5-14-2024

Localization and Collision Prediction via RF Transceiver and Radar Methods

Aarti Singh
Washington University – McKelvey School of Engineering

WASHINGTON UNIVERSITY IN ST. LOUIS

McKelvey School of Engineering
Department of Electrical and Systems Engineering

Dissertation Examination Committee:
Neal Patwari, Chair
Roger Chamberlain
Ulugbek Kamilov
Chenyang Lu
Mark Minor

Localization and Collision Prediction via RF Transceiver and Radar Methods

by
Aarti Singh

A dissertation presented to
the McKelvey School of Engineering
of Washington University in
partial fulfillment of the
requirements for the degree
of Doctor of Philosophy

May 2024
St. Louis, Missouri

Table of Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I extend my deepest gratitude to my teammates at the SPAN Lab, for their dedication, insights, and collaboration, that have enriched every part of my academic experience. This shared journey has been nothing less than a privilege, in witnessing their intellect, diligence, and brilliance, which has served as a source of inspiration. Through the countless (odd) hours spent in the lab, with the white-board as our witness, navigating the student- and research-life in their presence has been both a relief and an important asset.

I also like to send my deepest appreciation to the members of my teams at University of Utah, both from the Smart Helmet project and from the Flux Research group, for bearing with our mostly virtual presence and sometimes patchy connections, keeping a tally of the time-difference while scheduling meetings, and most of all, for their intellectual expertise. Most importantly, and for making all this happen, to say that I am grateful that our advisor guided us and placed his belief in our potential, is an understatement. The insights, the discussions, and the disposition earned under his guidance, are something that will be carried throughout our careers.

I also extend my sincerest respect to the ESE department at Mckelvey School of Engineering, for providing both professional and personal support during this journey. The dedicated staff has been a vital resource in navigating the academic life here at WashU. I am thankful for the tireless efforts that are carried on behind the scenes by the ESE department that lets us pursue our academic goals seamlessly. Also, it has been an honor to have studied under all the faculty members at ESE department and CSE department, and they are placed in the highest regard, for showing us their commitment towards the progress of science and to the pursuit of knowledge, along with friendly nods in the hallways, which has made these years filled with lifelong lessons and memories.

Lastly, but certainly not least, I can never mention my family and friends enough, for their endless encouragement, understanding, and love, which has been nothing less than a blessing, and which have been the very reason through which my aspirations have survived. The support and patience has provided me immense comfort in moments of

doubt and boundless joy in otherwise productive moments. Thank you for the countless sacrifices, for constantly rooting and cheering for me, and for being here.

Signing this with my warmest regards,

Aarti Singh

Washington University in St. Louis
May 2024

To my parents, my brothers, Divya, and Carl.

ABSTRACT OF THE DISSERTATION

Localization and Collision Prediction via RF Transceiver and Radar Methods

by

Aarti Singh

Doctor of Philosophy in Electrical Engineering

Washington University in St. Louis, 2024

Professor Neal Patwari, Chair

In domains like automation, particularly in advanced driver-assistance and collision avoidance systems for vehicles, the need for reliable sensing and predictive capabilities is paramount. While current sensor technologies allow for quick responses to safety-critical events, there is a growing emphasis on predictive capabilities to anticipate and prevent such incidents. This opens new ideas for the development of sophisticated RF-sensors and algorithms capable of accurately predicting potential harms before they occur. Moreover, along with the benefits of enhanced sensing capabilities comes the imperative to safeguard user-privacy. Various techniques, including encryption and differential privacy, are employed to ensure that RF-based services do not compromise user data, while innovative approaches such as discarding irrelevant data and utilizing privacy-preserving measurements like received signal strength and Doppler shift offer promising avenues for balancing functionality with privacy concerns.

In this thesis, we integrate the power of radio frequency sensing and rapid progress of the data-driven learning to bring forth ideas that can be applied to safety-critical applications for enhancing efficiency and user experience in an increasingly interconnected world. We present three products that are built using three types of radio devices with differing in their operating principles and thus, their capabilities to be utilized in three

unique scenarios; network-based cooperative sensing, standalone sensing, and sensing as a surveyor/monitor.

Our first novel contribution is an infrastructure-free approach to collision prediction using ultra-wideband (UWB) signals and inertial sensing. They employ a cooperative strategy based on pairwise ranges and velocities to predict future collisions, utilizing an improved algorithm to estimate relative kinematics despite noisy measurements. This method is complementary to existing technologies dependent on object properties, with UWB chosen for its precise measurements and independence from indoor object properties.

We continue our endeavor by introducing a systematic shift in the type of sensor used, by instead using a standalone sensor, such as radar, for collision prediction in noisy, cluttered environments with dynamic motion. We utilize the radar-Doppler data and a convolutional neural network (CNN) to predict collisions, adapting features from the environment to handle inaccuracies. Online learning and automated labeling techniques are employed to make the CNN adaptable, with experiments resulting in a labeled dataset for validation against other methods.

Finally, in order to provide a method to implement these safety-critical applications, we investigate how the sensors can 'police' or survey an area and build applications from extracting only the relevant data from the RF-signal measurements through a new measurement called Doppler spread. Outdoor experiments in a densely populated area are conducted, generating a labeled database and examining Doppler spread's effectiveness for a privacy-preserving localization system based on fingerprinting.

Chapter 1

Introduction

The need for expanding the sensing capabilities beyond the five senses has co-existed with humans, both as a fascination and as an ambition. Within just over a century, that imagination have successfully realized into a technology centering the radio frequency spectrum, which has boosted the extents of how much, how further, and how well humans can study our surroundings. From the birth of this idea of using radio waves, in 'seeing' distant objects using sensors like Radars since the 1880s, to its growth into becoming the means of 'speaking' and 'hearing' across the globe since the 1940s, and to its present-day absolute integration into our day-today lives via location-based, gesture-based, and internet-based services, we have witnessed the usage of radio frequency integrate thoroughly into our lives, truly serving as our sixth sense.

Apart from vying for 'superpowers', one utility of acquiring this enhanced sensing is to interact safely in the common spaces shared with other agents. We envision a world where an agent equipped with radio transceivers, can sense, perceive, and more importantly, predict the imminent events that might be detrimental to itself or to others. As we experience the RF-based services becoming commonplace and increasingly so regarding the magnitude of their presence, the opportunity to use the abundant information about the environment sensed through the radio sensors is more lucrative than ever. There is a flourishing commercial market for RF-enabled products, for example, Google Soli's radar-based wellness tracking [49], UAVs and self-driving vehicles' ground-up autonomy using collision warning systems, and proximity detection for manufacturing industries and ambient assisted living [49]. Moreover, with advancements in the data-driven algorithms, added computational resources at our disposal, efficient usage of the RF-spectrum, and cost-effective sensors, we are collectively driven to innovate RF-based products that aid in accurate, timely decision making for safe interactions within the environment.

1.1 Motivation of Dissertation

By implementing products in workplaces, homes, and transportation systems, that are equipped with handy and convenient sensors, we can potentially reduce accidents and improve overall safety. Automated systems can eliminate human error, and data-driven algorithms and processing resources, this dissertation demonstrates with the help of three different kinds of devices, implemented in completely new environments for each, and using different signal processing and algorithmic practices, how to utilize the surplus of sensor data without human intervention. Also, incorporating considerate measures towards user privacy, we can better share radio spectrum and allow for users to accurately place and navigate themselves. With the technologies enquired in this dissertation successfully deployed in future, we anticipate people being safer in an ever increasingly measured and networked world.

1.2 Survey of Research Area

Safety critical applications The field of automation experiences the most numerous, frequent, and expensive encounters among the agents, such as unmanned aerial vehicles (UAVs), robots, self-driving cars. Thus, the leading field for the development of the safety-critical applications are the advanced driver-assistance system (ADAS) and collision avoidance system (CAS) for vehicles [38], [6], [85]. There are various flavors to these solutions in terms of the requirements, operation, and accuracy. For example, there are both centralized and distributed algorithms to process the collected sensor data by the agents and make decisions regarding their safe interaction with other agents [24], [18], [55]. Alternatively, the agents can use infrastructure-provided data for making these decisions themselves [4]. In many cases, multiple tagged agents become 'network' of nodes, where paired communication with other tagged nodes can provide infrastructure-free and cooperative decisions about their kinematics relative to each other. This serves to an advantage for building collision avoidance systems, since working with only the relative kinematics (such as relative range, relative velocity), rather than absolute coordinate information of the objects, is sufficient for achieving event-free interactions between agents [1].

