2021

A PHD FILTER BASED RELATIVE LOCALIZATION FILTER FOR ROBOTIC SWARMS

Rupasinghe Thivanka Perera
University of Rhode Island, thiva@uri.edu

A PHD FILTER BASED RELATIVE LOCALIZATION FILTER FOR ROBOTIC SWARMS

BY

RUPASINGHE ARACHCHIGE THIVANKA NYOMAL PERERA

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

ELECTRICAL ENGINEERING

UNIVERSITY OF RHODE ISLAND

2021

MASTER OF SCIENCE THESIS

OF

RUPASINGHE ARACHCHIGE THIVANKA NYOMAL PERERA

APPROVED:

Thesis Committee:

Major Professor	Paolo Stegagno
	Richard Vaccaro
	Chengzhi Yuan
	Brenton DeBoef DEAN OF THE GRADUATE SCHOOL

UNIVERSITY OF RHODE ISLAND

2021

ABSTRACT

In this thesis, we present a Probability Hypothesis Density (PHD) filter based relative localization system for robotic swarms. The system is designed to use only local information collected by onboard lidar and camera sensors to identify and track other swarm members within proximity. The multi-sensor setup of the system accounts for the inability of single sensors to provide enough information for the simultaneous identification of teammates and estimation of their position. However, it also requires the implementation of sensor fusion techniques that do not employ complex computer vision or recognition algorithms, due to robots' limited computational capabilities. The use of the PHD filter is fostered by its inherent multi-sensor setup. Moreover, it aligns well with the overall goal of this localization system and swarm setup that does not require the association of a unique identifier to each team member. The system was tested on a team of four robots. This thesis content was accepted to DARS-SWARM 2021 conference [1].

ACKNOWLEDGMENTS

First and foremost I would like to acknowledge my supervisor and advisor Professor Paolo Stegagno for his immense support, mentorship during this project and throughout my graduate career. Thank you for the countless opportunities you have given me. Additionally, I would like to acknowledge my committe members Professor Richard J Vaccaro, Professor Chengzhi Yuan and committee chair Professor Manbir Sodhi. My sincere thank goes to all the URI staff who have helped me in many ways to make this research a success.

I wish to thank my loving wife Nuwanthi for providing me continuous love, encouragement and support throughout my entire graduate career and beyond. Without your support, it wouldn't have been possible. Also, I wish to thank my parents and my sister for their extended love and support. Finally, I wish to thank my friends who have been providing their support.

TABLE OF CONTENTS

ABSTRACT ii

ACKNOWLEDGMENTS iii

TABLE OF CONTENTS iv

LIST OF FIGURES vi

CHAPTER

1 Introduction 1

1.1 Localization methods in a Swarm 2

1.2 PHD filter approach 4

1.3 Statement of the problem 5

2 Problem Setting and Background 6

2.1 Multi-sensor PHD filter 8

3 PHD-Filter Based Relative Localization Module 11

3.1 Time Update 12

3.2 Lidar Measurement Update 14

3.3 Camera Measurement Update 18

4 Simulation 19

4.1 Simulation results 20

5 Experimental Setup 23

5.1 UGV Platform 24

5.2 Main Processing unit 25

Page

5.3 Odometry system 26

5.4 UGV Sensors 27

5.4.1 Lidar sensor 28

5.4.2 Omni-directional Cameras 29

5.5 The testing area 30

5.6 Lidar Measurements 31

5.7 Camera Measurements 32

6 Experimental Results 34

7 Conclusion 36

LIST OF REFERENCES 37

BIBLIOGRAPHY 41

LIST OF FIGURES

Figure **Page**

1 Left: the robotic swarm used to validate our localization system. Center: one of the robots used in this work. Right: a representation of the problem setting: triangles are robots, false positives are circles, × are lidar measurements l_k^h, straight dashed lines are camera measurements c_k^h, blind spots in the lidar measurements are represented as shaded areas. 7

2 Filter schematic 11

3 Probaility of Survival P_s^i 13

4 Probability of detection for blind spots created by camera structure 16

5 Robots block each other 17

6 Gazebo 20 UGVs simulation 19

7 Percentage of time for which each UGV error was greater than 30 cm 21

8 No of components estimated by the filter 22

9 UGV paths during the simulation - LC method 22

10 Final Robot Design 23

11 Robot Platform - DFRobot Cherokey [2] 24

12 Inside UGV 25

13 Odroid-XU4 25

14 Motor with Encoder 26

15 UGV Front View 27

16 YD-Lidar X4 [3] 28

17 Camera Sensor 29

Figure **Page**

18 Front and rear camera images 29

19 Testing area 30

20 Points provided by a typical lidar scan during the experiment. visualized using Rviz 31

21 ROS message - Single Lidar scan 32

22 Image sensor 33

23 Distance error of the three UGVs in the LC (left) and LO (right) experiments. 35

24 Comparison between the LO and LC experiments. Left: sum of the weights of all the components with LO (blue) and LC (red). Right: percentage of time that the error on the position of each robot is greater than 30cm with LO and LC. 35

CHAPTER 1

Introduction

In recent years, robotic swarms have been receiving increasing attention thanks to many potential applications [4]. Tasks as target search and tracking [5], search and rescue [6], exploration [7], information gathering, clean up of toxic spills [8, 9], and construction [10] have been proposed throughout the years.

A robotic swarm is defined as a group of low cost, relatively simple robots that intends to perform tasks in an unknown/undiscovered environments. In a swarm, each individual member performs its own task and collectively the swarm intends to achieve the main goal.

Robotics swarms are highly depend on its member locations. Having a centralised common positioning system for a swarm is not feasible in the above mentioned environments, hence each member must have the knowledge of other members in its attached frame of reference. This knowledge is gained by utilizing its own sensors without depending on an external system.

Most works focus on control algorithms; however, many control laws and collaborative swarm behaviors require the ability to identify other robots in the environment, and compute an estimate of their position.

To retrieve this information many localization algorithms in different operative conditions have been proposed for multi-robot systems. In cooperative localization (e.g., [11]), the robots communicate each other's odometry and relative measurements to compute the location of each team member in a common frame of reference, usually through an online Bayesian filter (e.g., [12, 13]) or estimator (e.g., [14]). However, the assumption of a common frame of reference accounts as a form of centralization and should be avoided in a robotic swarm setting.

1.1 Localization methods in a Swarm

In relative localization algorithms, the assumption of a common frame of reference is eased and the goal of each robot is to estimate the pose of other robots in its attached frame of reference. This has been addressed through Bayesian filters [15], geometrical arguments [16], or a combination of both [17]. Usually, both relative and cooperative localization algorithms require not only some position, bearing or distance measurements, but also that each measurement comes with the unique identifier of the measured robot. However, unique identification of each robot could be difficult or undesirable. Typical approaches include visually tagging each robot and extracting the tag through cameras [18], using dedicated infrared systems [19] or RFIDs [20]. Tagging and ID exchange in many cases are not viable solutions. It could be technically unfeasible, particularly in case of large numbers of robots or with sensors, as laser scanners, that do not allow for unique identification capabilities. It also accounts as a form of centralization, meaning that all robots need to know the same set of IDs. Last but not least, it may jeopardize the task to make explicit the identity of each robot, if the swarm is for example in an escorting or disguising mission.

In a number of papers, the problem of computing an estimate of other robot's location with untagged measurements has been referred to as localization with anonymous measurements [17, 21], or unknown data association. In [21], using odometry and untagged relative measurements communicated from other robots, the robots were able to produce tagged (i.e., associated with a unique identifier) relative pose estimates.

However, associating ids to each robot is not a mandatory condition to perform cooperative tasks as formation control [22], encircling [23], and connectivity maintenance [24], as long as each robot is able to identify that some entities in the environment are generically teammates, and compute an estimate of their relative positions. Moreover, in a robotic swarm setup, robots could have limited or no communication capabilities, and should rely only on local self-gathered measurements to perform their tasks.

In this situation, the choice of the sensor equipment endowed to the robots becomes even more crucial. On the one hand, the sensors should provide enough information to allow (non-unique) identification of other robots, and quantitative estimation of their relative position. On the other hand, robotic swarms are usually composed of relatively small and cheap robots, featuring limited computational capabilities that are not compatible with expensive informative rich sensor equipment. Single sensor approaches are limited by the sensing technology. Using distance sensors as lidars, robots can be easily mistaken for obstacles of similar size, and vice versa. On the other hand, camera sensors would be able to identify robots more reliably, but they would directly provide only bearing information, and distance estimates could be affected by consistent noise, have long convergence time, and require persisting excitation conditions [25, 26]. RGB-D sensors offer the best of both worlds but usually have limited fields of view, while the robots should be aware of teammates and obstacles in entirety of their surroundings.

1.2 PHD filter approach

Given the multi-target multi-sensor tracking nature of the proposed problem, a natural choice would be the employment of a Probability Hypothesis Density (PHD) filter. The PHD filter was first proposed in [27] as a recursive filter for multi-target multi-sensor tracking. The filter in its theoretical form would require infinite computational power. However, some authors have proposed Gaussian mixture [28] and particle based implementations [29] among others.

PHD filters have already been employed in multi-robot localization. In [30], the authors presented a PHD filter to incorporate absolute poses exchanged by robots and local sensory measurement to maintain robots' formation when communication fails. In [31], a team of mobile sensors was employed to cooperatively localize an unknown number of targets via PHD filter. However, in these two works a common frame of reference was assumed, which is not compatible with our setup. In [32], the authors implemented a PHD filter to fuse ground robot (UGV) odometry and aerial camera measurements to estimate the location and identity of the UGVs. However, only the aerial robot computes the position of the other robots, and not every team member. In [33], the authors used two different visual features to describe the target of interest enhancing the PHD-based tracking. However, this is an example of video tracking and the metric pose of the targets are not estimated. Therefore, none of the setups discussed in literature is compatible with the needs of a robotic swarm and our settings.

1.3 Statement of the problem

In this thesis we propose a multi-model approach in which we employ multiple sensors, fisheye cameras and laser scanners, to combine the recognition capability of the first with the accuracy of the second. However, this approach requires non-trivial data fusion techniques. Hence we propose a novel robo-centric implementation of the PHD filter for the fusion of lidar and camera measurements in a swarm setup.

The proposed filter design runs independently on each robot to compute estimates of other teammates position in its attached frame of reference. The measurements for the filter will be obtained using on-board lidar and camera sensors and will not depend on any external sensor data. The filter computations are done in the on-board main processing unit, which computes estimates in real time.

The filter will be first tested in a simulated robotic swarm where each swarm member is equipped with above mentioned sensors. Then we develop a UGV (Unmanned ground vehicle) to test the filter in a real robotic swarm. Each UGV will be equipped with said sensors and a processing unit as per filter requirement. The issues that arose during the experiment will be investigated and will be addressed by modifying filter parameters.

CHAPTER 2

Problem Setting and Background

The system we consider (Figure 1) consists of n UGVs $\{\mathcal{R}^1, \mathcal{R}^2, ..., \mathcal{R}^n\}$ in a 2D space, with n unknown and time-variant. The generic robot $\mathcal{R}^j$ is modeled as a rigid body moving in 2D space and is equipped with an attached reference frame $\mathcal{F}_j = \{O_j, X_j, Y_j\}$ whose origin coincides with a representative point of the robot. Let $q_h^j \in \mathbb{R}^2$ be and $\psi_h^j \in SO(1)$ respectively the position and orientation of $\mathcal{R}^h$ in $\mathcal{F}_j$, and let o_h^j be the position of $\mathcal{O}_h$ in $\mathcal{F}_j$. In the following, we indicate with $R(\phi)$ the elementary 2D rotation matrix of an angle ϕ:

$$R(\phi) = \left[\begin{array}{cc} cos(\phi) & -sin(\phi) \\ sin(\phi) & cos(\phi) \end{array} \right]. \tag{1}$$

Robot $\mathcal{R}^j$ is equipped with multiple sensors. First, the odometry module of $\mathcal{R}^j$ provides, at each time k, a measurement $U_k^j = [\Delta x_k^j \; \Delta y_k^j \; \Delta \psi_k^j]^T \in \mathbb{R}^2 \times SO(1)$ of the robot linear and angular displacement between two consecutive sampling instants $k-1$ and k on the XY plane.

$\mathcal{R}^j$ is also equipped with a lidar sensor. Lidar measurements are processed with a feature extraction algorithm that identifies all objects in the scan (including robots) whose size is comparable with the size of the robots. In general, we assume that there is an unknown number of objects in the environment that will be detected in the lidar as possible robots. Therefore, at each time step k the algorithm provides a set of l_k relative position measurements $L_k = \{l_k^1, ..., l_k^{l_k}\}$ in $\mathcal{F}_j$, representing the position of robots or obstacles in the field of view of the sensor. The sensor is affected by false positive (some measurements may not refer to actual objects) and false negative measurements (some object or robot may not be detected) due to obstructions and errors of the feature extraction algorithm.

