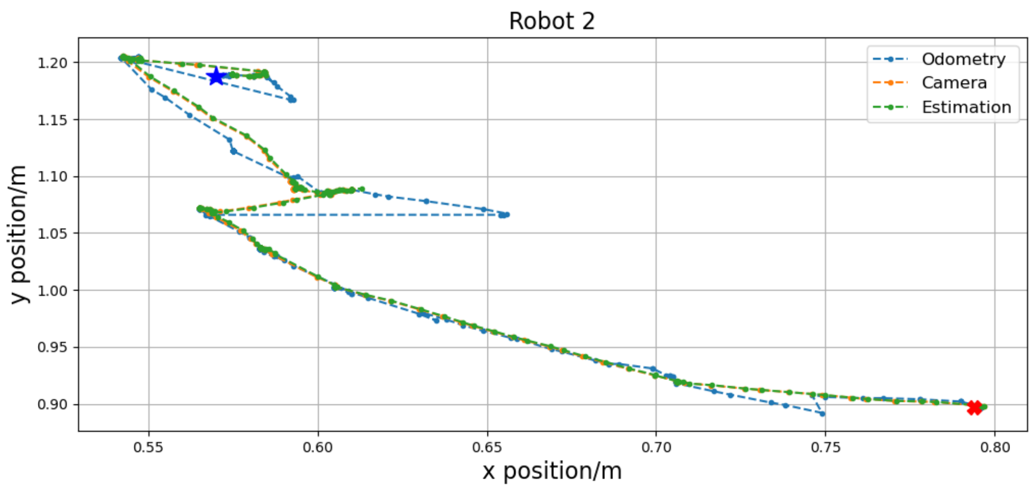

Figure A-31: Robot 2 Trajectory Rectangular Formation with PA

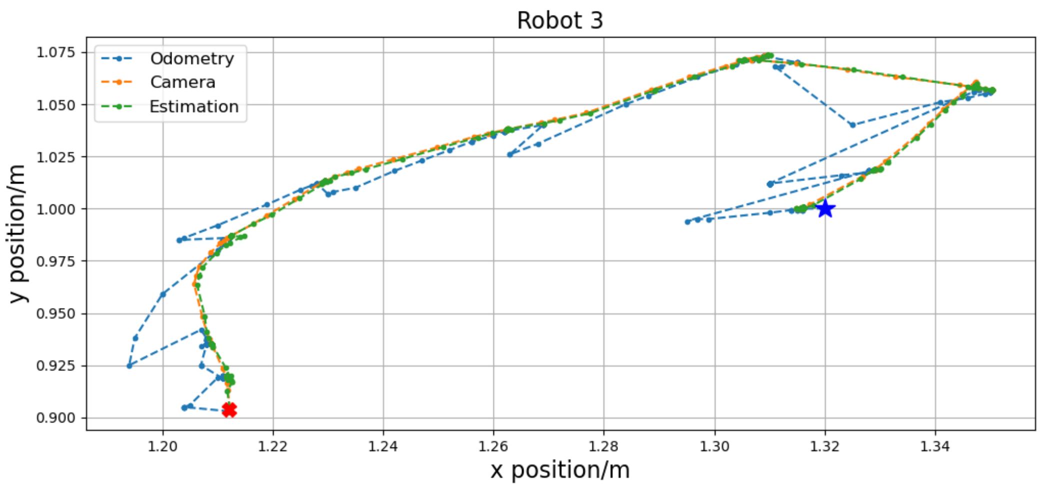

Figure A-32: Robot 3 Trajectory Rectangular Formation with PA

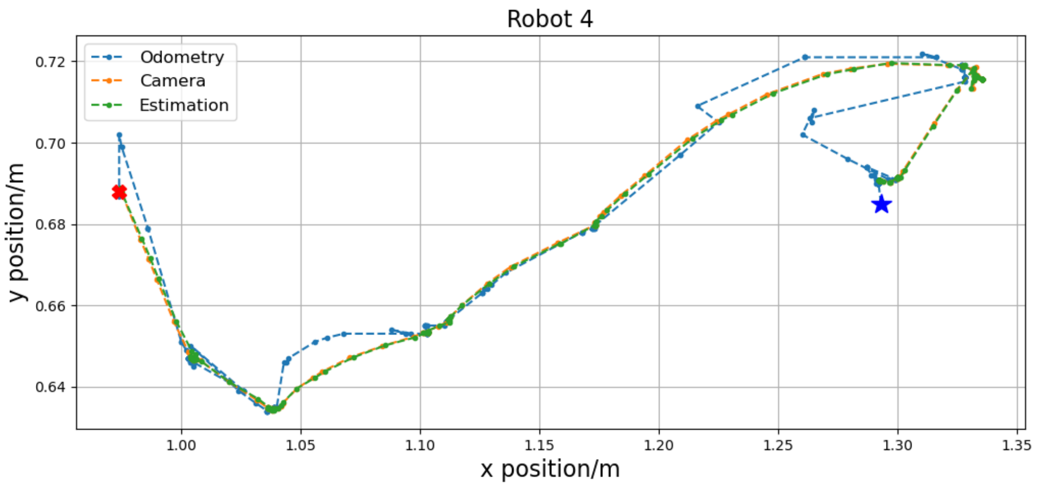

Figure A-33: Robot 4 Trajectory Rectangular Formation with PA

Bibliography

[1] F. Castanedo, “A review of data fusion techniques,” *The scientific world journal*, vol. 2013, 2013.