Prediction is better than detection The implementation of a collision avoidance system that aid in safe interactions of agents can be multi-faceted, where the most common element is to process the information from sensors during any these event, for example the instance of collision between agents. However, any decision obtained from the data to maintain safety can happen only *after* the safety compromising incident has occurred. Current data from sensors do not predict impending events *before* such events happen. Therefore, there is a scope for incorporating decision making capabilities with more advanced RF-sensors and algorithms to instead predict the detrimental events before they occur, which is crucial in altering the agents to prepare themselves upon receiving an alert according to the prediction. There are efforts for camera-based collision prediction algorithm, camera-based helmet for avoiding rear end collisions in real-time [63], [10].

Egocentric sensing Apart from prediction, next important aspect in realizing true autonomy is through sensing of the entire environment for possible harmful events with both tagged and un-tagged objects, for which standalone sensors are required. LiDARs (light detection and ranging) are expensive sensors that provide rich information about scenes and targets, but can be susceptible to severe weather. Ultrasonic sensors are affordable and compact and are suitable for very short-range detection. Vision-based solutions do not perform well for long distances, in poor light, and in complex, real-world conditions. Again, following the progress in ADAS and CAS systems, where radars are extensively applied to sense all objects with electromagnetic properties different from air that are present in the field of view, without requiring additional sensors on the surrounding objects. Frequency modulated continuous wave (FMCW) radars are also inexpensive, compact sensors that can measure the relative kinematics (range and range-rate) of the objects in the field-of-view in real-time [15] and thus are ideal for being the RF-tags on wearables such as smart helmets. However, choosing radars as the RF-sensors opens various challenges to explore. For example, building solutions using radars suffer in performance due to unwanted scattering and reflection that are a function of the material properties of the other objects and require advanced filtering and pre-processing techniques [7]. Moreover, such clutter experiences drift in its properties and thus, applying data-driven methods to radars in dynamic environments is challenging [3]. This article explores and proposes an elaborate methodology employing concepts from the online learning domain as a measure to solve these challenges in real-world systems built using radars.

Privacy preserving as an additional requirement to RF sensing. Lastly, this *all-seeing* feature of radars makes the objects and other agents susceptible to be surveyed upon, while they simply are present in the surrounding of the main agent. [11] Despite the usefulness and usability of the service any product delivers, maintaining the privacy of users can be the cornerstone in the determining the success of the product. There is large body of research using encryption, outsourcing, differential privacy, parallelization techniques to ensure that any RF-based services are not exposing any private information of the agents through their shared RF-data. However, one straight forward approach is to discard all the (IQ) data from the RF-signals that is irrelevant to the service in question, and the only relevant information is to then be saved, processed, or shared by the agent, sans any private information in that data. The challenges then exist in developing novel concepts regarding what measurement from the data can provide users with a quality service. The RF-signals are most commonly measured through their received signal strength (RSS) value, which is also considered privacy-preserving, that provides provide relevant information regarding the relative interactions between two agents. However, it is extremely susceptible to fluctuations, therefore relying on RSS will negatively impact any agent's sensing and thus, decision making capabilities [81]. The information extracted in the form of time of arrival (ToA) and the time difference of arrival (TDoA) of the RF-signals, also carry the advantage of being privacy-preserving, however, despite the *ns* level precision ToA values attainable via UWB signals, ToA measurements from radios require diligent time synchronization and multiple, simultaneous measurements for effective decision making[81]. To instead achieve ego-centric sensing capability which involves only one agent, fingerprinting methods can provide solutions, where depending on the number of data-points, the sensor data from simply one agent to extract features of the relevant information from RF-signals is enough towards obtaining the service in question, while depending on the fingerprint used (RSS, ToA, or TDoA), we can maintain the privacy feature as well. One such measurement is the shift in frequency experienced by the RF-signals due to the movement between the agents. This change in frequency, called the Doppler shift, is function of the speed and direction of relative motion between the agents. It can be extracted by the agent to share with a centralized or a cooperative algorithm, while discarding rest of the information, thus is considered as a privacy-preserving measurement. However, the surroundings of the agents create a more complex measurement by modifying the Doppler shift into what we call the Doppler Spread, which is a function of the local surroundings and movements, as well as the speed and direction of

relative motion between the agents. This richer measurement, thus has the potential to serve as a location-dependent measurement as is the subject of academic inquiry.

1.3 Contributions

The following is a list of publications that have appeared to support the work towards this doctoral research.

1. Singh A., Patwari N., " Multipath-rich outdoor CW measurement dataset ", Feb 2023, https://github.com/aarti7/DSLoc.
2. Tadik, S., Singh, A., Mitchell, F., et al., "Salt Lake City 3534 MHz Multi-Transmitter Measurement Campaign", June 2023, https://github.com/serhatadik/slc-3534MHz-meas.
3. Aarti Singh and Neal Patwari, "Online learning for dynamic impending collision prediction using FMCW radar", *IEEE Transactions on the Internet of Things*, published 26 Aug 2023, https://doi.org/10.1145/3616018.
4. Singh, Aarti; Dr. Neal Patwari, 2023, "24GHz FMCW Radar Indoors Mobile", https://doi.org/10.7 Harvard Dataverse, V1
5. J. Phillip Smith, Anh Luong, Shamik Sarkar, Harsimran Singh, Aarti Singh, Neal Patwari, Sneha K. Kasera, and Kurt Derr, "A novel software defined radio for practical, mobile crowd-sourced spectrum sensing", *IEEE Transactions on Mobile Computing*, vol. 22, no. 3, March 2023.
6. A. Singh and N. Patwari, "Collision Prediction using UWB and Inertial Sensing: Experimental Evaluation," 2021 IEEE International Conference on Autonomous Systems (ICAS), Montreal, QC, Canada, 2021, pp. 1-5, doi: 10.1109/ICAS49788.2021.9551118
7. A. Singh and N. Patwari, "Range-based Collision Prediction for Dynamic Motion," 2021 IEEE 18th Annual Consumer Communications and Networking Conference (CCNC), Las Vegas, NV, USA, 2021, pp. 1-6, doi: 10.1109/CCNC49032.2021.9369608.

1.4 Structure of Dissertation

In Chapters 2 and 3, we explore an infrastructure-free setting, where the UWB signals between multiple nodes, along with and inertial sensing to test a cooperative strategy of using pairwise ranges and velocities for predicting future collisions. We utilize the MDS algorithm to produce high quality estimates of the relative kinematics of agents by using noisy inter-node range measurements and node's acceleration data. Predicting the future trajectory without requiring known-location infrastructure is carried out as be complementary method to other adopted technologies involving other sensors that are dependent on size, shape, reflective properties, luminescence of, and distance between the objects involved in the collision. We employ the UWB as a sensor of choice as it promises extreme precise measurements robust against the multi-path witnessed within the indoor environments and being independent of physical properties of the objects that are commonly present indoors.

In Chapter 4, FMCW radars are used for proposed as the sensor of choice for collision prediction in noisy and cluttered environments, when motion is dynamic and changing. The range and radial velocity (range-rate) information obtained from the radar-Doppler maps (RDMs) are used for predicting impending collision with the help of a convolution neural network (CNN). To tackle inaccuracies when measurements happen in cluttered environments, the proposed deep learning framework extracts adaptable features from the (continuously changing) environment, thereby eliminating the need for static filtering. Moreover, we apply online learning strategies, as well as automated labelling using accelerometer measurements, in order to make the the CNN classifier adaptable and learn from the recent history of the dynamically changing environment. The experiments leading to data collection are also published as a healthy, labelled dataset which mimics real-world collision scenarios. We use the dataset to validate that our proposed method performs well in comparison with other non-learning collision warning methods and traditional machine learning methods.

In Chapter 5, we explore the additional feature of providing privacy preserving capabilities to the RF-signal measurements by proposing a new form of measurement, called the Doppler spread. A series of outdoor experiments in a densely populated city-wide area are conducted and the details of the final elaborated, labelled database are provided,

along with examining the characteristics of the Doppler spread as an effective, privacy-preserving data for a fingerprinting-based localization system.