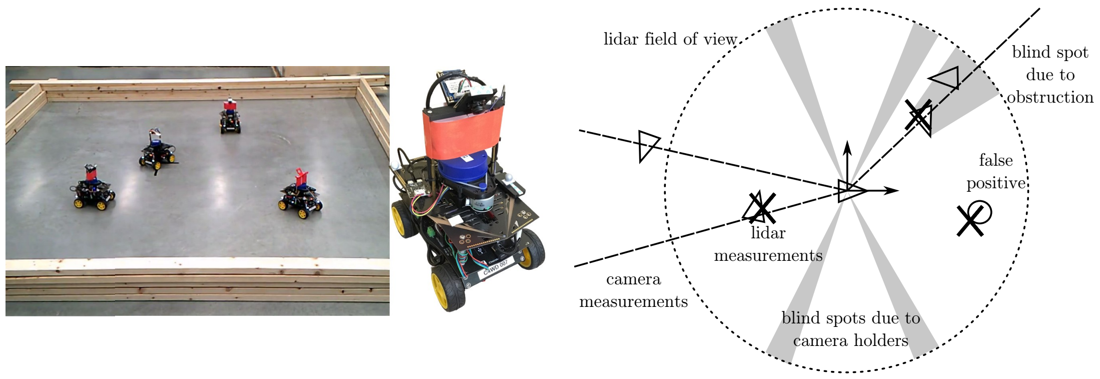


Figure 1: Left: the robotic swarm used to validate our localization system. Center: one of the robots used in this work. Right: a representation of the problem setting: triangles are robots, false positives are circles, $\times$ are lidar measurements l_k^h, straight dashed lines are camera measurements c_k^h, blind spots in the lidar measurements are represented as shaded areas.

Lastly, two fisheye cameras are mounted on $\mathcal{R}^j$, one oriented towards the front of the robot, and one towards the back. This setup allows us to identify robots in a 360° field of view. The images from the cameras are processed using a feature extraction algorithm that has the capability of identifying a generic robot based on color. The algorithm does not uniquely identify and label each robot. At time step k, the cameras provide a set of c_k bearing measurements $C_k = \{c_1, ...c_k^{c_k}\}$ in $\mathcal{F}_j$. Also in this case, there may be false positive (non-robots identified as robots) and false negative (missed robot detections) measurements. In the following, the camera and lidar measurements collected at time k will be denoted together as $Z_k = \{L_k, C_k\}$. Note that camera and lidar have different rates and in general are not synchronized. Therefore, without loss of generality, for some k it may be either $L_k = \emptyset$, or $C_k = \emptyset$, or both. A representation of all sensor readings is provided in Figure 1.

The objective of $\mathcal{R}^j$ is to compute at each time step k an estimate of the number $n(k)$ and positions of all robots in the environment.

2.1 Multi-sensor PHD filter

This Section provides the necessary background on the PHD filter and is mostly based on [27, 28, 29]. Assuming that there are n (with n unknown and variable over time) targets living in a space $\mathcal{X}$, the goal of the standard PHD filter is to compute an estimate of the PHD of targets in $\mathcal{X}$. The PHD $f_k(x)$ at time k is defined as the function such that its integral over any subset $S \subseteq \mathcal{X}$ is the expected number of targets $N(S)$ in that subset, i.e., $N(S) = \int_S f_k(x)dx$.

The PHD filter is a recursive estimator composed of two main steps: a time update and a measurement update. The time update is meant to produce a prediction of the PHD $f_{k|k-1}(x)$ at time step k given the estimate $f_{k-1|k-1}(x)$ at time $k-1$, through the time update equation:

$$f_{k|k-1} = b_{k|k-1}(x) + \int [P_s(x')f_{k|k-1}(x|x') + b_{k|k-1}(x|x')]f_{k-1|k-1}(x')dx' \tag{2}$$

where $b_{k|k-1}(x)$ is the probability that a new target appears in x between times $k-1$ and k, $P_s(x')$ is the probability that a target in x' at time $k-1$ will survive into step k, $f_{k|k-1}(x|x')$ is the probability density that a target in x' moves to x, and $b_{k|k-1}(x|x')$ is the probability that a new target spawns in x at time k from a target in x' at time $k-1$.

Note that both $f_{k-1|k-1}(x)$ and $f_{k|k-1}(x)$ are computed considering only the measurements up to time $k-1$. Measurements Z_k collected at time k are incorporated in the estimate through the measurements update to compute the posterior PHD:

$$\begin{aligned} & f_{k|k}(x) = \\ & f_{k|k-1}(x)\left[1 - P_D(x) + \sum_{z \in Z_k} \frac{P_D(x)g(z|x)}{\lambda c(z) + \int P_D(x')g(z|x')f_{k|k-1}(x')dx'}\right] \end{aligned} \tag{3}$$

where $P_D(x)$ is the probability that an observation is collected from a target with state x, $g(z|x)$ is the sensor likelihood function, and $\lambda c(z)$ expresses the probability that a given measurement z is a false positive.

Although elegant, equations (2) and (3) cannot be implemented in practice for generic functions. A popular approximation, the Gaussian Mixture PHD filter (GM-PHD) considers all PHD functions $f_{k-1|k-1}(x)$, $f_{k|k-1}(x)$, and $f_{k|k}(x)$ to be sums of weighted Gaussian functions in the form:

$$f_{*|*}(x) = \sum_i f^i_{*|*}(x) = \sum_i w^i_{*|*} \mathcal{N}(x; m^i_{*|*}, p^i_{*|*}) \tag{4}$$

where $f^i_{*|*}(x)$ is the generic i-th component, $w^i_{*|*}$, $m^i_{*|*}$, and $p^i_{*|*}$ are respectively the weight, mean and covariance matrix of the i-th component. Introducing the GM representation (4) in equation (2), and assuming that the probability of survival can be approximated as a constant for each component ($P_s(x') \simeq P^i_s$), the spawning probability $b_{k|k-1}(x|x')$ is zero, and the system model $f_{k|k-1}(x|x')$ and target birth probability $b_{k|k-1}(x)$ are Gaussian functions, the GM-PHD filter time update equation becomes:

$$f_{k|k-1} = b_{k|k-1}(x) + \sum_i w^i_{k-1|k-1} P^i_s \int f_{k|k-1}(x|x') f^i_{k-1|k-1}(x') dx' \tag{5}$$

Therefore, the PHD prediction will have a component for each component in the PHD posterior $f_{k-1|k-1}(x)$. Moreover, the integral term will be the same as a prediction step of the standard Kalman filter, so every component of $f_{k|k-1}(x)$ will be a Gaussian function, and it will be possible to compute the PHD prediction by simply applying component-wise the time update of a Kalman filter.

Introducing the GM representation (4) in equation (3), assuming that the probability of detection can be approximated as a constant $P_D(x) \simeq P^i_D$ for each component $f^i_{k+1|k}(x)$, and $c(z) = 0$, the GM-PHD filter measurement update equation becomes:

$$f_{k|k}(x) = \sum_i f^i_{k|k-1}(x) \left(1 - P^i_D\right) + \sum_i \sum_{z \in Z_k} \frac{P^i_D f^i_{k|k-1}(x) g(z|x)}{\sum_i \int P^i_D g(z|x') f^i_{k|k-1}(x') dx'}. \tag{6}$$

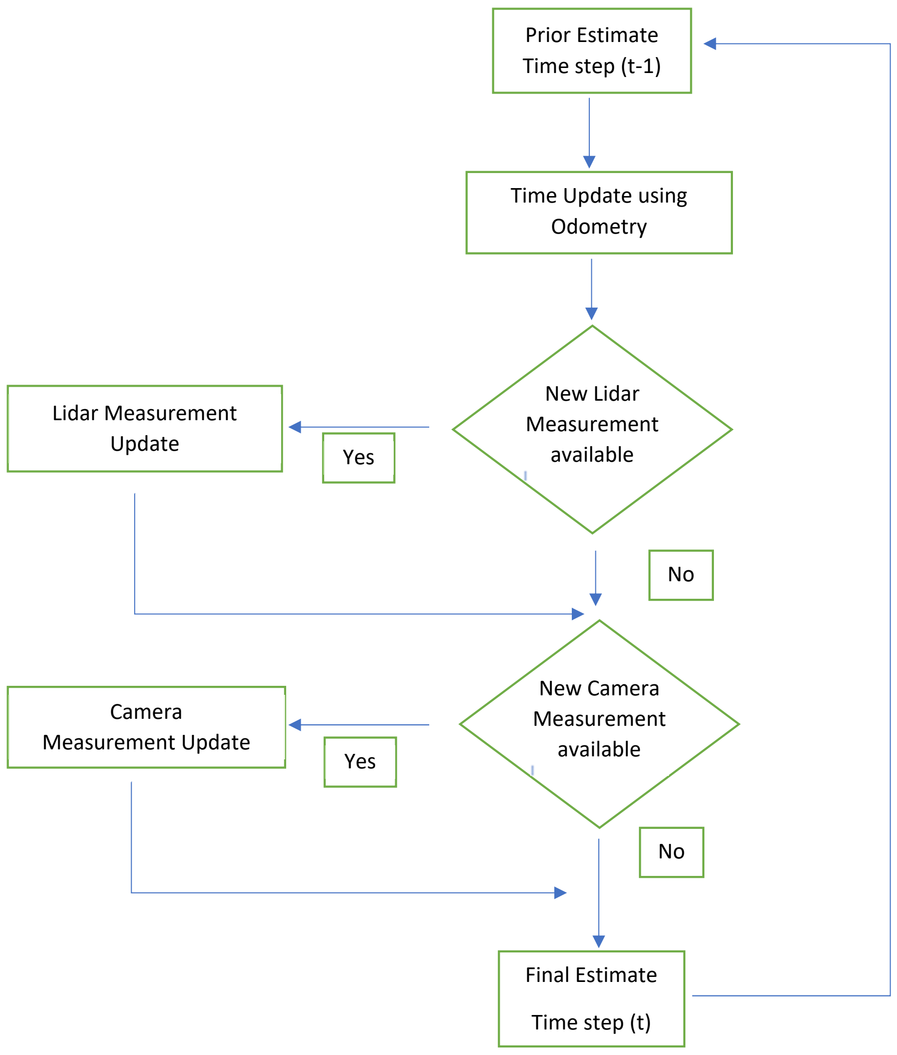
showing that, if Z_k contains m measurements, each component $f^i_{k|k-1}(x)$ generates $m+1$ components in $f_{k|k}(x)$. Moreover, if $g(z|x)$ is a Gaussian function, the last term is a sum of Gaussian functions, each function being the result of a single-component Kalman filter measurement update step.

An additional pruning step is needed to limit the number of components in the PHD. In fact, if all components were kept at each time step, their number would grow exponentially with the number of measurements. Therefore, all components whose weight is below a given threshold at the end of the measurement update are eliminated.

It is clear from its formulation that the PHD filter is inherently multi-sensor. In fact, when multiple sensors are present, multiple measurement updates can be applied consecutively, each one as a component-wise Kalman filter update step.

CHAPTER 3

PHD-Filter Based Relative Localization Module



Prior Estimate Time step (t-1)
Time Update using Odometry
New Lidar Measurement available
Yes
Lidar Measurement Update
No
New Camera Measurement available
Yes
Camera Measurement Update
No
Final Estimate Time step (t)

Figure 2: Filter schematic

Following the scheme presented in the Section 2.1, our localization module consists of a time update step and two measurement update steps. Note that the module is asynchronous, so there is no particular order or sequence in which these steps are performed. While the time update is periodically performed, the measurement updates are performed if and when measurements become available.

The estimated state of the target robots is their position in $\mathcal{F}_j$, $m^i_{k|k} = q^j_*(k) \in \mathbb{R}^2$, where the $*$ expresses the concept that the i-th component may refer to any of the tracked target robots. The covariance of $m^i_{k|k}$ is therefore $p^i_{k|k} \in \mathbb{R}^{2\times 2}$.

3.1 Time Update

During the time update, the owner's $\mathcal{R}^j$ odometry $U^j_k = [\Delta x^j_k \ \Delta y^j_k \ \Delta\psi^j_k]^T$ is used to update the mean and covariance of all components of the PHD. The k^{th} time update for the generic i^{th} component is given by:

$$m^i_{k|k-1} = R(\Delta\psi^j_k)(m^i_{k-1|k-1} - [\Delta x^j_k \ \Delta y^j_k]^T) \tag{7}$$

$$p^i_{k|k-1} = R(\Delta\psi^j_k) p^i_{k-1|k-1} R(\Delta\psi^j_k)^T + R(\Delta\psi^j_k) Q_{k-1} R(\Delta\psi^j_k)^T \tag{8}$$

$$w^i_{k|k-1} = P^i_s w^i_{k-1|k-1} \tag{9}$$

where P^i_s is the survival probability from time step $k-1$ to the time step k of the i^{th} component $f^i_{k-1|k-1}$, and Q_{k-1} is the system noise.