[2] R. E. Kalman, “A new approach to linear filtering and prediction problems,” 1960.

[3] J. Sasiadek and P. Hartana, “Sensor data fusion using kalman filter,” in *Proceedings of the Third International Conference on Information Fusion*, vol. 2, pp. WED5–19, IEEE, 2000.

[4] F. Caron, E. Duflos, D. Pomorski, and P. Vanheeghe, “Gps/imu data fusion using multisensor kalman filtering: introduction of contextual aspects,” *Information fusion*, vol. 7, no. 2, pp. 221–230, 2006.

[5] S. F. Schmidt, “Application of state-space methods to navigation problems,” in *Advances in control systems*, vol. 3, pp. 293–340, Elsevier, 1966.

[6] L. Jetto, S. Longhi, and G. Venturini, “Development and experimental validation of an adaptive extended kalman filter for the localization of mobile robots,” *IEEE Transactions on Robotics and Automation*, vol. 15, no. 2, pp. 219–229, 1999.

[7] S. Safari, F. Shabani, and D. Simon, “Multirate multisensor data fusion for linear systems using kalman filters and a neural network,” *Aerospace Science and Technology*, vol. 39, pp. 465–471, 2014.

[8] M. Kordestani, M. Dehghani, B. Moshiri, and M. Saif, “A new fusion estimation method for multi-rate multi-sensor systems with missing measurements,” *IEEE Access*, vol. 8, pp. 47522–47532, 2020.

[9] S. Huang and G. Dissanayake, “Robot localization: An introduction,” *Wiley Encyclopedia of Electrical and Electronics Engineering*, pp. 1–10, 1999.

[10] P. K. Panigrahi and S. K. Bisoy, "Localization strategies for autonomous mobile robots: A review," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, Part B, pp. 6019–6039, 2022.

[11] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[12] C. Chen, X. Lu, A. Markham, and N. Trigoni, "Ionet: Learning to cure the curse of drift in inertial odometry," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, Apr. 2018.

[13] G. Hu, J. Shao, F. Liu, Y. Wang, and H. T. Shen, "If-matching: Towards accurate map-matching with information fusion," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 1, pp. 114–127, 2016.

[14] J. Prieto, S. Mazuelas, A. Bahillo, P. Fernandez, R. M. Lorenzo, and E. J. Abril, "Adaptive data fusion for wireless localization in harsh environments," *IEEE Transactions on Signal Processing*, vol. 60, no. 4, pp. 1585–1596, 2012.

[15] R. F. Brena, A. A. Aguileta, L. A. Trejo, E. Molino-Minero-Re, and O. Mayora, "Choosing the best sensor fusion method: A machine-learning approach," *Sensors*, vol. 20, no. 8, 2020.

[16] S. Soltani, M. Kordestani, P. K. Aghaee, and M. Saif, "Improved estimation for well-logging problems based on fusion of four types of kalman filters," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 2, pp. 647–654, 2017.

[17] P. Tsinganos and A. Skodras, "On the comparison of wearable sensor data fusion to a single sensor machine learning technique in fall detection," *Sensors*, vol. 18, no. 2, 2018.

[18] D. Hall and J. Llinas, "An introduction to multisensor data fusion," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 6–23, 1997.

[19] B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi, "Multisensor data fusion: A review of the state-of-the-art," *Information Fusion*, vol. 14, no. 1, pp. 28–44, 2013.

[20] H. Zhu, Y. Li, J. Yu, H. Leung, and Y. Li, "Ensemble registration of multisensor images by a variational bayesian approach," *IEEE Sensors Journal*, vol. 14, no. 8, pp. 2698–2705, 2014.

[21] H. Seraji and N. Serrano, "A multisensor decision fusion system for terrain safety assessment," *IEEE Transactions on Robotics*, vol. 25, no. 1, pp. 99–108, 2009.

[22] C. Chen, R. Jafari, and N. Kehtarnavaz, "Improving human action recognition using fusion of depth camera and inertial sensors," *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 1, pp. 51–61, 2014.

[23] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *Signal processing, sensor fusion, and target recognition VI*, vol. 3068, pp. 182–193, Spie, 1997.

[24] D. Crisan and A. Doucet, "A survey of convergence results on particle filtering methods for practitioners," *IEEE Transactions on signal processing*, vol. 50, no. 3, pp. 736–746, 2002.

[25] L. Teslić, I. Škrjanc, and G. Klančar, "Ekf-based localization of a wheeled mobile robot in structured environments," *Journal of Intelligent & Robotic Systems*, vol. 62, pp. 187–203, 2011.