Chapter 2

Range-based Collision Prediction for Dynamic Motion

2.1 Introduction

Autonomous and real-time collision prediction and collision avoidance is crucial in a world filled with multiple mobile entities operating in the close vicinity of each other. Collisions, which do happen [22], are both life-threatening and expensive. GPS and lidar are insufficient to reliably predict the collision of small objects moving quickly towards each other, e.g., multiple drones, or a smart helmet and a baseball. In many cases, it will be possible to add a radio frequency (RF) tag to robots, vehicles, or objects that need to monitor to avoid collisions. However, RF tag localization systems are insufficient to predict collisions because they do not predict future positions, and they require a fixed, known-location infrastructure which may not be present or may be too inconvenient or expensive to deploy for the application. Finally, using coordinates as an intermediate step in the process of predicting collisions is sub-optimal [1]. We argue that, fundamentally, collision prediction from range measurements should be distributed. In this paper, we present a method to address this gap by enabling mobile agents of any size or speed to predict impending collisions without relying on a centralized infrastructure.

A known-location infrastructure (or anchors) is usually utilized for obtaining the absolute coordinate knowledge of the mobile objects as is done in [65], however, their availability may not be always possible. In addition of deployment and maintenance costs, the infrastructure can be a single point of failure. In cases where GPS is used to know the locations of the infrastructure, GPS signals will be unavailable indoors or in tunnels for

example, thus, rendering a GPS-based collision avoidance system inaccurate and ineffective. Moreover, deploying a local positioning system infrastructure is inappropriate when robots must operate globally or in emergency situations e.g. first-responder drones or self-driving vehicles. Fortunately, collision between two objects does not require the coordinates of each object in a global coordinate system because the collision between two objects is a matter of their relative kinematics, such as their relative position, velocity, and acceleration. The perspective of this paper is that collision prediction should be local, distributed, and relative, freeing us from requiring an infrastructure or a global reference.

A popular approach to obtain relative positions is multidimensional scaling (MDS) [72]. In MDS, 'dissimilarities' between each pair of objects (e.g., ranges between robots in our case) are mapped into a low dimensional (2D or 3D) relative coordinate as an output. The distances between robots are preserved as much as possible and a relative position mapping of the robots is formed without the use of known-location infrastructure. However, MDS is a centralized algorithm as it requires all the dissimilarities to be known by one processing unit. For N robots, classical MDS has a computational complexity of $O(N^3)$. A distributed method of estimating location is proposed in [21], is but implemented using an infrastructure. Based on the same work, another approach is presented to obtain a relative map of objects in motion, which although does not require known-location infrastructure, but is centralized in its implementation [12]. However, in order to achieve true autonomy and independence from a centralized decision maker, collision detection and prediction decision should be made in a decentralized manner by each device.

Another challenge with using MDS to generate kinematics over a time period is that since there is no fixed frame of reference, the generated map can undergo random translation, rotation, and flip. Therefore, without infrastructure, successive application of MDS over time is going to provide incorrect kinematics. A modification of classical MDS such that a common frame of reference is maintained for position and higher order kinematics (velocity and acceleration) are obtained is implemented in [60]. Using higher order derivatives of squared distance measurements, the relative kinematics are estimated. However, this method is highly sensitive to noise in the range measurements. However, in order to predict collision we need relative kinematics which are tolerant to noisy measurements.

We present a robust, decentralized, infrastructure-less algorithm that produces high quality estimates of the relative kinematics of robots by using noisy inter-node range measurements and intra-node acceleration data. With the estimated kinematics, this distributed scheme predicts the future trajectory without requiring known-location infrastructure and any impending collision. In addition to the decentralized and infrastructure-free approach, our solution can be complementary to other widely adopted technologies for collision prediction. These technologies involving either active sensors such as lidar [79], radar [76], or passive sensors [19] such as cameras, are dependent on size, shape, reflective properties, luminescence of, and distance between the objects involved in the collision. The algorithm presented in this paper is free of such limitations involving physical properties of the objects.

2.2 Problem Statement

We take a network of N unknown-location nodes in a D-dimensional space. Over a period of time, node i collects pairwise range measurements between itself and its neighbors $\mathcal{N}_i$, and we use $\delta_{i,j}^t$ for $j \in \mathcal{N}_i$ at time t. A real-world scenario is considered in which nodes move with arbitrary motion. The problems we explore include:

1. estimation of the coordinates $\boldsymbol{x}_i^t$ for $i = 1, \ldots, N$ and $t = 1, \ldots, T$, where $\boldsymbol{x}_i \in \mathcal{R}^D$ given pairwise range measurements $\{\delta_{i,j}^t\}$ and individual acceleration measurements, $\boldsymbol{\alpha}_i^t$, both taken over the time window $t = 1, \ldots, T$;
2. predicting an impending collision between i and any neighbor node in $\mathcal{N}_i$ at a time soon after $t = T$.

We assume that the primary goal of the system is to predict collisions, but in the case of an impending collision, the recent positions may be useful for the system reaction, for example, to know which direction to swerve.

2.3 Proposed Algorithm

To achieve the goals we articulate in the Introduction, in this section we formulate a new cost function based on errors between measured and calculated ranges and acceleration over time for all nodes. Our insight is that our formulation allows for a distributed, low computational complexity algorithm in which each device estimates its recent positions locally. It minimizes its local cost function and broadcasts its position estimates to its neighbors. Each node successively refines its recent position estimates based on its range and acceleration measurements in addition to the most recent received position estimates from its neighbors. This distributed nature ensures a decrease in the local cost functions, which contribute additively. Thus each sensor contributes to the minimization of the global cost function. Further, the local optimization step is low complexity because it is based on finding the minimum of a quadratic expression. Finally, the distributed optimization is guaranteed to converge, because it is using majorization approach which guarantees each round's global cost is non-increasing. Our approach follows the *scaling by majorizing a complicated function* (SMACOF) [27] approach, but expands it to enable simultaneous estimation of multiple recent positions, and for use of acceleration measurements in the cost function. After we present our algorithm for recent position estimation, we then present in Section 3.3.2 how these estimates are used to extrapolate and predict collisions.

2.3.1 Proposed Cost Function

Consider N mobile nodes with their position at time t represented as $X^t = [\boldsymbol{x}_1^t, .., \boldsymbol{x}_N^t]^T$. Our global cost function is:

$$\begin{aligned} S = &\sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} \sum_{t=1}^{T} w_{i,j}^t \left[\delta_{i,j}^t - d_{i,j}(X^t) \right]^2 \\ &+ \sum_{t=1}^{T} r_i \left[\boldsymbol{\alpha}_i^t - \boldsymbol{a}_i(\boldsymbol{x}_i^t) \right]^2, \end{aligned} \tag{2.1}$$

where the first term represents the error between measured distances $\delta_{i,j}^t$ and the actual distances based on location coordinates $d_{i,j}(X^t)$, which are calculated as,

$$d_{i,j}(X^t) = \|\boldsymbol{x}_i^t - \boldsymbol{x}_j^t\| = \sqrt{(\boldsymbol{x}_i^t - \boldsymbol{x}_j^t)^T(\boldsymbol{x}_i^t - \boldsymbol{x}_j^t)}. \tag{2.2}$$

Whenever a node's velocity changes, its non-zero acceleration is measured by its accelerometer. We incorporate this extra information in the latter part of the sum, representing the error between the measured acceleration $\boldsymbol{\alpha}_i^t$ and acceleration $\boldsymbol{a}_i^t$ calculated from the coordinate path as follows:

$$\begin{aligned} \boldsymbol{a}_i(\boldsymbol{x_i^t}) &= \left(\boldsymbol{x}_i^{t+1} - \boldsymbol{x}_i^t\right) - (\boldsymbol{x}_i^t - \boldsymbol{x}_i^{t-1}) \\ &= \boldsymbol{x}_i^{t+1} + \boldsymbol{x}_i^{t-1} - 2\boldsymbol{x}_i^t. \end{aligned} \tag{2.3}$$

Our goal is to minimize this cost function in a distributed manner to provide optimal location estimates $\{\boldsymbol{x_i^t}\}$ for node.