Ideally, the survival probability P^i_s, depends on the real probability that a target disappear. In a robotic swarm context, this probability would be extremely low in the whole domain. Therefore, we have used it as a design parameter to meet the objectives of the localization module. Coherently with the task and motivation of this thesis, only local information is required and available to each robot. Using P^i_s, we prefer to let too far components fade. At this aim, we use an inverse sigmoid function (Figure 3) to compute P^i_s:

$$P^i_s = \frac{1}{(1.05 + e^{4(||m^i_{k-1|k-1}||-4)})} \tag{10}$$

This creates a circular area in around R^j in which it tracks other robots. To account for targets that enters into this area from outside, a birth target component $b_{k|k-1}(x)$ is added at each time update, such that its mean, covariance and weight

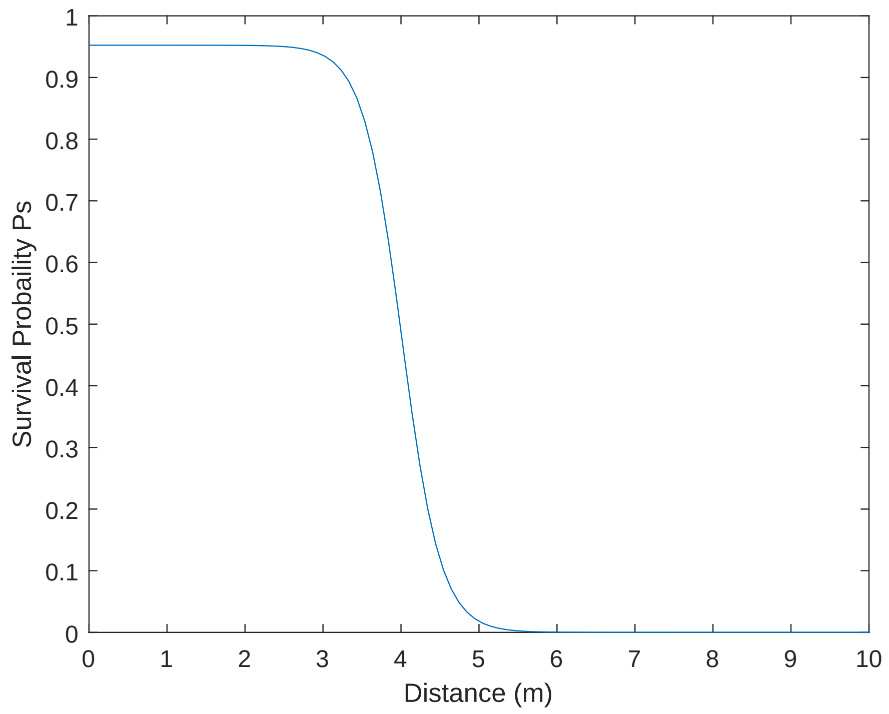


Figure 3: Probaility of Survival P_s^i

are respectively:

$$m^b = \begin{pmatrix} 0 & 0 \end{pmatrix}, \ p^b = \begin{pmatrix} 4 & 0 \\ 0 & 4 \end{pmatrix}, \ w^b = 0.001 \tag{11}$$

The assigned weight is very low so if there is no correspondence with the measurements (i.e., at least one measurement without a good correspondence with one or more components of the PHD prior), $b_{k|k-1}(x)$ will be pruned immediately. The choice of limiting the area in which each robot tracks its teammates is also beneficial for the scalability of the method. In fact, even if the swarm was comprised of hundreds of agents, each robot would only track the ones that are closer to it, therefore linking the computational complexity of the filter to the density of the swarm rather than to the total number of robots.

3.2 Lidar Measurement Update

After the time update, the lidar measurement update is performed only when new measurements are available. We assume that the lidar at time k collect l_k position measurements $l_k^h \in L_k, h = 1, ..., l_k$ in $\mathcal{F}_j$. Following equation (6), each component $f_{k|k-1}^i$ of $f_{k|k-1}$ generates $l_k + 1$ components $f_{k|k}^{i(l_k+1)}$, $f_{k|k}^{i(l_k+1)+h}$, $h = 1, ..., l_k$ in $f_{k|k}$. One component $f_{k|k}^{i(l_{k+1})}$ has the same mean $m_{k|k}^{i(l_{k+1})} = m_{k|k-1}^i$ and same covariance $p_{k|k}^{i(l_{k+1})} = p_{k|k-1}^i$ of the original component, while the weight is updated as:

$$w_{k|k}^{i(l_k+1)} = (1 - {}^L P_d^i) w_{k|k-1}^i \tag{12}$$

where ${}^L P_d^i$ is the probability that a target corresponding to component $f_{k|k-1}^i$ is detected by the lidar. The other l_k components are created using measurement update equations of the Kalman filter:

$$m_{k|k}^{i(l_k+1)+h} = m_{k|k-1}^i +^L K_k^i (l_k^h - m_{k|k-1}^i) \tag{13}$$

$$p_{k|k}^{i(l_k+1)+h} = (I -^L K_k^i {}^L H_k) p_{k|k-1}^i \tag{14}$$

$$w_{k|k}^{i(l_k+1)+h} = {}^L P_d^i w_{k|k-1}^i \mathcal{N}\{l_k^h; m_{k|k-1}^i, p_{k|k-1}^i\} \tag{15}$$

where ${}^L H_k$ is the lidar observation matrix and K_k is the associated Kalman gain:

$${}^L H_k = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad {}^L K_k^i = p_{k|k-1}^i {}^L H_k^T ({}^L H_k p_{k|k-1}^i {}^L H_k^T + {}^L R_k)^{-1} \tag{16}$$

where ${}^L R_k$ is the covariance of the noise on the lidar measurements, that is determined experimentally as:

$${}^L R_k = \begin{pmatrix} 0.0025 & 0 \\ 0 & 0.0025 \end{pmatrix} \tag{17}$$

The probability of detection ${}^L P_d^i$ is a key parameter for the success of the filter. For each component $f_{k|k-1}^i$, ${}^L P_d^i$ is calculated considering four factors that limit the lidar sensor ability to detect objects. Distance, blind spots caused by the camera holders, obstruction of a robot by another robot, and interference caused by other lidar sensors. The final ${}^{L}P_d^j$ is the product of all those factors:

$$ {}^{L}P_d^j ={}^{L} P_{d|dis}^i *{}^{L} P_{d|cs}^i *{}^{L} P_{d|b}^i *{}^{L} P_{d|in}^i \tag{18}$$

The first factor is the distance of each component $f_{k|k-1}^i$ which is related to the lidar sensor range. If a component is located beyond the range of the lidar, then it will not be detected. In our particular case, the range of the lidar is limited to $2m$. A sigmoid function was used to calculate ${}^{L}P_{d|dis}^i$:

$$ {}^{L}P_{d|dis}^i = \frac{1}{\left(1.02 + e^{8*(||m_{k+1|k}^i||-1.5)}\right)} \tag{19}$$

The second factor is due to the two pillars that support the fish eye cameras on $\mathcal{R}^j$, that create four blind spots in the field of view (FOV) of the lidar, whose centers $\alpha_i, i = 1, \ldots, 4$ and angular width $\beta_i, i = 1, \ldots, 4$ were determined experimentally. Denoting with $\theta_{k|k-1}^i$ the bearing angle of the mean of the i-th component, a sum of Gaussian functions is implemented to calculate ${}^{L}P_{d|cs}^i$ (Figure 4):

$$ {}^{L}P_{d|cs}^j = \sum_{i=1}^{4} \left(1 - \mathcal{N}\{\theta_{k|k-1}^i; \alpha_i, \beta_i/2\}\right). \tag{20}$$

The third factor ${}^{L}P_{d|b}^i$ models the situation in which robots block each other from the FOV of the lidar, that is therefore unable to collect a measurement for the robot that is behind. Similar situation seen in Figure 5. Hence the probability of detection of each component is reduced incorporating a zero mean Gaussian function ${}^{L}P_{d|b}^i$ based on i) the angle difference θ_{diff} between pairs of components; and ii) their Euclidean distance $||m_{k+1|k}^i||$ from R^j. When θ_{diff} becomes close to zero for some pair, the robot which has the shortest Euclidean distance from $\mathcal{R}^j$ will block the other robot. For the generic component $f_{k+1|k}^i(x)$, using all the components, ${}^{L}P_{d|b}^i$ is calculated as:

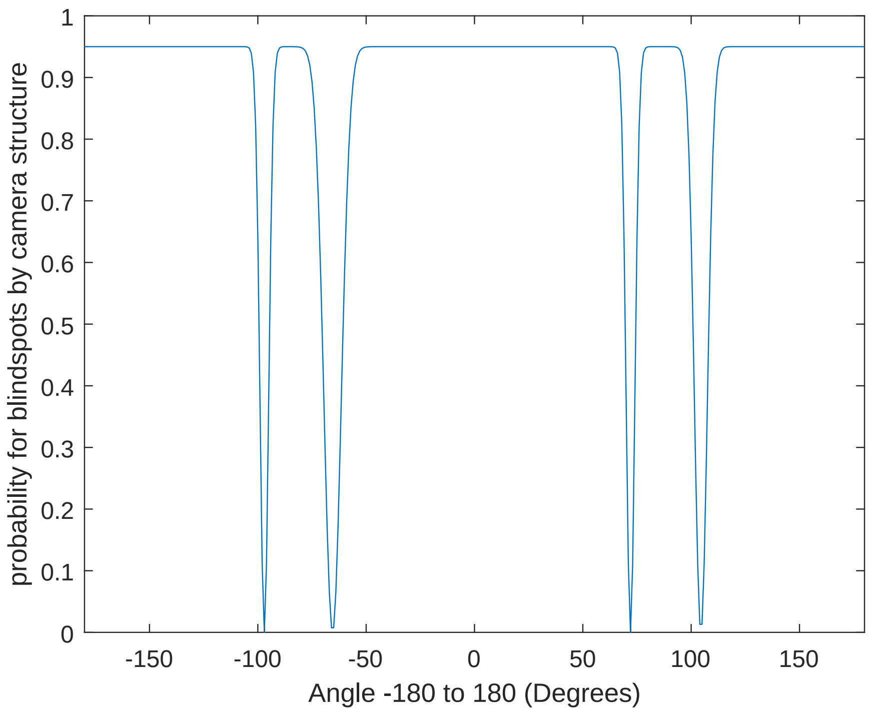


Figure 4: Probability of detection for blind spots created by camera structure

$$\theta_{diff} = |\theta^{j}_{k|k-1} - \theta^{i}_{k|k-1}| \tag{21}$$

$${}^{L}P^{i}_{d|b} = \sum_{\forall\{i,j\}:||m^{j}_{k|k-1}||<||m^{i}_{k|k-1}||} (1 - w^{j}_{k|k-1}\mathcal{N}(\theta_{diff}; 0, 3deg)) \tag{22}$$

The last factor ${}^{L}P^{i}_{d|in}$ models the interference caused by the lidar sensors mounted on the other robots, that we noticed during the testing phase. Whenever two lidar sensors are pointing at each other, their readings record null (invalid) measurements in correspondence to the other robots. Given the rotational nature of the internal mechanical structure of the lidars, this interference manifested itself as a loss of a measurement associated with the appearance of null measurements with a pseudo-periodic pattern. To model this interference, we considered that, along with the measurements, the lidar provides the intensity of the returning laser beam. When interference occurs, it will zero the lidar intensity $l^{\theta}_{int} \in L_k$,

$\vartheta = 0, 0.5, \ldots, 360$. Therefore we used the intensity readings to calculate ${}^{L}P^{i}_{d|in}$ for $f^{i}_{k+1|k}(x)$:

$$ {}^{L}P^{i}_{d|in} = \sum_{\forall \{l^{\vartheta}_{int}=0, \vartheta=0,0.5,\ldots 360\}} \frac{1 - 0.6 * e^{-(\theta^{i}_{k|k-1} - \angle l^{\vartheta}_{int})^2}}{2 * c^2} \tag{23} $$

where c denotes the covariance of each $l^{\vartheta}_{int} = 0$.