[26] Z. Kurt-Yavuz and S. Yavuz, "A comparison of ekf, ukf, fastslam2. 0, and ukf-based fastslam algorithms," in *2012 IEEE 16th International Conference on Intelligent Engineering Systems (INES)*, pp. 37–43, IEEE, 2012.

[27] A. Giannitrapani, N. Ceccarelli, F. Scortecci, and A. Garulli, "Comparison of ekf and ukf for spacecraft localization via angle measurements," *IEEE Transactions on aerospace and electronic systems*, vol. 47, no. 1, pp. 75–84, 2011.

[28] L. D'Alfonso, W. Lucia, P. Muraca, and P. Pugliese, "Mobile robot localization via ekf and ukf: A comparison based on real data," *Robotics and Autonomous Systems*, vol. 74, pp. 122–127, 2015.

[29] L. Yan, B. Liu, and D. Zhou, "The modeling and estimation of asynchronous multirate multisensor dynamic systems," *Aerospace Science and Technology*, vol. 10, no. 1, pp. 63–71, 2006.

[30] B. Chen, W.-A. Zhang, and L. Yu, "Distributed fusion estimation with missing measurements, random transmission delays and packet dropouts," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1961–1967, 2014.

[31] M. Kordestani, M. Dehghani, B. Moshiri, and M. Saif, "A new fusion estimation method for multi-rate multi-sensor systems with missing measurements," *Ieee Access*, vol. 8, pp. 47522–47532, 2020.

[32] J. Zhang, Y. Lu, W. Wang, Y. Zhang, and Q. Wu, "Multi-sensor data fusion based on electro-optical tracking system," in *4th International Symposium on Advanced Optical Manufacturing and Testing Technologies: Optical Test and Measurement Technology and Equipment*, vol. 7283, pp. 584–589, SPIE, 2009.

[33] X. Zhai, H. Jing, and T. Vladimirova, "Multi-sensor data fusion in wireless sensor networks for planetary exploration," in *2014 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, pp. 188–195, IEEE, 2014.

[34] C. Li, S. Wang, Y. Zhuang, and F. Yan, "Deep sensor fusion between 2d laser scanner and imu for mobile robot localization," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 8501–8509, 2019.

[35] C. E. Magrin and E. Todt, "Multi-sensor fusion method based on artificial neural network for mobile robot self-localization," in *2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE)*, pp. 138–143, 2019.

[36] J. Ghommam, H. Mehrjerdi, M. Saad, and F. Mnif, "Formation path following control of unicycle-type mobile robots," *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 727–736, 2010.

[37] H. A. Patel and D. G. Thakore, "Moving object tracking using kalman filter," *International Journal of Computer Science and Mobile Computing*, vol. 2, no. 4, pp. 326–332, 2013.

[38] T. H. Dinh, M. D. Phung, T. H. Tran, and Q. V. Tran, "Localization of a unicycle-like mobile robot using LRF and omni-directional camera," *CoRR*, vol. abs/1611.09431, 2016.

[39] J. N. Greenberg and X. Tan, "Dynamic optical localization of a mobile robot using kalman filtering-based position prediction," *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 5, pp. 2483–2492, 2020.

[40] Q. Li, R. Li, K. Ji, and W. Dai, "Kalman filter and its application," in *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, pp. 74–77, IEEE, 2015.

[41] C. K. Chui, G. Chen, *et al.*, *Kalman filtering*. Springer, 2017.

[42] L. Armesto, S. Chroust, M. Vincze, and J. Tornero, "Multi-rate fusion with vision and inertial sensors," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, vol. 1, pp. 193–199 Vol.1, 2004.

[43] A. Smyth and M. Wu, "Multi-rate kalman filtering for the data fusion of displacement and acceleration response measurements in dynamic system monitoring," *Mechanical Systems and Signal Processing*, vol. 21, no. 2, pp. 706–723, 2007.

[44] L. Yan, D. Zhou, M. Fu, and Y. Xia, "State estimation for asynchronous multirate multisensor dynamic systems with missing measurements," *IET Signal Processing*, vol. 4, no. 6, pp. 728–739, 2010.

[45] R. R. Yager, "On ordered weighted averaging aggregation operators in multicriteria decisionmaking," *IEEE Transactions on systems, Man, and Cybernetics*, vol. 18, no. 1, pp. 183–190, 1988.

[46] A. K. Nasir and H. Roth, "Pose estimation by multisensor data fusion of wheel encoders, gyroscope, accelerometer and electronic compass," *IFAC Proceedings Volumes*, vol. 45, no. 4, pp. 49–54, 2012. 1st IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control.

[47] I. Zunaidi, N. Kato, Y. Nomura, and H. Matsui, "Positioning system for 4-wheel mobile robot: Encoder, gyro and accelerometer data fusion with error model method," 2006.

[48] 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.

[49] E. Zwetsloot, "Consensus Algorithms for Single-Integrator Multi-Agent Systems," Master's thesis, Delft University of Technology, 2023.