Following is an equivalent expression for S:

$$S = \sum_i S_i. \tag{2.4}$$

Therefore, at each node i, we have a local cost function as:

$$\begin{aligned} S_i = &\sum_{t=1}^{T} \sum_{j \in \mathcal{N}_i} w_{i,j}^t \left[\delta_{i,j}^t - d_{i,j}(X^t)\right]^2 \\ &+ \sum_{t=1}^{T} r_i \left[\boldsymbol{\alpha}_i^t - \boldsymbol{a}_i(\boldsymbol{x_i^t})\right]^2. \end{aligned} \tag{2.5}$$

We note that S_i is local to i since it only depends on the measurements available at i and positions of neighbour nodes. Minimizing this local cost function will result in position estimates of this node. The majorization approach guarantees non-increasing local cost. Implementing our approach at each robot constructs the backbone of this distributed method. Each local cost distributes additively over the network, thus each sensor contributes to the minimization of the global cost function (2.1) by minimizing its own local

cost function (2.5). This way, our algorithm produces a sequence of position estimates with non-increasing global cost.

2.3.2 Majorization

To minimize of our local cost function, we use a majorization approach inspired by SMACOF [27], but adding acceleration, and with multiple measurements over time. SMACOF is a gradient-decent algorithm which majorizes the cost function of the MDS problem. Iteratively minimizing the majorizing function guarantees a monotonously decreasing sequence of cost values. Our algorithm starts the minimization of its cost function as per equation (2.5) from an initial estimate of position and updates the estimates until the update in position is smaller than a certain value after a certain number of iterations. The final position estimates are the coordinates which minimize the summed weighted squared error. The majorizing function $T_i(\boldsymbol{x}_i, \boldsymbol{y}_i)$ of $S_i(\boldsymbol{x}_i)$ satisfies: (i) $T_i(\boldsymbol{x}_i, \boldsymbol{y}_i) \geq S_i(\boldsymbol{x}_i)$ for all y_i, and (ii) $T_i(\boldsymbol{x}_i, \boldsymbol{x}_i) = S_i(\boldsymbol{x}_i)$. The majorizing function $T_i(\boldsymbol{x}_i, \boldsymbol{y}_i)$ is thus defined as:

$$\begin{aligned} T_i(x_i, y_i) = \sum_{t=1}^{T} \sum_{j \in \mathcal{N}_i} w_{i,j}^t [(\delta_{i,j}^t)^2 + (d_{i,j}(X^t))^2 \\ - \frac{2\delta_{i,j}^t}{d_{i,j}^t(Y)} (\boldsymbol{x}_i^t - \boldsymbol{x}_j^t)^T (\boldsymbol{y}_i^t - \boldsymbol{y}_j^t)] \\ + \sum_{t=1}^{T} r_i \left[(\boldsymbol{\alpha}_i^t - \boldsymbol{a}_i^t(\boldsymbol{x}_i^t) \right]^2 . \end{aligned} \tag{2.6}$$

Here, the majorization function T_i is quadratic in nature, so we can find its minimum analytically. The analytical solution comes from differentiating it with respect to $\{\boldsymbol{x}_i^t\}$ and equating it zero. This minimization is done iteratively until convergence is reached to provide the values of $\{\boldsymbol{x}_i^t\}$ for which the majorized cost function S_i is low.

2.3.3 Proposed Algorithm

The proposed method is described in Algorithm 1. It should be noted that,

1. The algorithm requires initialized positions $X^{(0)}$. Generally, each round is initialized using the projected coordinates of positions from the previous round's estimates. The first time a neighbor j appears to node i, it must somehow provide the algorithm with $\{\boldsymbol{x}_j^t\}_t$. We leave a distributed initialization for this infrequent case to future work. Here, we use classical MDS to generate $\{\boldsymbol{x}_j^t\}_t$ when there is no estimate from a prior round.

2. The Euclidean distances (used here as the 'dissimilarities') do not have a frame of reference. They are measured locally – node i computes its range to each neighbor, and node i also measures its own acceleration vector $\boldsymbol{\alpha}_i^{(t)}$.

3. The weights applied to measured acceleration r_i should be chosen to account for accuracy of acceleration measurement for each node i. Lower values of r_i indicate more noisy measurements.

4. For all the measured $\delta_{i,j}$, the weights $w_{i,j}$ are simplistically set as equation 2.7, in essence making *all* the other j nodes as i's neighbors i.e. $j \in \mathcal{N}_i$.

$$w_{i,j} = \begin{cases} 1, & \text{if } \delta_{i,j} \text{ is measured.} \\ 0, & \text{otherwise.} \end{cases} \tag{2.7}$$

   However they can be adaptively set as a function of measurements $\delta_{i,j}$ such as to reflect its accuracy, such that less accurate measurements are down-weighted in the overall cost function and only the nodes with less noisy measurements are counted as neighbors.

The two helper variables used in the algorithm are q and b given by:

$$q_i^{-1} = \sum_{j \in \mathcal{N}_i} w_{i,j} + r_i \tag{2.8}$$

and $\boldsymbol{b}_i^{(k)} = [b_1, b_2, .., b_N]^T$ is a vector given by

$$\begin{aligned} b_j &= w_{i,j}[1 - \delta_{i,j}^t / d_{i,j}(X^{(k)})] \\ b_i &= \sum_{j \in \mathcal{N}_i} w_{i,j} \delta_{i,j}^t / d_{i,j}(X^{(k)}) \end{aligned} \tag{2.9}$$

Algorithm 1: Location estimation with ranges and acceleration data

Inputs : $\left\{\delta_{i,j}^t\right\}, \{w_{i,j}^t\}, \epsilon, \{r_i\}, X^{(0)}, \{\boldsymbol{\alpha}_i^t\}$ ***Initialize*** : $k = 0, S^{(0)}, a_i$ compute q_i from Equation (2.8)

repeat

  k = k+1;

  for $i = 1$ **to** N **do**

    for $t = 1$ **to** T **do**

      compute $\boldsymbol{b}_i^{k-1}$;

      $\boldsymbol{x}_i^t \leftarrow q_i r_i(2\boldsymbol{x}_i^{t+1} + 2\boldsymbol{x}_i^{t-1} - \boldsymbol{x}_i^{t+2} - \boldsymbol{x}_i^{t-2} + \boldsymbol{\alpha}^{t+1} + \boldsymbol{\alpha}^{t-1} - \boldsymbol{\alpha}^t) + q_i X^{(k-1)} \boldsymbol{b}_i^{(k-1)}$;

      $S^{(k)} \leftarrow S^{(k)} - S_i^{(k-1)} + S_i^{(k)}$;

    send $\left\{\boldsymbol{x}_i^t\right\}_t$ to friend nodes;

    send $\boldsymbol{S}^{(k)}$ to node $(i+1)$ mod N;

until $S^{(k-1)} - S^{(k)} < \epsilon$;

2.3.4 Regression Based Collision Prediction Algorithm

Mobile agents in a network run a risk of colliding with each other and thus, need to have autonomous decisions making capabilities whether to pursue or move away from a trajectory. Predicting any future collisions between two agents involves predicting whether individual motions of those two moving agents will intersect, which depends on the knowledge of relative kinematics of those two. Using the relative locations estimated from previous stage, we can predict locations into the nearby future. Regression analysis is widely used for prediction and forecasting, as they reveal the causal relationships between a dependent variable and one or a collection of independent variables in a fixed dataset, which can later be used to estimate causal relationships using new observational data.

We choose polynomial regression since it works very well for non-linear interpolation problems. In our case, we seek to predict the future locations of the agent or robot which has a non-linear relationship with time due to the dynamic nature of the motion. The output of the polynomial regression in such a scenario can also be interpreted as higher order kinematics. Given T data points $(t_i, \boldsymbol{x}_i)$, where, dependent variable t_i is a time instance and $\boldsymbol{x}_i$ is the corresponding location of a robot i, for $i = (1, 2, ...T)$, we fit a 2^{nd} degree polynomial to approximate the robot's locations,

$$x_i = p_2 + p_1 t_i + p_0 t_i^2 \tag{2.10}$$

Here, p_0, p_1, and p_2 make the coefficient of this polynomial that predicts the location x_i with estimation errors. We use least squares approximation, in which the polynomial coefficients can be obtained by minimizing the sum of the error squares. These coefficients are used to extrapolate values for the polynomial, giving us future relative locations, thereby, giving us future inter-object distances of two robots. We define *future* as a time-frame that is equal to or less than the reaction time of the robot. If the minimum inter-robot distance threshold between two robots is crossed within this time into future, a collision is predicted. We define collisions based on various distance thresholds and can detect collisions by comparing the extrapolated distances with these user-defined distance thresholds. In Section 2.4, we evaluate the performance of this regression-based collision prediction method for different distance thresholds, while also analyzing the location estimated from our algorithm.