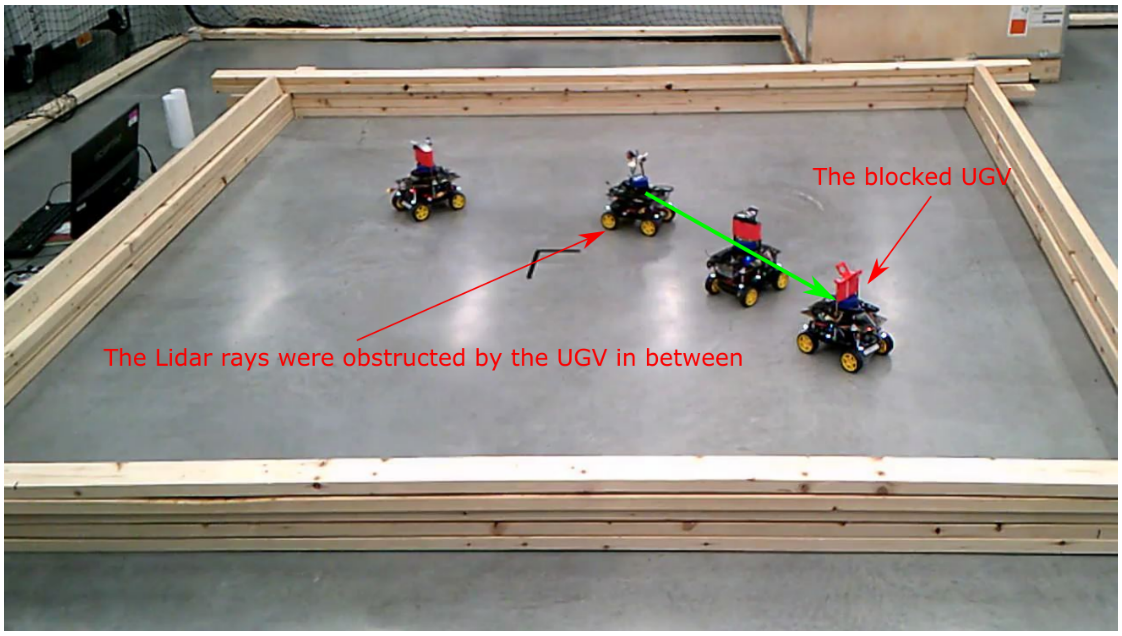


Figure 5: Robots block each other

3.3 Camera Measurement Update

Similar to the lidar measurement update, the camera update is performed only when new camera measurements $C_k = \{c_k^1, ...c_k^{c_k}\}$ are available. Each c_k^1 is provided as a 2D normalized vector pointing in the direction of a target. Following equation (6), each component $f^i_{k|k-1}$ generates $c_k + 1$ components in $f_{k|k}$. As for the lidar measurements, one component $f^{i(c_k+1)}_{k|k}$ has the same mean $m^{i(c_k+1)}_{k|k} = m^i_{k|k-1}$ and covariance $p^{i(l_k+1)}_{k|k} = p^i_{k|k-1}$ of the original component, while the weight is updated as:

$$w_k^{i(c_k+1)} = (1 - {}^C P_d^i) w^i_{k|k-1} \tag{24}$$

The other c_k components are computed using the Kalman filter equations:

$$m^{i(c_k+1)+h}_{k|k} = m^i_{k|k-1} + {}^C K_k^i (c_k^h - m^i_{k|k-1}) \tag{25}$$

$$p^{i(c_k+1)+h}_{k|k} = (I - {}^C K_k^i {}^C H_k) p^i_{k|k-1} \tag{26}$$

$$w^{i(c_k+1)+h}_{k|k} = {}^C P_d^i w^i_{k|k-1} \mathcal{N}\{c_k^h; m^i_{k|k-1}, p^i_{k|k-1}\} \tag{27}$$

where ${}^C P_d^i$ is the camera probability of detection, H_k is the observation matrix, and K_k is Kalman gain:

$$H_k = \left(\frac{-y^i}{||m^i_{k|k-1}||}, \frac{x^i}{||m^i_{k|k-1}||} \right) \qquad K_k^i = p^i_{k|k-1} H_k^T (H_k p^i_{k|k-1} H_k^T + {}^C R_k)^{-1} \tag{28}$$

where $m^i_{k|k-1} = [x^i \; y^i]^T$ and ${}^C R_k = 25deg$ is the covariance of the noise of the camera measurements.

The probability of detection ${}^C P_d^i$ is computed as the product of two factors, distance from $\mathcal{R}^j$ and obstruction of a robot by another robot:

$${}^C P_d^i = {}^C P^i_{d|dis} * {}^C P^i_{d|b} \tag{29}$$

The first factor is computed with an inverse sigmoid function:

$${}^C P^i_{d|dis} = \frac{1}{(1.1 + e^{5*(||m^j_{k+1|k}||-4)})} \tag{30}$$

while for the second factor, ${}^C P^i_{d|b} = {}^L P^i_{d|b}$.

CHAPTER 4

Simulation

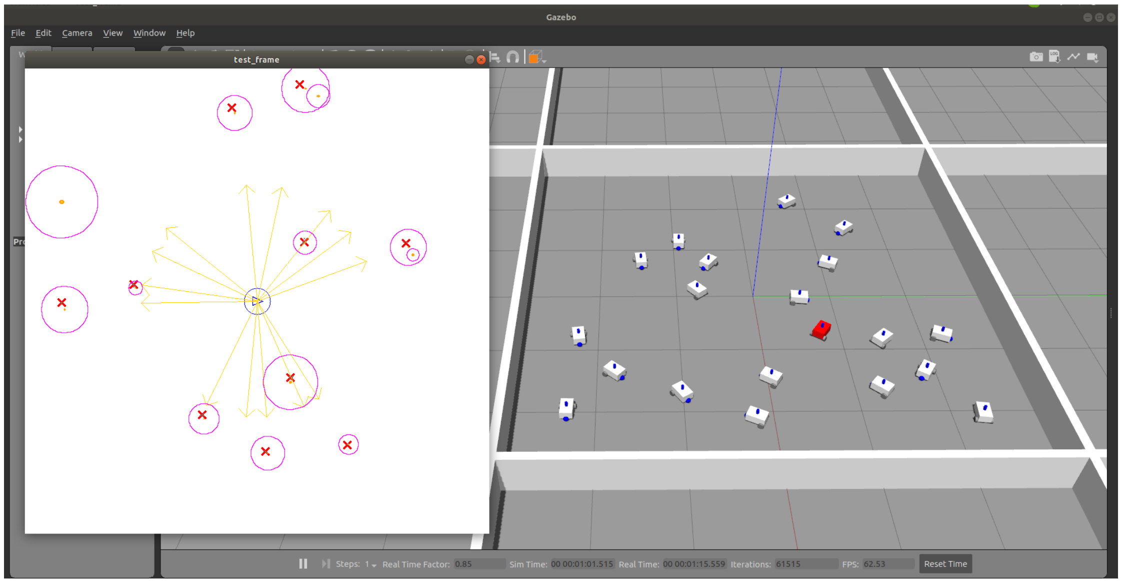

Gazebo
File Edit Camera View Window Help
test_frame
Steps: 1 Real Time Factor: 0.85 Sim Time: 00 00:01:01.515 Real Time: 00 00:01:15.559 Iterations: 61515 FPS: 62.53 Reset Time

Figure 6: Gazebo 20 UGVs simulation

The filter was first tested in simulation before testing in real robot experiment. A swarm of UGVs was simulated in Gazebo/ROS. The UGV model was equipped with a simulated lidar sensor. The implementation of a fish-eye camera was difficult due to unavailability of its manufacture parameters and the computational load of simultaneously simulating 10 -20 camera sensors, hence the camera measurements were generated directly using the UGV locations. A zero mean Gaussian noise was added to the camera measurements to simulate realistic measurements.

Figure 6 shows the typical simulation setup with a 6m x 6m square testing area with walls to keep the UGVs in close proximity to each other. The experiment consists of 20 UGVs where a single UGV executes the filter to estimate the position of other 19 UGVs. All UGVs perform a pseudo random motion with obstacle avoiding capabilities. In Figure 6 the red colored UGV performs the filter and its knowledge is visualized in the visualizer window on the left side. The visualizer was developed in openCV to interpret the filter output and simplify the debugging process. Additionally, it visualizes the measurements provided by the sensors. The Visualizer provides 360 degree view around the UGV with the 2m range in each side. 2 meters was considered as the lidar sensor range and 3m for the camera sensor. In the Visualizer, the symbol 'x' shows the lidar measurement, arrows represents the bearing measurements from the camera and the circles show the weight of each component of the estimated PHD. The experiment was conducted for 300 seconds and the filter results were recorded to a ROS bag. The ROS bag was later analysed using Matlab software to generate the plots.

4.1 Simulation results

Three plots were generated using simulated results to evaluate the performance of the filter and illustrate the benefits of the proposed multi-sensor approach. The multi-sensor approach was compared by running the filter with (Lidar+Camera - LC) and without camera measurements (Lidar only - LO).

The bar chart in Figure 7 represents the percentage of time for which each UGV error was greater than 30 cm only considering the time when the UGV was within the sensor range. The filter estimates was compared with the actual UGV locations provided by the simulator to compute the error. Overall out of 20 UGVs only four UGVs had the error greater than 25 percent in LC method and 9 robots in LO. It is well seen in the Bar chart that LC method performs well compared to LO method.

Figure 8 represents the total sum of the weight of all components with respect to the time during the whole experiment. The filter estimated in average 15 UGVs in the duration of the simulation using LC method.

In Figure 9 the plot shows the path for each UGV during the simulation when

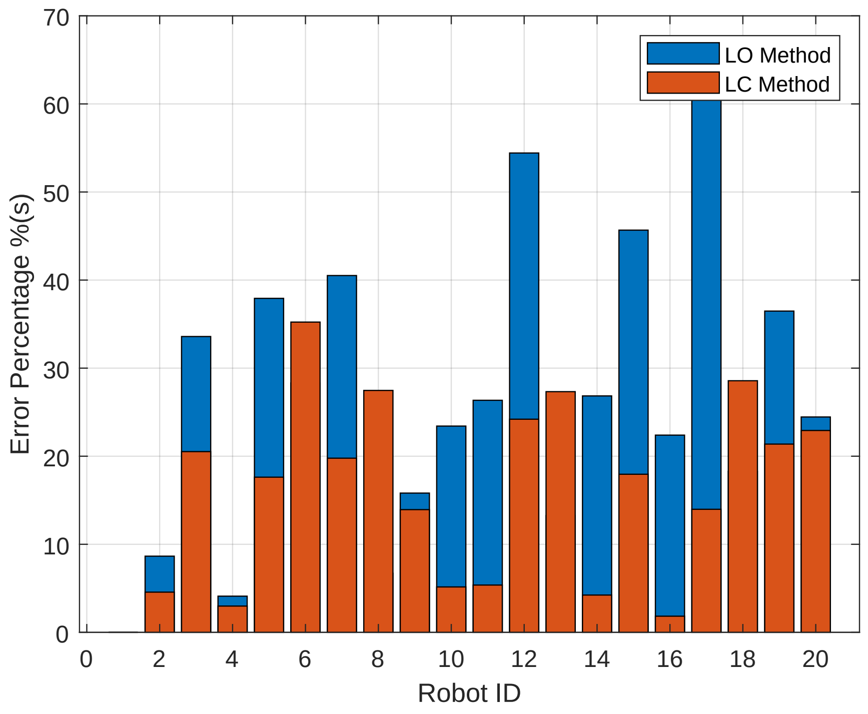


Figure 7: Percentage of time for which each UGV error was greater than 30 cm

LC method used. The symbol 'x' represents the mean of the components computed by the filter. At each time step, the UGV path and the relevant estimates were plotted. Overall the filter was able to follow an accurate path of the UGVs. Note that the UGV paths that are not near estimates refers to UGV that were out of the sensor range during that time step.

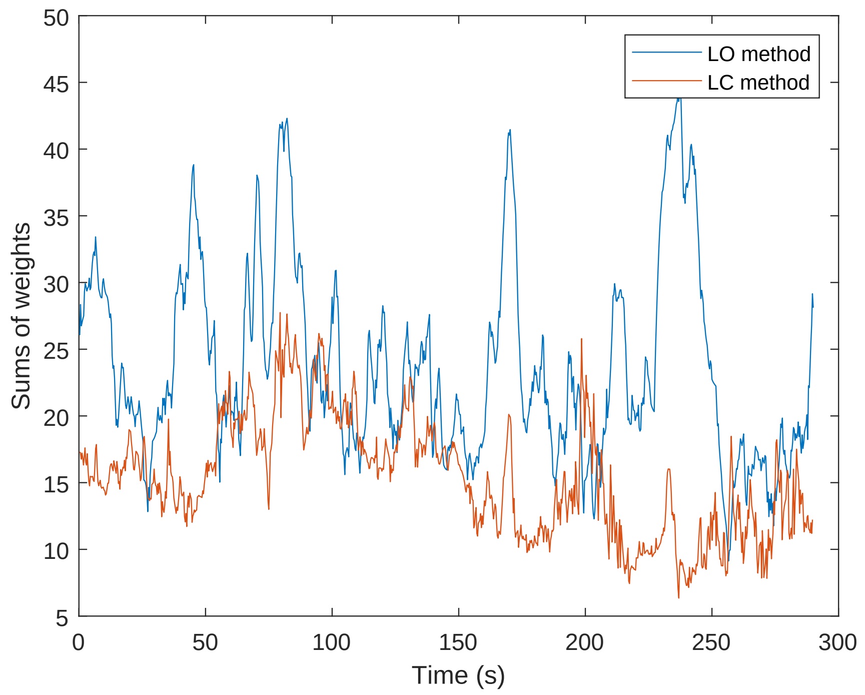


Figure 8: No of components estimated by the filter

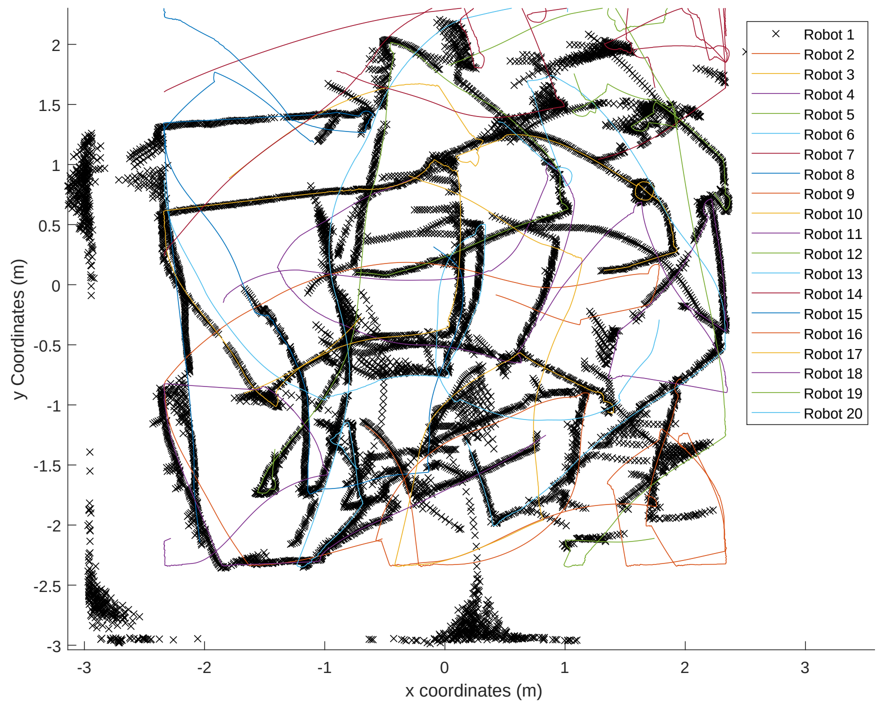


Figure 9: UGV paths during the simulation - LC method

CHAPTER 5

Experimental Setup

The relative localization system has been tested in robot experiments with four Unmanned Ground Vehicles (UGV). During a typical experiment, the UGVs will run a simple pseudo-random motion with obstacle avoidance. All computations are done on the on-board Odroid and using on-board sensors. Similar to the simulation, one UGV performs the estimation algorithm and the final estimates were published on a ROS topic and recorded to a ROS bag along with ground truth provided by an Optitrack motion capture system.