2.4 Results

In this section we test our proposed algorithm in terms of location estimation and collision prediction using simulated data. We demonstrate the performance of the proposed algorithm on a network of 4 robots with unknown location arranged on a uniform grid of 4x4 area units. Three robots are stationary and one robot is made to travel in a circular motion which contributes to it having acceleration. No known-location infrastructure is present and only inter-robot ranges and each robot's acceleration are calculated and used by our proposed algorithm. We trust that this simulation can be realized with an experiment conducted with any autonomous robots having UWB tags for ranging and inertial measurement units for acceleration measurements. In that case, the ranges obtained will be prone to error and noise. For this preliminary study, we introduce noise to ranging measurement, with standard deviation of 0.02 meters as per the findings in [1]. Using this range data, we present the results in the following section.

2.4.1 Location Estimation

We first demonstrate the quality of location estimates generated from our algorithm. For the $N = 4$ robots, 1 robot moving in a near-circular motion for multiple laps and 3 being stationary (thus, all having relative motion), we are able to track the curved trajectory that was followed by the mobile robot over time. We take a window of 20 instances at once $(T = 20)$, and for all i in N, we estimate 20 locations $\{x_i^t\}$, one for each time instance of the window. In a sliding window manner, next window's 20 estimates are calculated. We keep the middle point $(T/2)$ as an 'average' solution from that window with respect to the center point of that window's ground truth. Plotting $T/2's$ estimates versus ground truth, in Figure 3.2 we see location estimates for only robot-2 (n2). The ground truth of each robot and its estimated positions are marked 'o' and '$\triangleright$' respectively, where the lines represent the offset between the estimates and ground truth. In Figure 2.2, each window's such middle location estimates (in red) for robot-2 are plotted against its ground truth (black) for its circular trajectory. Also, the stationary robots (n $0, 1, 3$) are plotted for their estimated locations and ground truth, alongside robot-2's circular motion so as to provide a complete picture of the experiments. We see that our algorithm is able to estimate the locations of all the robots, more importantly of robot-2, and commendably able to follow the trajectory of robot-2's motion over time.

Next, we compare the performance of our algorithm with another MDS-based localization algorithm [60]. For one time window, this method provides joint higher order relative kinematics estimates (position, velocity, and acceleration), however, they are extremely sensitive to ranging noise. Figure 3.3 shows the root mean square error (RMSE) of location estimated for each robot by the two methods when compared to ground truth. We can see that our algorithm outperforms the other MDS-based implementation, which is credited to our algorithm's capability of providing better position estimates by incorporating the robots' acceleration information in cost function minimization. This makes the algorithm to trace the motion well and provide a lower RMSE by one order of magnitude when compared to the other MDS-based method.

We emphasizes that none of the robots' locations are known before the start of our algorithm, i.e. there is no known-location infrastructure (anchors) present in the system. Every node independently runs our proposed algorithm to estimate an optimal solution for its position. That is, after receiving it's neighbours optimal solution and using its own

acceleration, each node conducts a successful minimization of its local cost function as explained in Section 2.3.2. However, we translate one device to become the origin ($n0$) to constitute a consistent frame of reference, although it should be noted that this does not change the relative positions among the robots, and yet is helpful when we need to compare against a ground truth with similar translation. Additionally, the received estimates from one device's neighbours can be noisy and hence can impact the optimal solution for that node. We explore the performance of estimated locations when different amount of noise is present in measured inter-robot ranges. In Figure 2.3b, we see that as the noise in the range measurements is increased, the error in the position estimated with our algorithm also increases for all the nodes n1,n2, and n3.

In order to avoid collision, a device or robot needs to make a prediction whether it is going to collide or not, before it happens. For collision prediction, only relative position between two robots is enough to predict any future distance threshold violation that counts as collision. Using relative location estimates from our algorithm, we extrapolate distances from the model explained in 3.3.2. Let us say that the future separation i.e. distance $\hat{d}^t_{i,j}$ between two robots i and j goes lower than a threshold at some time into the nearby future. We define future as a time window equal to reaction time of a robot. If the predicted distances between two robots go under a set threshold within this window, robots can not swerve to avoid collision and collision in the future is predicted. The reaction time τ for each robot is taken as 0.09 *seconds* for all simulations. Within this reaction time into the future, if the predicted distances distance $\hat{d}^t_{i,j}$ is smaller than any pre-decided minimum-distance threshold d_{thd}, it is counted as a collision. Note that separation between the center of two robots is at least 2 times its radius r, which is equal to the diameter of one robot (0.34 meter is the diameter of one common irobot). Adding extra distance value ϵ_d to $2r$, we get minimum-distance threshold, given as

$$d_{thd} = \epsilon_d + 2r \tag{2.11}$$

If future inter-robot distances into the future are under this value it is counted as a collision. Figure 3.4 shows the receiver operating characteristic (ROC) where the probability of detection (P_D) of collision is given as the function the probability of false alarm (P_{FA}). We report that our extrapolation based collision prediction model is able to provide higher probability of detection for the same probability of false alarms when compared with the

other kinematics method FACT [1]. Our algorithm gives 100% detection with a 2% false alarm, where as FACT gives a detection of 80% at the same false alarm rate. Another collision detection approach based on pairwise regression performs even worse. This is because our algorithm's localization provides accurate position estimates for any degree of complex motion, which lets us extrapolate complex trajectories very well, giving extremely accurate future inter-robot distances and kinematics to predict collisions. The 2^{nd} degree regression's coefficients are able to extrapolate the future locations while taking each robot's acceleration into account, a trait not achievable by FACT or pairwise regression.

Lastly, as explained in Section 3.5, robots equipped with UWB tags for ranging have inter-robot ranges prone to noise. Prediction of future locations, and thereafter collisions, by using these noisy range measurements can be studied as a function of noise. In order to investigate the effect of noise, we test our collision prediction methodology for noisy range measurements with varying standard deviation. The results corroborates the intuition that the detection accuracy deteriorates with increase in noise as is shown in Figure 2.4b.

2.5 Conclusions

This paper presented a new approach to estimate locations and predict collisions for systems involving mobile devices that are capable of communicating their range measurements to each other. Our method uses each device's acceleration into estimating location coordinates for every device, achieving a commendable location estimation performance. The algorithm extends further by predicting collisions into future and it does not require a known-location infrastructure or a central decision maker to achieve a good collision prediction performance. We test its performance in simulation settings and provide a detailed analysis of results.