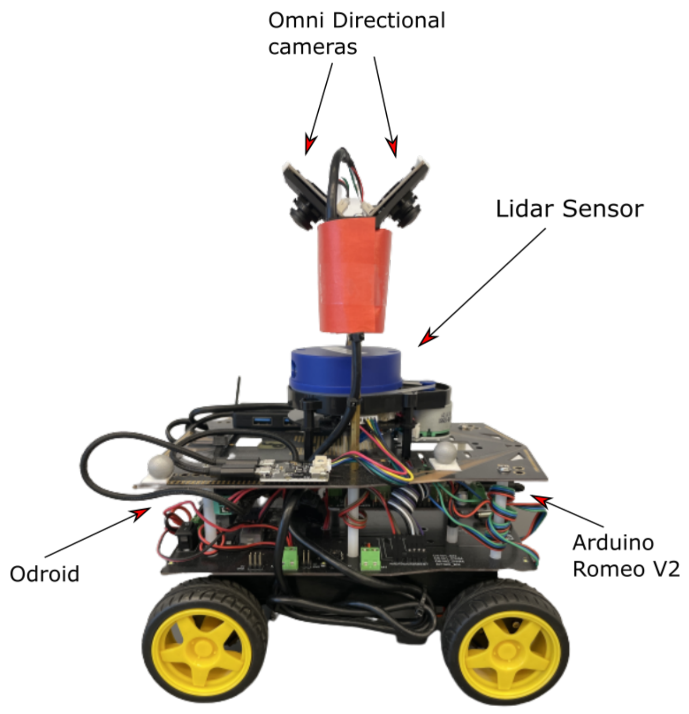


Figure 10: Final Robot Design

Figure 10 shows the final design of the UGV that is used for the experiment. The design consist of a four-wheeled base equipped with two Omni-directional cameras, one Lidar sensor, a main processing unit, a lower level processing unit, wheel encoders and a 12V battery pack to provide power. Additionally each UGV was retrofitted with a red color strip on the camera structure to improve the detection in camera measurements.

5.1 UGV Platform

The UGVs were constructed using a commercially available four wheeled differential drive robot platform, the DFRobot Cherokey (22.5cm x 17.5cm). Each UGV is equipped with wheel encoders and a Romeo V2 (an Arduino Robot Board (Arduino Leonardo) with Motor Driver). The Romeo V2 processes and executes the low level control to follow desired velocity commands.

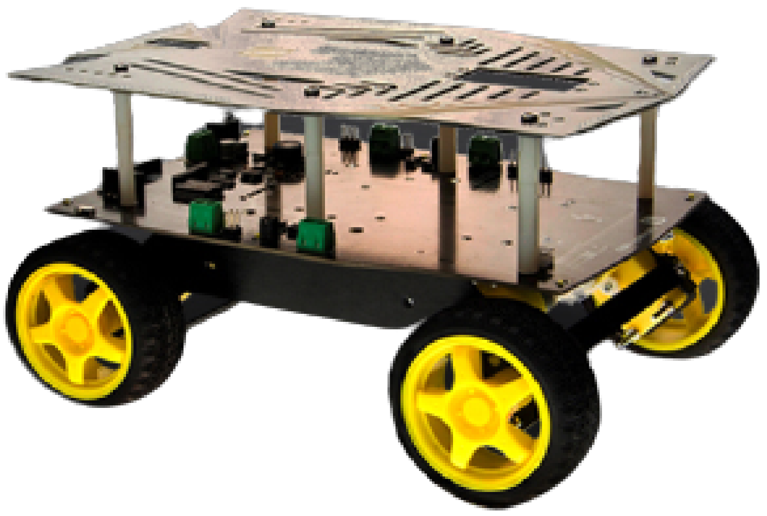


Figure 11: Robot Platform - DFRobot Cherokey [2]

The DFRobot Cherokey has two levels in its platform. The lower level platform is embedded with motor controllers and the power distribution for the motors. The space between two levels is utilized to accommodate the Battery pack, main processing unit, voltage divider circuits and Arduino robot board. The top level is utilized to attach the three sensors.

5.2 Main Processing unit

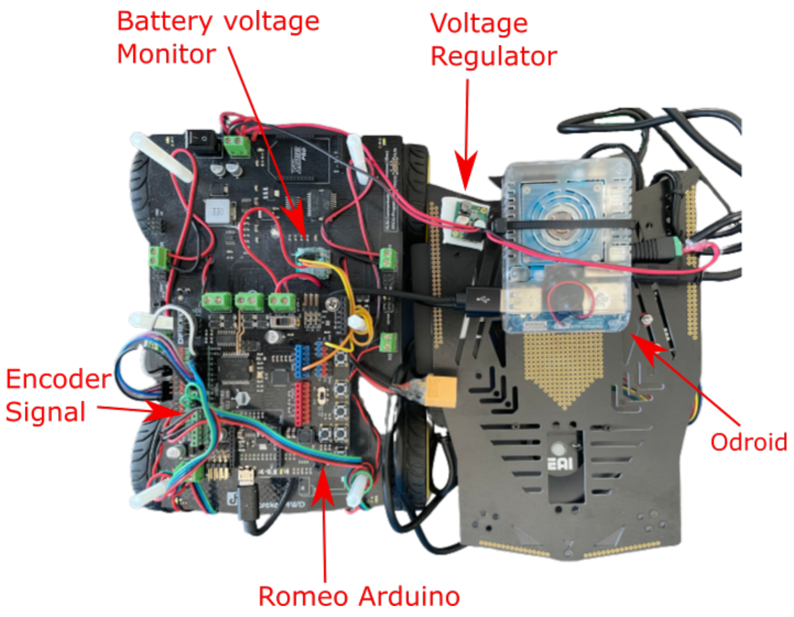


Figure 12: Inside UGV

For processing higher level tasks, an Odroid-XU4 - a small single board computer - is mounted on the robot. The Odroid-XU4 hosts an Exynos5422 Cortex™-A15 2Ghz Quad core and a Cortex™-A7 1.5Ghz Quad core CPUs with Mali-T628 MP6 GPU [34]. The Odroid runs a GNU-Linux OS along with Robot Operating System (ROS) to manage sensor data collection and real time processing.

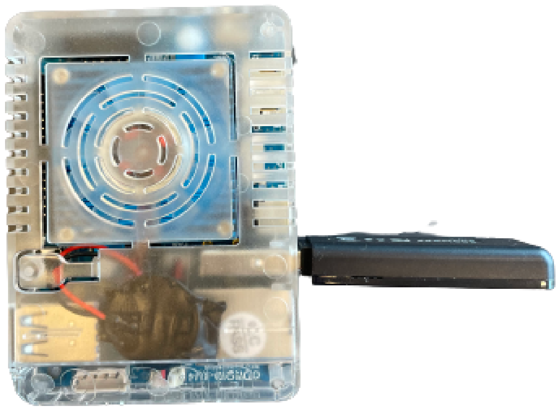


Figure 13: Odroid-XU4

5.3 Odometry system

The odometry system was implemented to compute the displacement of the robot. The displacement during a given time period is required to compute the time update of the filter. The implementation of the odometry was done by counting the impulses provided by the motor encoders for a given time period. The encoders are connected to Romeo Arduino board's interrupt pins to count the received impulses.

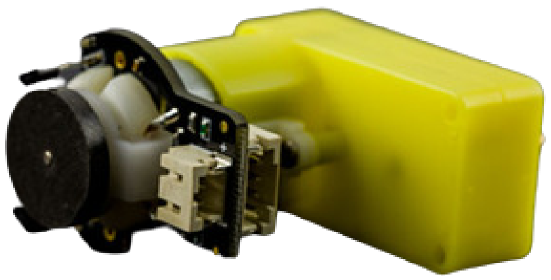


Figure 14: Motor with Encoder

For a single motor revolution, the encoder provides 16 pulses. The 120:1 gear box attached to the motor increases the encoder pulses to 1920 per wheel revolution. By counting the pulses P_t at time t, the linear displacement D_t and Angular displacement θ_t is given by:

$$D_t = \frac{\pi * 2 * r_w * 0.5 * (P_t^L + P_t^R)}{1920} \tag{31}$$

$$\theta_t = \frac{\pi * 2 * r_w * (P_t^L - P_t^R)}{1920 * w_{ax}} \tag{32}$$

where P_t^L, P_t^R are pulses of the left side motor and pulsars of the right side motor, r_w is the wheel radius, and w_{ax} is the UGV axis width.

5.4 UGV Sensors

Each UGV is equipped with a Lidar and two omni directional cameras that are integrated to have 360 degree field of view. All three sensors are connected to the Odroid using standard USB ports. Both the lidar and the cameras are fixed in the origin O_j of $\mathcal{F}_j$ and aligned with the X axis, eliminating rotational and translational complexities during the image and scan processing.

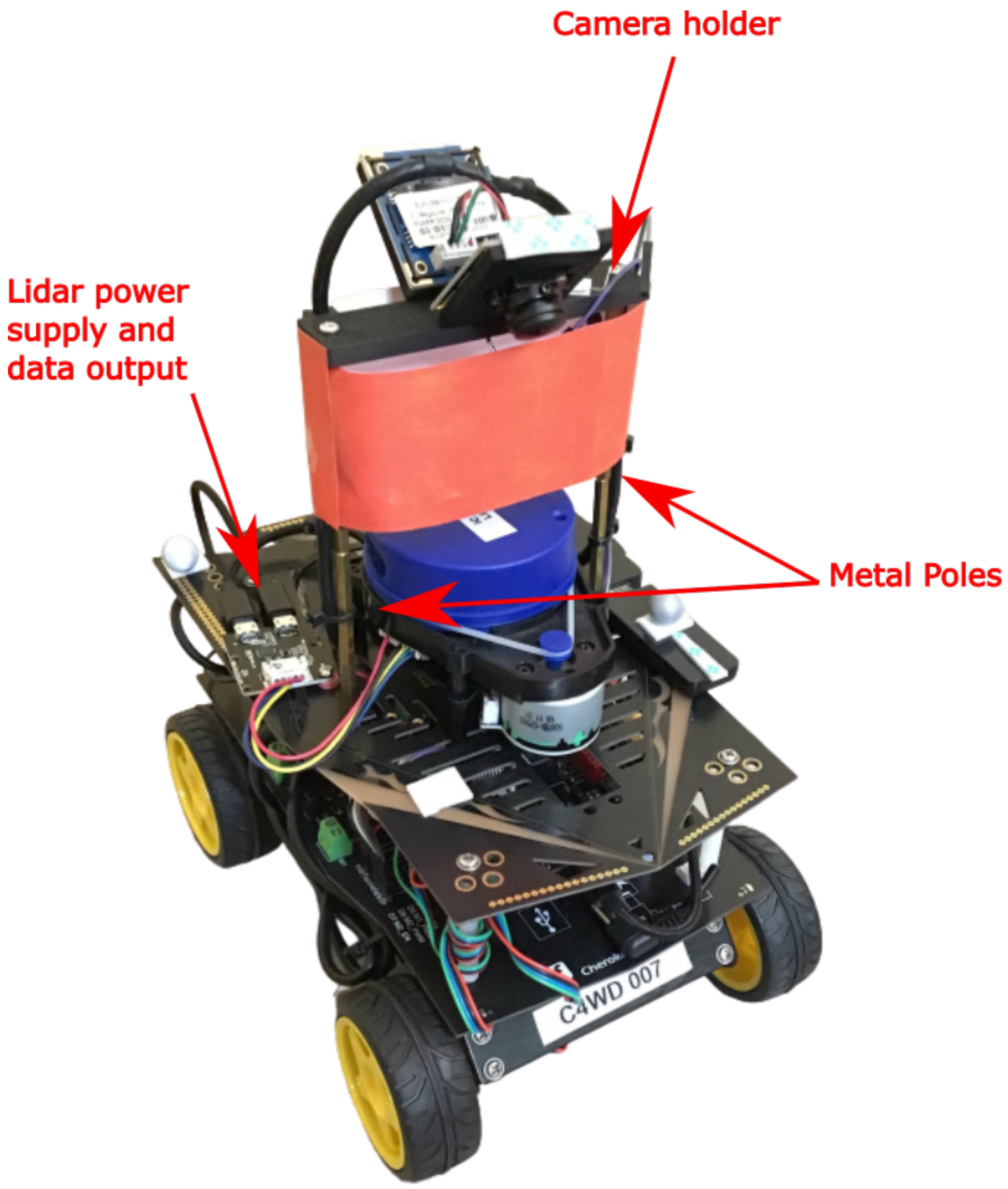


Figure 15: UGV Front View

5.4.1 Lidar sensor

The Lidar model XU4 from YDLidar was selected as the lidar sensor for the UGV. The XU4 is a low-cost, light-weight, belt driven, two dimensional range finder. It can provide range information up to 10m in all 360 degrees at 7 Hz frame rate. For a single revolution it provides 720 range data points with the resolution of data point per 0.5 degree. The lidar is mounted on the center of the top platform to align with the UGV's frame of reference system. A ROS software package is provided with the lidar that publishes the scan data. A separate algorithm was developed to read scan data from the ROS topic and search for other UGVs.

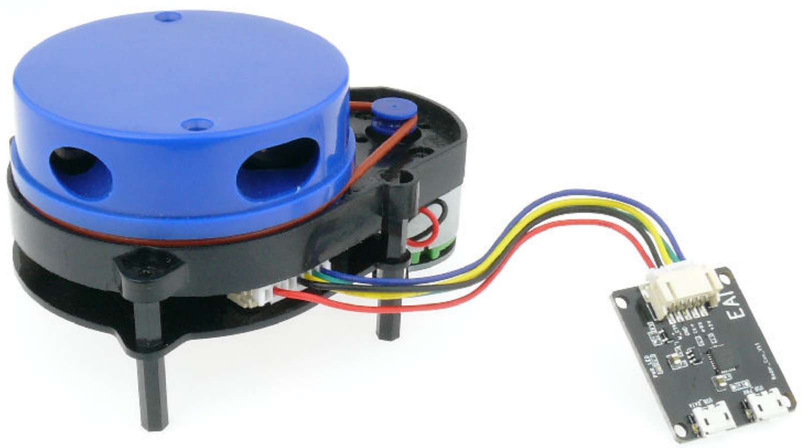


Figure 16: YD-Lidar X4 [3]

5.4.2 Omni-directional Cameras

The two cameras are standard USB web cameras equipped with a 180 degree fish-eye lens. Each camera provides two megapixel 1920x1080 resolution images at 30 fps rate.

Figure 17: Camera Sensor

A camera holder was designed and 3D printed to hold the cameras above the lidar at an height of 130mm from UGV and a 45 downward degree angle. This specific height and angle is designed to maximize the horizontal FOV of the combined camera images.

All the UGVs have been equipped with a red strip around the camera holder in order to allow camera tracking via color extraction.

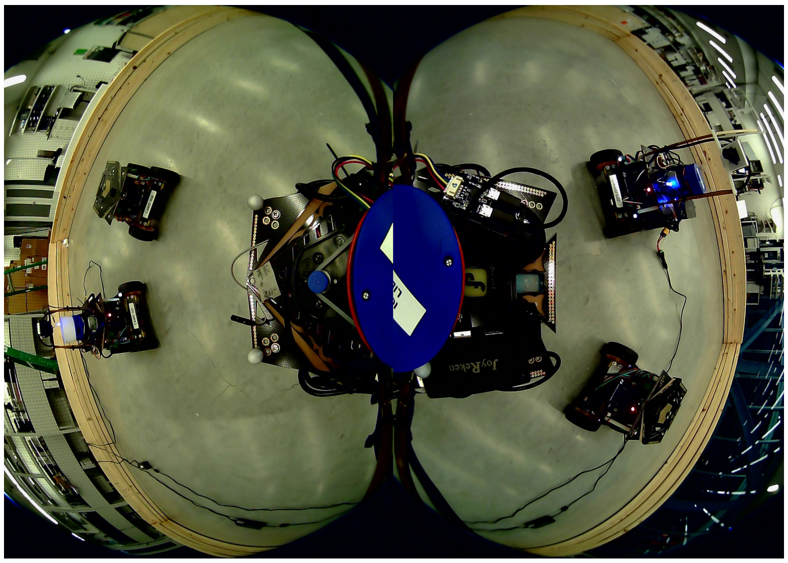


Figure 18: Front and rear camera images

5.5 The testing area

The testing area is a 3m x 3m square space with raised walls to avoid disturbances from the external environment and keep the robots in proximity of each other. However, it is also larger than the FOV of the lidar to allow robots to exit and re-enter the tracking radius. Ground truth of the actual position of the robots is provided at each time by an Optitrack 6Dof motion tracking system.

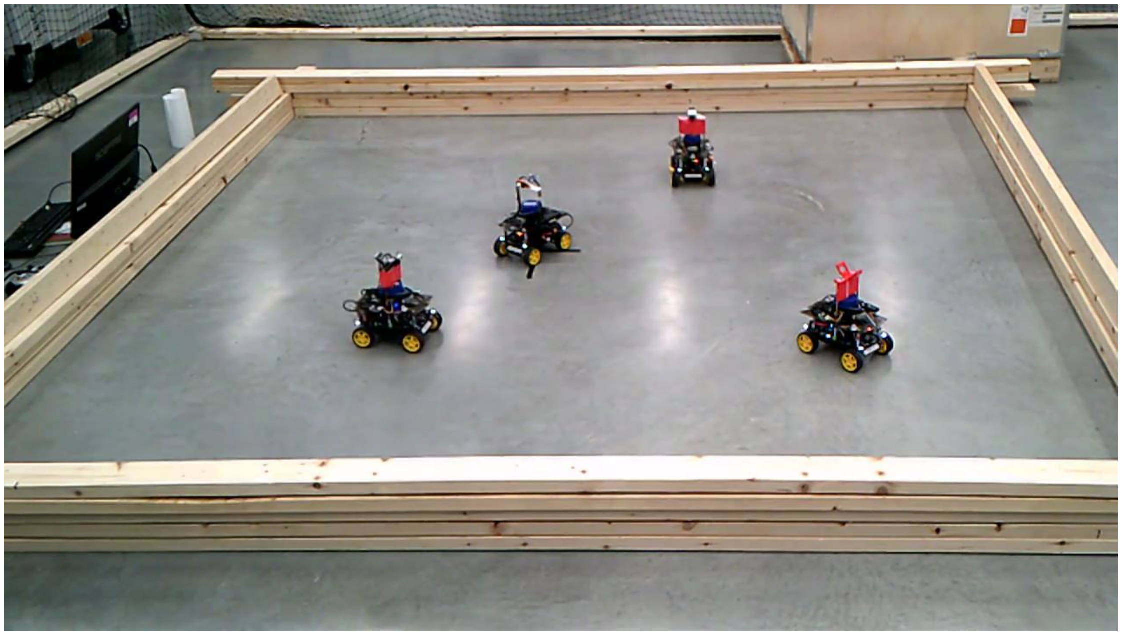


Figure 19: Testing area

5.6 Lidar Measurements

An algorithm was developed to obtain each lidar scan published in the ROS topic and search for other UGV. During a typical lidar scan, the laser beam is reflected when it hits on other UGV's lidar structure which has a shape of a circle. As seen in Figure 20 each UGV was represented by small set of points, which has a shape of an arch when inspected closely. Arc lengths of these points were computed and compared with the diameter of lidar structure to distinguish UGVs from larger obstacles. The measured diameter of typical lidar structure is 65mm.

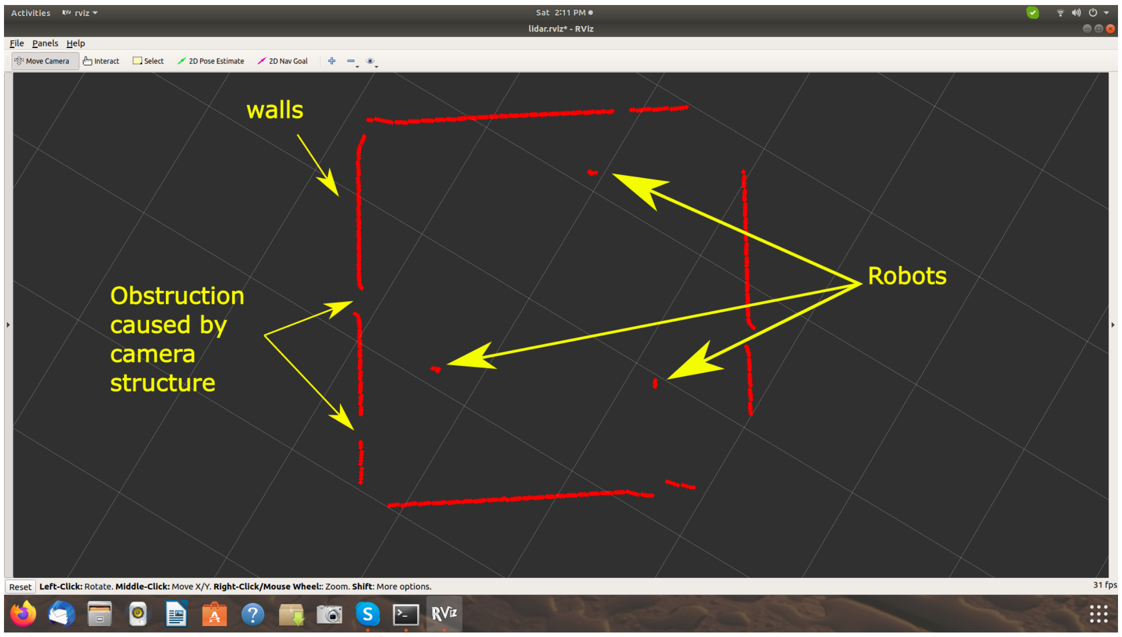
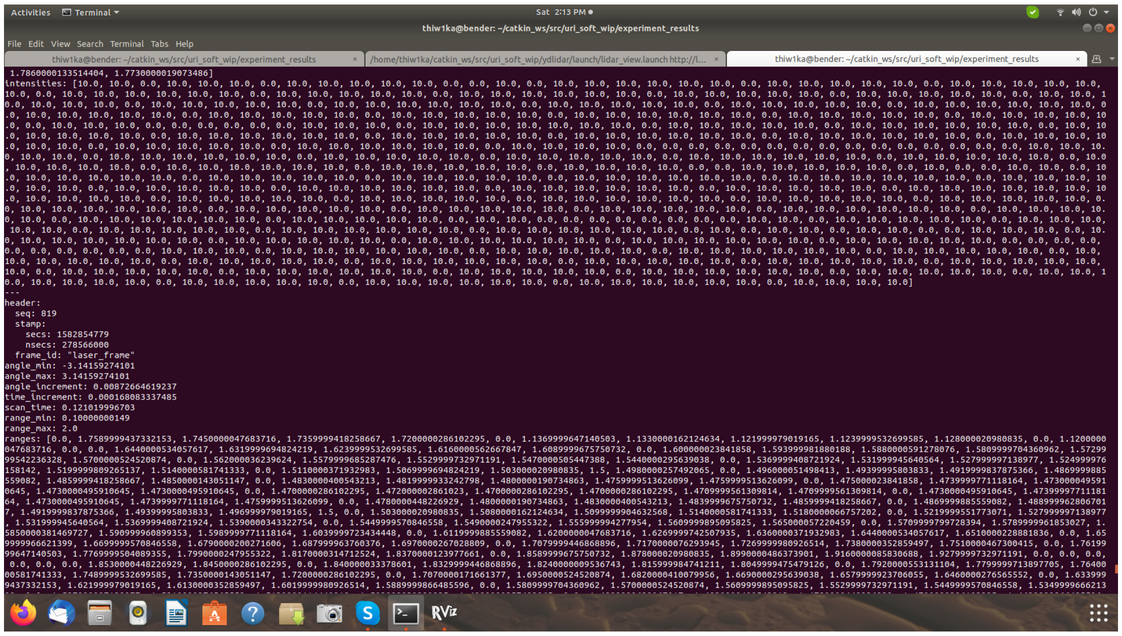

walls
Robots
Obstruction caused by camera structure
lidar.rviz* - RViz
File Panels Help
Move Camera
Interact
Select
2D Pose Estimate
2D Nav Goal
Reset
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.
31 fps

Figure 20: Points provided by a typical lidar scan during the experiment. visualized using Rviz