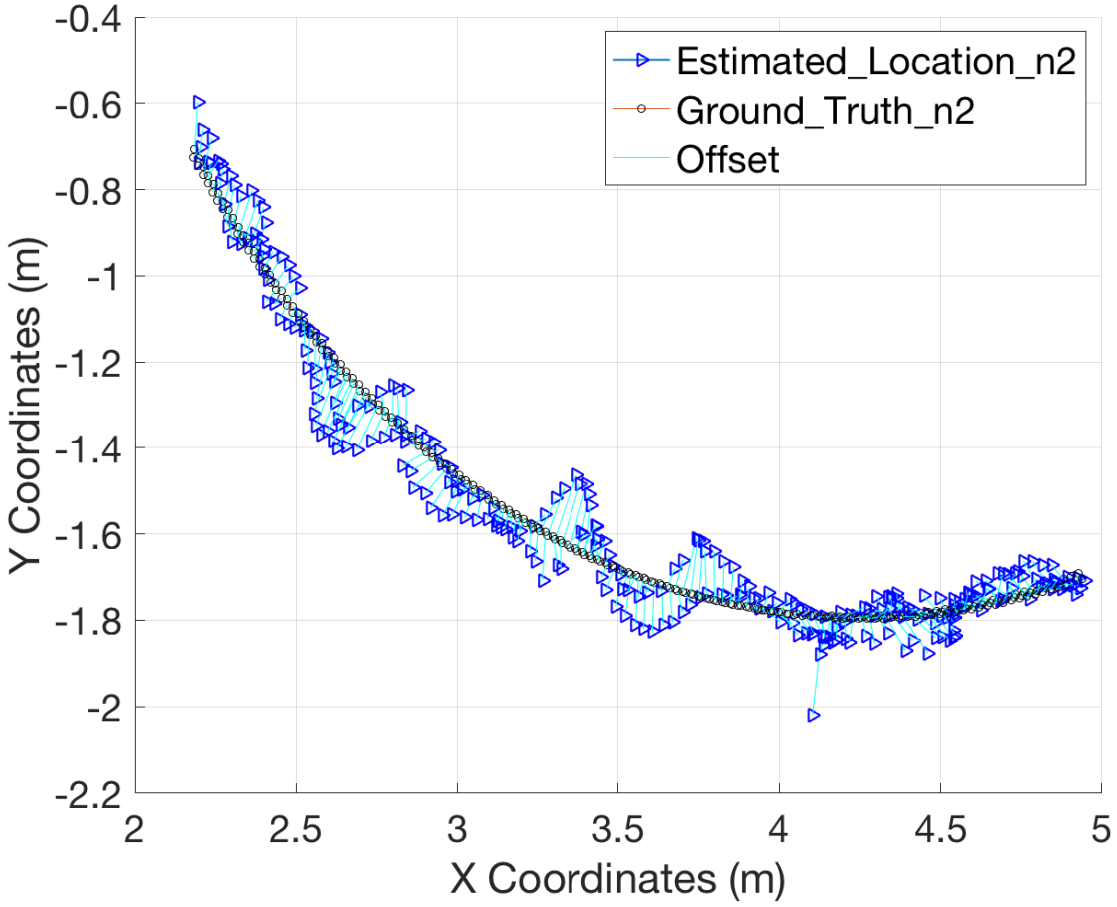


Figure 2.1: Location estimates only for the moving robot: Blue triangles represents our algorithm's location estimates plotted on top of black circles representing the ground truth of the robot in motion. The measured distances have added noise with st. deviation of 0.02 and algorithm runs only when a user-defined proximity distance threshold is crossed.

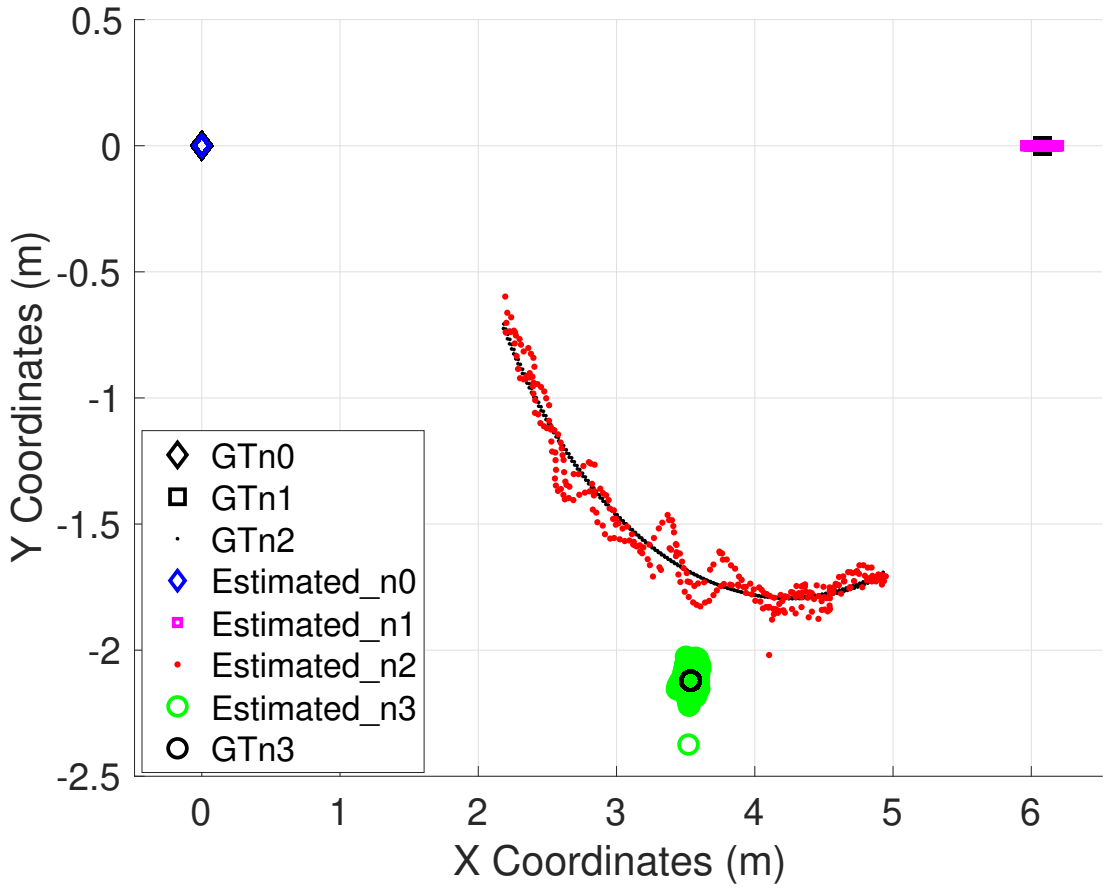
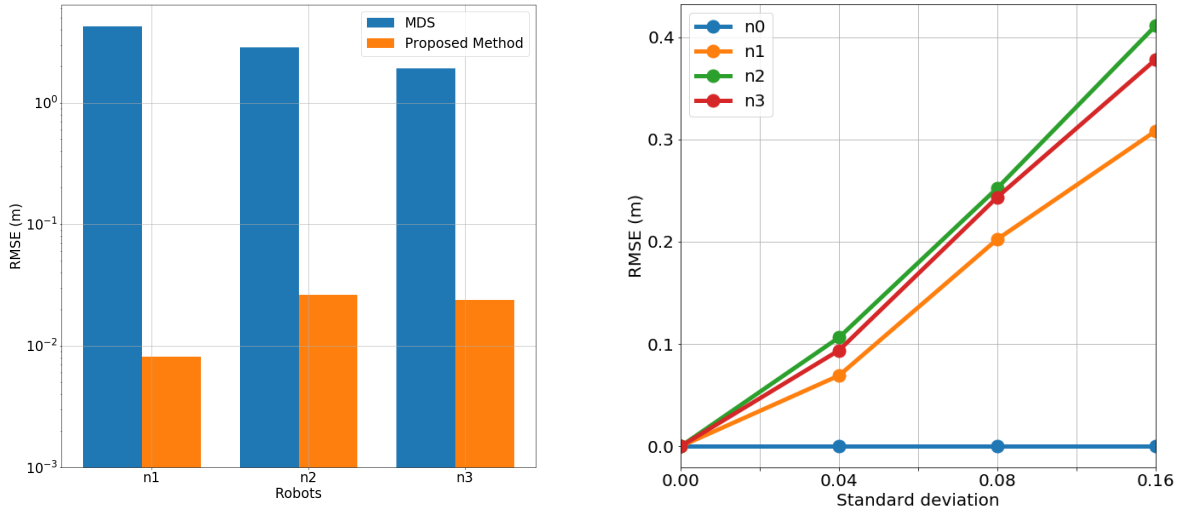


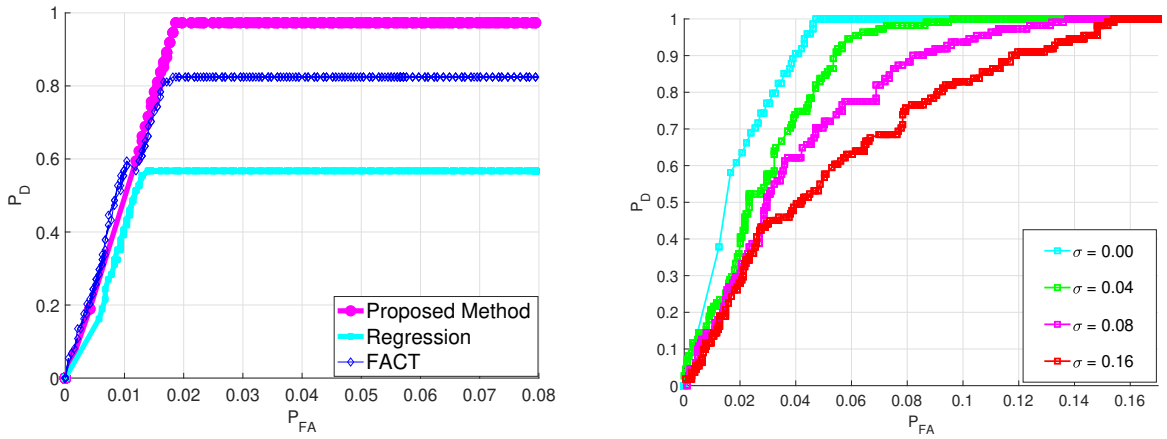
Figure 2.2: Location estimates for all the robots in the system (4 here): Each robot's locations estimated by our algorithm are plotted on top of black markers representing the ground truth for that respective robot. The measured distances have added noise with st. deviation of 0.02 and algorithm runs only when a user-defined proximity distance threshold is crossed.



(a) RMSE comparison location estimates from MDS and our algorithm, with distances having added noise of standard deviation 0.02

(b) RMSE of the position estimates for each node versus various levels of noise in the range measurement

Figure 2.3: Receiver Operating Characteristic



(a) Receiver Operating Characteristic plot for a comparison between our collision prediction method, FACT, and pairwise regression

(b) Proposed Method's Receiver Operating Characteristic plot for various amount of noise added into ranges collected

Figure 2.4: Receiver Operating Characteristic

Chapter 3

Experiments for Range-based Collision Prediction for Dynamic Motion

3.1 Introduction

Autonomous and real-time collision prediction and collision avoidance is crucial in a world filled with multiple mobile entities operating in the close vicinity of each other. Collisions, which do happen [22], are both life-threatening and expensive. GPS and lidar are insufficient to reliably predict the collision of small objects moving quickly towards each other, e.g., multiple drones, or a smart helmet and a baseball. In many cases, it will be possible to add a radio frequency (RF) tag to nodes, vehicles, or objects that need to monitor to avoid collisions. However, RF tag localization systems are insufficient to predict collisions because they do not predict future positions, and they require a fixed, known-location infrastructure to calculate global map of nodes' locations, which may not be present, inconvenient, or expensive to deploy for the application. We argue that, fundamentally, collision prediction from range measurements should be distributed, local, and relative. Fortunately, collision between two objects does not require the coordinates of each object in a global coordinate system because the collision between two objects is a matter of their relative kinematics, such as their relative position, velocity, and acceleration. In this paper, we present a method to address this gap by enabling mobile agents of any size or speed to predict impending collisions without relying on a centralized infrastructure or a global reference.

A popular approach to obtain relative positions is multidimensional scaling (MDS). In MDS, 'dissimilarities' between each pair of objects are mapped into a low dimensional

relative coordinates such that the distances between nodes are preserved as much as possible. MDS has been utilized in the localization research extensively [65], however, MDS is a centralized algorithm as it requires all dissimilarities to be known by one processing unit. For N nodes, classical MDS has a computational complexity of $O(N^3)$. A distributed method of estimating location is proposed in [21], is implemented with known-location infrastructure nodes. Based on the same work, another approach is presented to obtain a relative map of objects in motion, which although does not require known-location infrastructure, but is centralized in its implementation [12]. Another challenge with using MDS to generate relative kinematics over a time period is that, since there is no fixed frame of reference, the generated map can undergo random translation, rotation, and flip. Therefore, without infrastructure, successive applications of MDS over time provides incorrect kinematics. A modification of classical MDS such that a common frame of reference is maintained for position and higher order kinematics (velocity and acceleration) is implemented in [60]. The relative kinematics are estimated using higher order derivatives of squared distance measurements. However, this method is highly sensitive to noise in range measurements. In order to predict collision from RF range measurements we need relative kinematics estimators which are tolerant to noisy measurements. One extension of this work is implemented to produce a kinematics based collision prediction model, but it is limited to only linear motion [1]. We present a robust, decentralized, infrastructure-less algorithm 'Autonomous Collision Estimation for Dynamic Motion' (ACED), that produces quality relative kinematics of moving objects by using noisy inter-node range measurements and intra-node acceleration data. With the estimated kinematics, this distributed scheme predicts the future trajectory and any impending collision without requiring known-location devices. In addition, our solution can be complementary to other widely adopted technologies for collision prediction. These technologies involving either active sensors such as lidar [79] or radar [76], or passive sensors [19] such as cameras, are dependent on size, shape, reflective properties, luminescence of, and distance between the objects involved in the collision. The algorithm presented in this paper is free of such limitations involving physical properties of the objects.

3.2 Problem Statement

We take a network of N unknown-location nodes in a D-dimensional space. Over a period of time, node i collects pairwise range measurements between itself and its neighbors $\mathcal{N}_i$. We denote the range measurement between node i and $j \in \mathcal{N}_i$ at time t as $\delta_{i,j}^t$. A real-world scenario is considered in which nodes move with arbitrary motion. The problems we explore include:

1. estimation of the coordinates $\boldsymbol{x}_i^t$ for $i = 1, \ldots, N$ and $t = 1, \ldots, T$, where $\boldsymbol{x}_i \in \mathcal{R}^D$ given pairwise range measurements $\{\delta_{i,j}^t\}$ and individual acceleration measurements, $\boldsymbol{\alpha}_i^t$, both taken over the time window $t = 1, \ldots, T$;
2. predicting an impending collision between i and any neighbor node in $\mathcal{N}_i$ at a time soon after $t = T$.

We assume that the primary goal of the system is to predict collisions, but in the case of an impending collision, the recent positions may be useful for the system reaction, for example, to know which direction to swerve.

3.3 Proposed Algorithm

To achieve the goals we articulate in the Introduction, in this section, we formulate a new cost function based on errors between measured and calculated ranges and acceleration over time for all nodes. Our insight is that distributed tracking can be formulated in a distributed, low computational complexity manner by using a majorization framework. In this framework, each device estimates its recent positions locally by minimizing its local cost function and broadcasting its newest position estimates to its neighbors. Each node successively refines its recent position estimates based on its range and acceleration measurements in addition to the most recent received position estimates from its neighbors. The majorization approach ensures non-increasing local cost functions. Since these local costs contribute additively to the global cost, thus, the global cost function will be non-increasing. Further, the local optimization step is low complexity because it is based on finding the minimum of a quadratic (majorizing) expression. Finally, the distributed

optimization is guaranteed to converge, because the majorization approach guarantees each round's global cost is non-increasing. Our algorithm follows similarly to the *scaling by majorizing a complicated function* (SMACOF) [16] approach, but expands SMACOF to enable simultaneous estimation of multiple recent positions, and to enable use of acceleration measurements in the cost function.

3.3.1 Proposed Cost Function

As described in Section 3.2, our problem is to estimate node positions $X = \{\boldsymbol{x}_i^t\}_{i,t}$ to match measured ranges $\{\delta_{i,j}^t\}$ and node acceleration measurements $\{\boldsymbol{\alpha}_i\}$. Our cost function S penalizes any coordinates that increase the squared error. We divide S into components for each node, $S(X) = \sum_i S_i(X)$, where the local cost function $S_i(X)$ is:

$$S_i(X) = \sum_{t=1}^{T} \left[\sum_{j \in \mathcal{N}_i} w_{i,j}^t \left[\delta_{i,j}^t - d_{i,j}(X) \right]^2 + r_i \left[\boldsymbol{\alpha}_i^t - \boldsymbol{a}_i(X) \right]^2 \right],$$

where the first term represents the error between measured distances $\delta_{i,j}^t$ and the actual distances based on location coordinates $d_{i,j}(X^t)$, which are calculated as,

$$d_{i,j}(X) = \|\boldsymbol{x}_i^t - \boldsymbol{x}_j^t\| = \sqrt{(\boldsymbol{x}_i^t - \boldsymbol{x}_j^t)^T (\boldsymbol{x}_i^t - \boldsymbol{x}_j^t)}. \tag{3.1}$$

Whenever a node's velocity changes, its non-zero acceleration is measured by its accelerometer. We incorporate this extra information in the latter part of the sum, representing the error between the measured acceleration $\boldsymbol{\alpha}_i^t$ and acceleration $\boldsymbol{a}_i^t$ which calculated from the coordinate path travelled as:

$$\boldsymbol{a}_i(X) = \left(\boldsymbol{x}_i^{t+1} - \boldsymbol{x}_i^t \right) - (\boldsymbol{x}_i^t - \boldsymbol{x}_i^{t-1}). \tag{3.2}$$

Our approach finds $\hat{X} = \text{argmin}_X S(X)$, in a distributed manner, to provide location estimates $\{\hat{\boldsymbol{x}}_i^t\}$.