The arc lengths which were less than 65mm were extracted as the possible measurements for other UGVs. Range and bearing for these possible measurements were translated into Cartesian coordinates, rotated to UGV frame of reference and provided as the lidar measurements to the filter.



```
 1.7860000133514404, 1.7730000019073486]
intensities: [10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 0.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0,
10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 1
0.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0
.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10
.0, 0.0, 10.0, 10.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10
.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10
.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 10.0, 10.0, 10.
0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0
, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 0.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 0.0, 0.0, 10.0, 10.0, 0.0, 10
.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10
.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10
.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.
0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.
0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0,
 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 0.0, 10.0, 10.0, 0.0, 10.0, 10.0, 0.0, 0.0, 10.0, 10.0, 10.0, 0.0, 10.
0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0,
10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0,
10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 1
0.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0, 10.0, 10.0, 0.0, 10.0, 10.0, 10.0, 10.0]
---
header:
  seq: 819
  stamp:
    secs: 1582854779
    nsecs: 278566000
  frame_id: "laser_frame"
angle_min: -3.14159274101
angle_max: 3.14159274101
angle_increment: 0.00872664619237
time_increment: 0.000168083337485
scan_time: 0.121019996703
range_min: 0.10000000149
range_max: 2.0
ranges: [0.0, 1.7589999437332153, 1.7450000047683716, 1.7359999418258667, 1.7200000286102295, 0.0, 1.1369999647140503, 1.1330000162124634, 1.121999979019165, 1.1239999532699585, 1.128000020980835, 0.0, 1.1200000
047683716, 0.0, 0.0, 1.6440000534057617, 1.6319999694824219, 1.6239999532699585, 1.6160000562667847, 1.6089999675750732, 0.0, 1.600000023841858, 1.593999981880188, 1.5880000591278076, 1.5809999704360962, 1.57299
99542236328, 1.5700000524520874, 0.0, 1.562000036239624, 1.5579999685287476, 1.5529999732971191, 1.5470000505447388, 1.5440000295639038, 0.0, 1.5369999408721924, 1.531999945640564, 1.527999997138977, 1.524999976
158142, 1.5199999809265137, 1.5140000581741333, 0.0, 1.5110000371932983, 1.5069999694824219, 1.503000020980835, 1.5, 1.4980000257492065, 0.0, 1.496000051498413, 1.49399995803833, 1.4919999837875366, 1.4869999885
559082, 1.4859999418258667, 1.4850000143051147, 0.0, 1.4830000400543213, 1.4819999933242798, 1.4800000190734863, 1.4759999513626099, 1.4759999513626099, 0.0, 1.475000023841858, 1.4739999771118164, 1.473000049591
0645, 1.4730000495910645, 1.4730000495910645, 0.0, 1.4700000286102295, 1.472000002861023, 1.4700000286102295, 1.4700000286102295, 1.4709999561309814, 1.4709999561309814, 0.0, 1.4730000495910645, 1.47399997711181
64, 1.4730000495910645, 1.4739999771118164, 1.4759999513626099, 0.0, 1.4780000448226929, 1.4800000190734863, 1.4830000400543213, 1.4839999675750732, 1.4859999418258667, 0.0, 1.4869999885559082, 1.488999962806701
7, 1.4919999837875366, 1.49399995803833, 1.496999979019165, 1.5, 0.0, 1.503000020980835, 1.5080000162124634, 1.5099999904632568, 1.5140000581741333, 1.5180000066757202, 0.0, 1.5219999551773071, 1.527999997138977
, 1.531999945640564, 1.5369999408721924, 1.5390000343322754, 0.0, 1.5449999570846558, 1.5490000247955322, 1.555999994277954, 1.5609999895095825, 1.565000057220459, 0.0, 1.5709999799728394, 1.5789999961853027, 1.
5850000381469727, 1.590999960899353, 1.5989999771118164, 1.6039999723434448, 0.0, 1.6119999885559082, 1.6200000047683716, 1.6269999742507935, 1.6360000371932983, 1.6440000534057617, 1.6510000228881836, 0.0, 1.65
9999966621399, 1.6699999570846558, 1.6790000200271606, 1.687999963760376, 1.6970000267028809, 0.0, 1.7079999446868896, 1.7170000076293945, 1.7269999980926514, 1.7380000352859497, 1.7510000467300415, 0.0, 1.76199
99647140503, 1.7769999504089355, 1.7990000247955322, 1.8170000314712524, 1.8370000123977661, 0.0, 1.8589999675750732, 1.878000020980835, 1.8990000486373901, 1.9160000085830688, 1.9279999732971191, 0.0, 0.0, 0.0,
 0.0, 0.0, 0.0, 1.8530000448226929, 1.8450000286102295, 0.0, 1.840000033378601, 1.8329999446868896, 1.8240000009536743, 1.815999984741211, 1.8049999475479126, 0.0, 1.7920000553131104, 1.7799999713897705, 1.76400
00581741333, 1.7489999532699585, 1.7350000143051147, 1.7200000286102295, 0.0, 1.7070000171661377, 1.6950000524520874, 1.6820000410079956, 1.6690000295639038, 1.6579999923706055, 1.6460000276565552, 0.0, 1.633999
9437332153, 1.621999979019165, 1.6130000352859497, 1.6019999980926514, 1.5889999866485596, 0.0, 1.5809999704360962, 1.5700000524520874, 1.5609999895095825, 1.5529999732971191, 1.5449999570846558, 1.5349999666213
```

Figure 21: ROS message - Single Lidar scan

5.7 Camera Measurements

The camera measurement are generated using a color extraction method. The color extraction algorithm first defines the boundaries for RGB (Red, Green, Blue) color space to filter the red color. When an image is received from an UGV camera, it is sent through the color filter to eliminate other colors except red. The remaining red color blobs were extracted and their pixel coordinates are used to compute the bearing measurements.

Spherical coordinate system was used to compute the angular measurements from the pixel coordinates. Given pixel coordinate x_p, y_p for an identified blob, pixel distance D_s between sensor and fish eye lens, the angular measurement in camera frame of reference ${}^{c}\theta^{1}$:

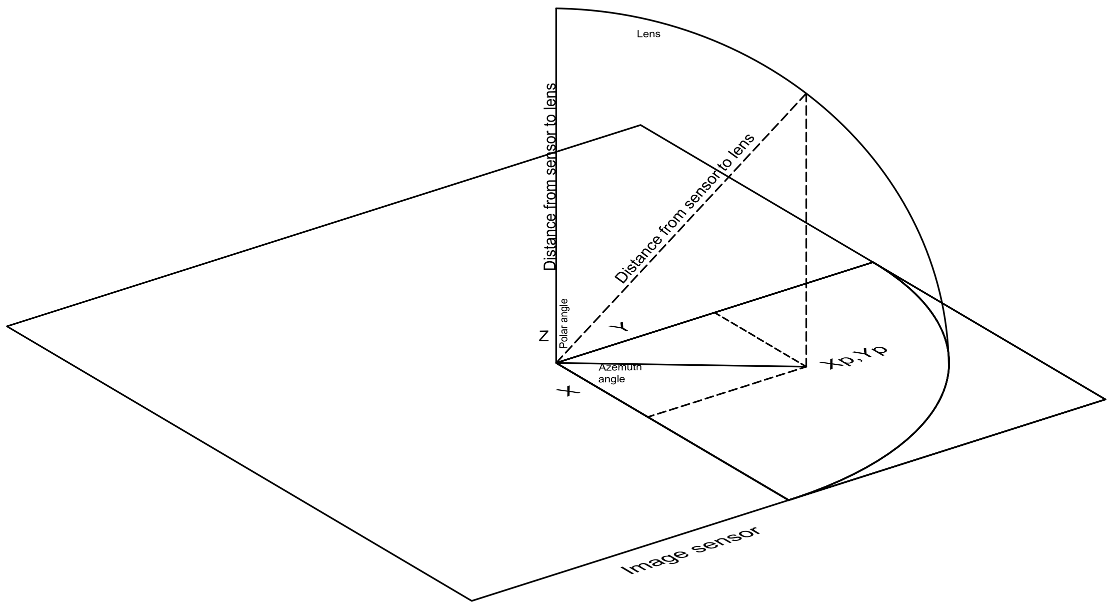


Figure 22: Image sensor

$$\theta_{azimuth} = \sqrt{{x_p}^2 + {y_p}^2} \tag{33}$$

$$\theta_{polar} = \arcsin(\frac{\sqrt{{x_p}^2 + {y_p}^2}}{D_s}) \tag{34}$$

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = \begin{pmatrix} \sin\theta_{polar} * \cos\theta_{azimuth} \\ \sin\theta_{polar} * \sin\theta_{azimuth} \\ \cos\theta_{polar} \end{pmatrix} \tag{35}$$

$$Z^1_{R_j} = R^{R_j}_c * Z_c \tag{36}$$

$$^c\theta^1 = \arctan\frac{y_{R_j}}{x_{R_j}} \tag{37}$$

Where $R^{R_j}_c$ is the rotational matrix from camera frame of reference to UGV frame of reference and y_{R_j},x_{R_j} are the coordinates in UGV frame of reference. Figure 22 shows the geometric argument for the sensor and the lens.

CHAPTER 6

Experimental Results

The experimental data was analysed in Matlab software to highlight the benefit of the proposed multi-sensor approach with (Lidar+Camera - LC) and without (Lidar only - LO) providing the camera measurements.

Figure 23 shows the distance error for each robot from the closest component whose weight is greater than 0.1. Overall, the LC method performs well except for some instants near time 150s and 190s in which the robot performing the estimate was consistently in a corner of the arena, hence with a limited field of view, effectively leading to robot's UGV 07 position not being measured for several tens of seconds. The plots show also that the LC method outperforms the LO method being able to keep the error bounded for most of the time when measurements are available.

A numerical comparison between the LC and LO methods is provided in Figure 24. In order to quantify the better performance of the LC filter, we have computed the percentage of time for which each robot's distance error is greater than $30cm$. The values, reported in the table in Figure 24(left), show how the employment of camera measurements in addition to the lidar greatly reduces the error time of a factor 2 to 5. Finally, in Figure 24(right) we report the total sum of the weight of all components with respect to time during the whole experiment. From this plot, it is possible to establish that the LC method is more effective in correctly estimating the number of robots, and therefore in eliminating estimates that refer to objects in the environment and not robots.

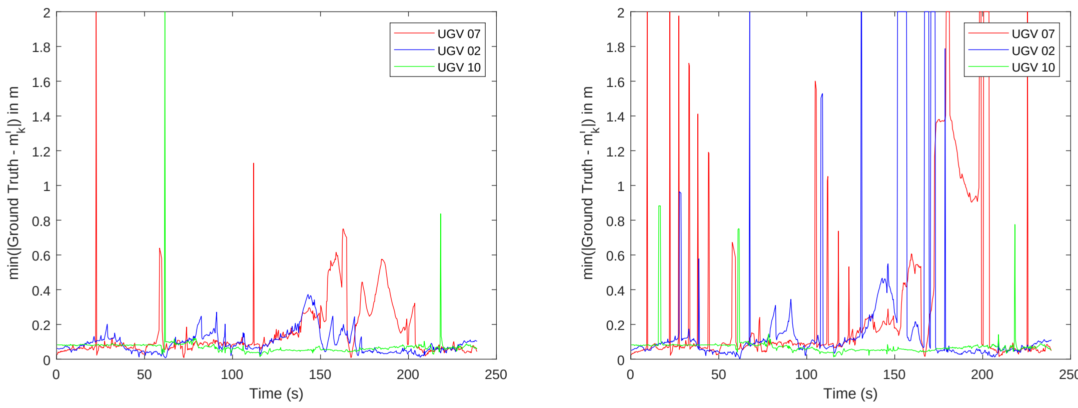
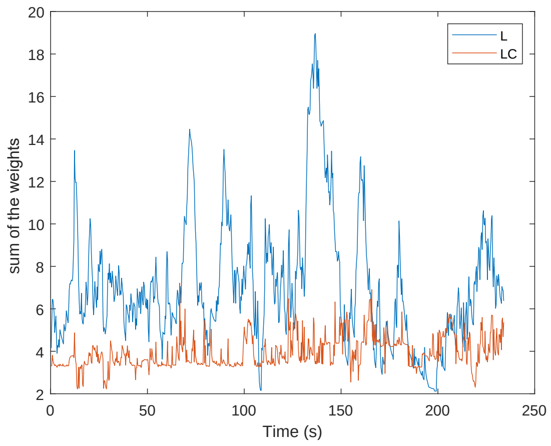


Figure 23: Distance error of the three UGVs in the LC (left) and LO (right) experiments.



Method	LO	LC
UGV 07	20.7%	12.7%
UGV 02	10.9%	2.6%
UGV 10	1%	0.2%

Figure 24: Comparison between the LO and LC experiments. Left: sum of the weights of all the components with LO (blue) and LC (red). Right: percentage of time that the error on the position of each robot is greater than 30cm with LO and LC.

CHAPTER 7

Conclusion

In this thesis we have presented a multi-sensor relative localization system for robotic swarms based on the PHD filter. Our system has been tested with real robot experiments, and evaluated against a single-sensor method based on the same principle. The results show that the multi-sensor approach performs better than the single-sensor method.

In the future, on the one hand we plan on improving the relative localization and include negative information measurements to simultaneously track robots and obstacles. On the other hand, we plan to pair the localization system with a decentralized formation control methods to perform real-world tasks as exploration, SLAM, patrolling and human-swarm interaction.

LIST OF REFERENCES

[1] R. T. Perera, C. Yuan, and P. Stegagno, “A phd filter based localization system for robotic swarms,” in *Proceedings DARS-SWARM2021 conference*, June 2021.

[2] “Cherokey: 4WD Mobile Robot for Arduino - DFRobot,” Mar 2021, [Online; accessed 20. Mar. 2021]. [Online]. Available: https://www.dfrobot.com/product-896.html

[3] “Review: Low cost YDLidar X4 sees 360º all around itself,” Mar 2021, [Online; accessed 20. Mar. 2021]. [Online]. Available: https://www.elektormagazine.com/news/review-low-cost-ydlidar-x4-sees-360-all-around-itself

[4] Zhong-yangZheng and Yang Tan, “Research advance in swarm robotics,” *Defence Technology*, vol. 9, pp. 18–39, 2013.

[5] M. Senanayake, I. Senthooran, J. C. Barca, H. Chung, J. Kamruzzaman, and M. Murshed, “Search and tracking algorithms for swarms of robots: A survey,” *Robotics and Autonomous Systems*, vol. 75, pp. 422 – 434, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889015001876

[6] M. Bakhshipour, M. J. Ghadi, and F. Namdari, “Swarm robotics search & rescue: A novel artificial intelligence-inspired optimization approach,” *Applied Soft Computing*, vol. 57, pp. 708 – 726, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494617301072

[7] K. N. McGuire, C. De Wagter, K. Tuyls, H. J. Kappen, and G. C. H. E. de Croon, “Minimal navigation solution for a swarm of tiny flying robots to explore an unknown environment,” *Science Robotics*, vol. 4, no. 35, 2019. [Online]. Available: https://robotics.sciencemag.org/content/4/35/eaaw9710

[8] E. Zahugi, M. Shanta, and T. Prasad, “Oil spill cleaning up using swarm of robots,” *Advances in Intelligent Systems and Computing*, vol. 178, pp. 215–224, 01 2013.

[9] N. Kakalis and Y. Ventikos, “Robotic swarm concept for efficient oil spill confrontation,” *Journal of hazardous materials*, vol. 154, pp. 880–7, 07 2008.

[10] M. Kayser, L. Cai, S. Falcone, C. Bader, N. Inglessis, B. Darweesh, and N. Oxman, “Design of a multi-agent, fiber composite digital fabrication system,” *Science Robotics*, vol. 3, no. 22, 2018. [Online]. Available: https://robotics.sciencemag.org/content/3/22/eaau5630

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

[12] F. Bravo, A. Vale, and I. Ribeiro, “Particle-filter approach for cooperative localization in unstructured scenarios,” *Lecture Notes in Electrical Engineering*, vol. 15, 01 2008.

[13] G. Huang, N. Trawny, A. Mourikis, and S. Roumeliotis, “Observability-based consistent ekf estimators for multi-robot cooperative localization,” *Autonomous Robots*, vol. 30, pp. 99–122, 09 2011.

[14] E. D. Nerurkar, S. I. Roumeliotis, and A. Martinelli, “Distributed maximum a posteriori estimation for multi-robot cooperative localization,” in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 1402–1409.

[15] A. Howard, M. J. Mataric, and G. S. Sukhatme, “Putting the ’i’ in ’team’: an ego-centric approach to cooperative localization,” in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 1, 2003, pp. 868–874 vol.1.

[16] X. S. Zhou and S. I. Roumeliotis, “Determining 3-d relative transformations for any combination of range and bearing measurements,” *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 458–474, 2013.

[17] A. Franchi, G. Oriolo, and P. Stegagno, “Mutual localization in multi-robot systems using anonymous relative measurements,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1302–1322, 2013. [Online]. Available: https://doi.org/10.1177/0278364913495425

[18] M. Ye, B. D. O. Anderson, and C. Yu, “Bearing-only measurement self-localization, velocity consensus and formation control,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 2, pp. 575–586, 2017.

[19] R. Falconi, S. Gowal, and A. Martinoli, “Graph based distributed control of non-holonomic vehicles endowed with local positioning information engaged in escorting missions,” in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 3207–3214.

[20] D. Katić and A. Rodić, “Intelligent multi robot systems for contemporary shopping malls,” in *IEEE 8th International Symposium on Intelligent Systems and Informatics*, 2010, pp. 109–113.

[21] P. Stegagno, M. Cognetti, G. Oriolo, H. H. Bülthoff, and A. Franchi, “Ground and aerial mutual localization using anonymous relative-bearing measurements,” *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1133–1151, 2016.

[22] A. Rashid and B. Abdulrazaaq, "A survey of multi-mobile robot formation control," *International Journal of Computer Applications*, vol. 181, pp. 12–16, 04 2019.

[23] A. Franchi, P. Stegagno, and G. Oriolo, "Decentralized multi-robot encirclement of a 3d target with guaranteed collision avoidance," *Autonomous Robots*, vol. 40, 07 2015.

[24] L. Siligardi, J. Panerati, M. Kaufmann, M. Minelli, C. Ghedini, G. Beltrame, and L. Sabattini, "Robust area coverage with connectivity maintenance," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2202–2208.

[25] M. Hossein Mirabdollah and B. Mertsching, "Bearing only mobile robots' localization: Observability and formulation using sis particle filters," in *2011 International Conference on Communications, Computing and Control Applications (CCCA)*, 2011, pp. 1–5.

[26] A. Martinelli and R. Siegwart, "Observability analysis for mobile robot localization," *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, 01 2005.

[27] R. Mahler, "The multisensor phd filter: I. general solution via multitarget calculus," *Proceedings of SPIE - The International Society for Optical Engineering*, 05 2009.

[28] B.Vo and W.Ma, "The gaussian mixture probability hypothesis density filter," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, 2006.

[29] W. Junjie, Z. Lingling, S. Xiaohong, and M. Peijun, "Distributed computation particle phd filter," 2015.

[30] A. Wasik, P. Lima, and A. Martinoli, "A robust localization system for multi-robot formations based on an extension of a gaussian mixture probability hypothesis density filter," *Autonomous Robots*, pp. 395–414, 2020.

[31] P. Dames and V. Kumar, "Autonomous localization of an unknown number of targets without data association using teams of mobile sensors," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 850–864, 2015.

[32] P. Stegagno, M. Cognetti, L. Rosa, P. Peliti, and G. Oriolo, "Relative localization and identification in a heterogeneous multi-robot system," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 1857–1864.

[33] Jingjing Wu, Yang Wang, and Shiqiang Hua, "Adaptive multifeature visual tracking in a probability-hypothesis-density filtering framework," *Signal Processing*, vol. 93, no. 2915-2926, pp. 850–864, 2013.

[34] "ODROID-XU4 Special Price – ODROID," Mar 2021, [Online; accessed 20. Mar. 2021]. [Online]. Available: https://www.hardkernel.com/shop/odroid-xu4-special-price

BIBLIOGRAPHY

"Cherokey: 4WD Mobile Robot for Arduino - DFRobot," Mar 2021, [Online; accessed 20. Mar. 2021]. [Online]. Available: https://www.dfrobot.com/product-896.html

"ODROID-XU4 Special Price – ODROID," Mar 2021, [Online; accessed 20. Mar. 2021]. [Online]. Available: https://www.hardkernel.com/shop/odroid-xu4-special-price

"Review: Low cost YDLidar X4 sees 360º all around itself," Mar 2021, [Online; accessed 20. Mar. 2021]. [Online]. Available: https://www.elektormagazine.com/news/review-low-cost-ydlidar-x4-sees-360-all-around-itself

Bakhshipour, M., Ghadi, M. J., and Namdari, F., "Swarm robotics search & rescue: A novel artificial intelligence-inspired optimization approach," *Applied Soft Computing*, vol. 57, pp. 708 – 726, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494617301072

Bravo, F., Vale, A., and Ribeiro, I., "Particle-filter approach for cooperative localization in unstructured scenarios," *Lecture Notes in Electrical Engineering*, vol. 15, 01 2008.

B.Vo and W.Ma, "The gaussian mixture probability hypothesis density filter," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, 2006.

Dames, P. and Kumar, V., "Autonomous localization of an unknown number of targets without data association using teams of mobile sensors," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 850–864, 2015.

Falconi, R., Gowal, S., and Martinoli, A., "Graph based distributed control of non-holonomic vehicles endowed with local positioning information engaged in escorting missions," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 3207–3214.

Franchi, A., Oriolo, G., and Stegagno, P., "Mutual localization in multi-robot systems using anonymous relative measurements," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1302–1322, 2013. [Online]. Available: https://doi.org/10.1177/0278364913495425

Franchi, A., Stegagno, P., and Oriolo, G., "Decentralized multi-robot encirclement of a 3d target with guaranteed collision avoidance," *Autonomous Robots*, vol. 40, 07 2015.

Hossein Mirabdollah, M. and Mertsching, B., “Bearing only mobile robots' localization: Observability and formulation using sis particle filters,” in *2011 International Conference on Communications, Computing and Control Applications (CCCA)*, 2011, pp. 1–5.

Howard, A., Mataric, M. J., and Sukhatme, G. S., “Putting the 'i' in 'team': an ego-centric approach to cooperative localization,” in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 1, 2003, pp. 868–874 vol.1.

Huang, G., Trawny, N., Mourikis, A., and Roumeliotis, S., “Observability-based consistent ekf estimators for multi-robot cooperative localization,” *Autonomous Robots*, vol. 30, pp. 99–122, 09 2011.

Jingjing Wu, Yang Wang, and Shiqiang Hua, “Adaptive multifeature visual tracking in a probability-hypothesis-density filtering framework,” *Signal Processing*, vol. 93, no. 2915-2926, pp. 850–864, 2013.

Junjie, W., Lingling, Z., Xiaohong, S., and Peijun, M., “Distributed computation particle phd filter,” 2015.

Kakalis, N. and Ventikos, Y., “Robotic swarm concept for efficient oil spill confrontation,” *Journal of hazardous materials*, vol. 154, pp. 880–7, 07 2008.

Katić, D. and Rodić, A., “Intelligent multi robot systems for contemporary shopping malls,” in *IEEE 8th International Symposium on Intelligent Systems and Informatics*, 2010, pp. 109–113.

Kayser, M., Cai, L., Falcone, S., Bader, C., Inglessis, N., Darweesh, B., and Oxman, N., “Design of a multi-agent, fiber composite digital fabrication system,” *Science Robotics*, vol. 3, no. 22, 2018. [Online]. Available: https://robotics.sciencemag.org/content/3/22/eaau5630

Mahler, R., “The multisensor phd filter: I. general solution via multitarget calculus,” *Proceedings of SPIE - The International Society for Optical Engineering*, 05 2009.

Martinelli, A. and Siegwart, R., “Observability analysis for mobile robot localization,” *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, 01 2005.

McGuire, K. N., De Wagter, C., Tuyls, K., Kappen, H. J., and de Croon, G. C. H. E., “Minimal navigation solution for a swarm of tiny flying robots to explore an unknown environment,” *Science Robotics*, vol. 4, no. 35, 2019. [Online]. Available: https://robotics.sciencemag.org/content/4/35/eaaw9710

Nerurkar, E. D., Roumeliotis, S. I., and Martinelli, A., “Distributed maximum a posteriori estimation for multi-robot cooperative localization,” in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 1402–1409.

Perera, R. T., Yuan, C., and Stegagno, P., “A phd filter based localization system for robotic swarms,” in *Proceedings DARS-SWARM2021 conference*, June 2021.

Rashid, A. and Abdulrazaaq, B., “A survey of multi-mobile robot formation control,” *International Journal of Computer Applications*, vol. 181, pp. 12–16, 04 2019.

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

Senanayake, M., Senthooran, I., Barca, J. C., Chung, H., Kamruzzaman, J., and Murshed, M., “Search and tracking algorithms for swarms of robots: A survey,” *Robotics and Autonomous Systems*, vol. 75, pp. 422 – 434, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889015001876

Siligardi, L., Panerati, J., Kaufmann, M., Minelli, M., Ghedini, C., Beltrame, G., and Sabattini, L., “Robust area coverage with connectivity maintenance,” in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2202–2208.

Stegagno, P., Cognetti, M., Oriolo, G., Bülthoff, H. H., and Franchi, A., “Ground and aerial mutual localization using anonymous relative-bearing measurements,” *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1133–1151, 2016.

Stegagno, P., Cognetti, M., Rosa, L., Peliti, P., and Oriolo, G., “Relative localization and identification in a heterogeneous multi-robot system,” in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 1857–1864.

Wasik, A., Lima, P., and Martinoli, A., “A robust localization system for multi-robot formations based on an extension of a gaussian mixture probability hypothesis density filter,” *Autonomous Robots*, pp. 395–414, 2020.

Ye, M., Anderson, B. D. O., and Yu, C., “Bearing-only measurement self-localization, velocity consensus and formation control,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 2, pp. 575–586, 2017.

Zahugi, E., Shanta, M., and Prasad, T., “Oil spill cleaning up using swarm of robots,” *Advances in Intelligent Systems and Computing*, vol. 178, pp. 215–224, 01 2013.

Zhong-yangZheng and Yang Tan, “Research advance in swarm robotics,” *Defence Technology*, vol. 9, pp. 18–39, 2013.

Zhou, X. S. and Roumeliotis, S. I., “Determining 3-d relative transformations for any combination of range and bearing measurements,” *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 458–474, 2013.