Note that $S_i(X)$ is local to i^{th} node since it only depends on the measurements available at i^{th} node and positions of its neighbour nodes. Minimizing $S_i(X)$ with respect to $\{\boldsymbol{x}_i^t\}_t$ results in new position estimates for node i. Implementing our approach at each node

constructs the backbone of this distributed method. We use majorization at node i to guarantee non-increasing local cost.

Our method is described in Algorithm 1 in [67]. The algorithm is iterative and must be given initial position estimates. Generally, each time the algorithm is run, it is initialized using the coordinates of positions from the previous round's estimates, $\boldsymbol{x}_i^t$ for $i = 0, \ldots, N-1$. The first time a neighbor j appears to node i, it must provide its own locations, $\{\boldsymbol{x}_j^t\}_{t=0,\ldots,N}$. Here, we use classical MDS to generate any coordinates $\{\boldsymbol{x}_j^t\}$ for which there are no prior round estimates.

3.3.2 Regression Based Collision Prediction Algorithm

Using the relative locations estimated from previous stage, we predict locations into the near future. Regression analysis is widely used for prediction and forecasting, as it reveals the causal relationships between a dependent variable and one or a collection of independent variables. We choose quadratic regression since trajectories are quadratic in constant acceleration, and polynomial regression generally works well for non-linear interpolation problems. In our case, we predict the future trajectory of the node, which has a non-linear relationship with time due to the dynamic nature of the motion. The output of the polynomial regression in such a scenario can also be interpreted as higher order kinematics. Given T data points $(t, \boldsymbol{x}_i^t)$, where the independent variable t is a time instance and $\boldsymbol{x}_i^t$ is the corresponding location of node i at times $t \in \{1, \ldots, T\}$, we fit a 2^{nd} degree polynomial, such that the polynomial can capture the non-linear motion of the node, and include the current position, the velocity, and the acceleration in calculating position as a function of time t. Thus, we approximate node i's location for real-valued t,

$$\hat{\boldsymbol{x}}_i(t) = \boldsymbol{p}_2 + \boldsymbol{p}_1 t + \boldsymbol{p}_0 t^2, \tag{3.3}$$

where, $\boldsymbol{p}_0$, $\boldsymbol{p}_1$, and $\boldsymbol{p}_2$ are the polynomial coefficients,which we estimate using the least squares approximation giving the estimated coordinates $\hat{X}$. Using the coefficients, the algorithm extrapolates future relative locations of each node, thereby, giving future inter-object distances of each pair of nodes. We are interested in the *near future*, i.e., the time-frame that is equal to or less than the reaction time of the node, which is application

dependent. If the minimum inter-node distance threshold between two nodes is crossed within this near future, a collision is predicted.

3.4 Experiments

3.4.1 Hardware

We conduct a series of experiments with mobile nodes to predict collisions between any pair of nodes. Each node follows the architecture as described in [52]. Each such node is attached to a iRobot Create that moves the node as programmed. Lastly, a Raspberry Pi3 processor is attached on top of each node, which both lets us program the node movement, and measures the acceleration of each node via a BN0055 IMU sensor [70].

3.4.2 Multi-node Ranging Protocol

Each node measures ranges between itself and all other nodes, and no anchors are present in the system. An efficient way to measure all $\binom{N}{2}$ ranges between the N nodes is to use the efficient multi-node ranging protocol in [1], which requires only N message exchanges per cycle to get all the ranges.

3.4.3 Setup

We set $N = 4$ floor nodes to move as depicted in Figure 3.1 in a 6m $\times$ 6m area. Each mobile node (top right in Figure 3.1) undergoes acceleration as detailed in Table 3.1, constantly between its starting position and its stopping position in Test I and II. Test III has node 3 in motion at constant speed, and due to its motion in a circle, the magnitude of its acceleration is 0.125 m/s^2. We collect UWB ranges between every pair of nodes at a rate of 18 ranges per second. The acceleration of each node is measured via the IMU sensors and collected by their attached Raspberry Pi at a rate of 100 samples per second. We route

the ranges and acceleration data collected by each node to a central processing unit for offline algorithm testing and result generation. Note that this offline implementation is just for convenience during tests; our distributed algorithm can be implemented in firmware at each node, and will be our future work. Each Raspberry Pi is NTP time synchronized, such that the timestamps for ranges and acceleration can be matched to produce 18 range-acceleration pairs per second. Lastly, to record the ground truth coordinates of each node during each experiment, we use a 16-camera OptiTrack motion capture system, which enables millimeter accuracy [57]. The results are explained in Section 3.5.

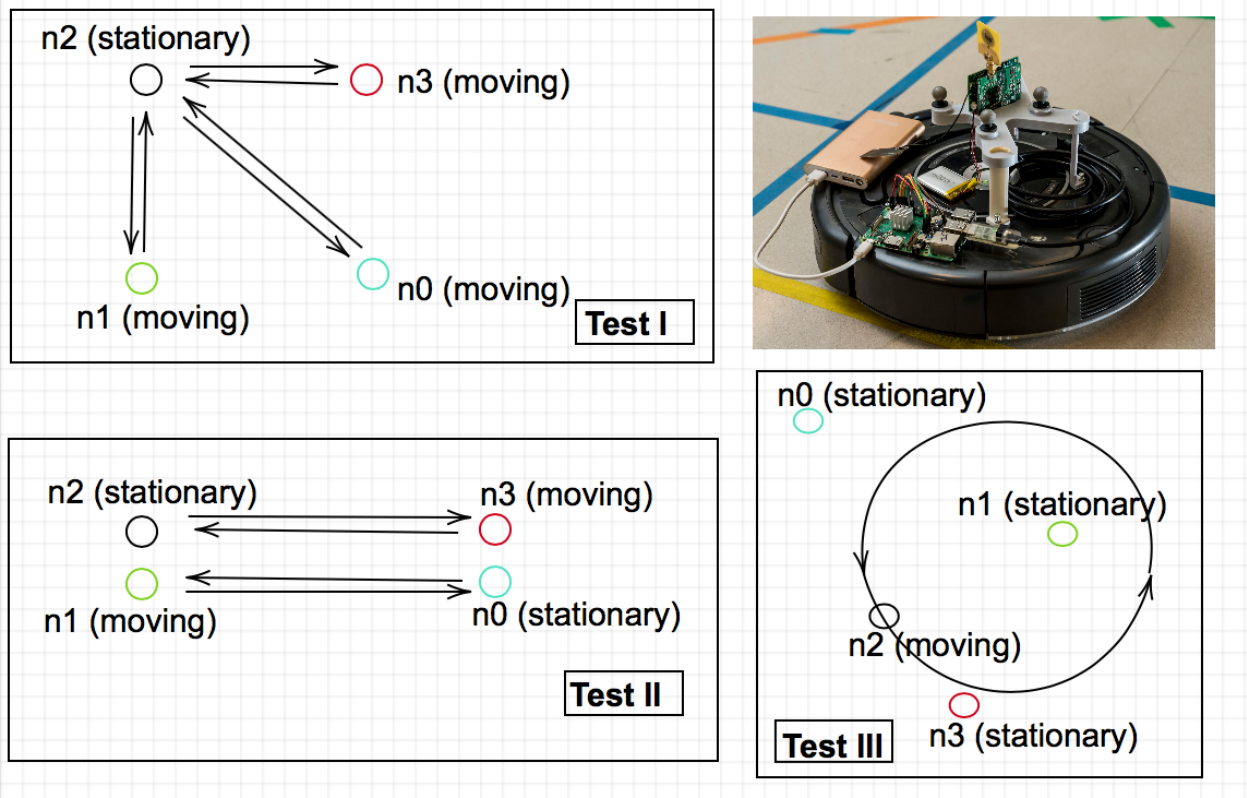


Figure 3.1: Overview of the motions conducted by each node in three tests, and (Top Right) photo of hardware setup.

	Test I	**Test II**	**Test III**
Stationary Nodes	1	2	3
Mobile Nodes	3	2	1
Acceleration (m/s^2)	0.125, 0.09, 0.06	0.125, .06	0.125

Table 3.1: Node Setup in Three Tests

3.5 Results

3.5.1 Location Estimation

We first demonstrate the quality of location estimates generated from the ACED algorithm. For any of the 4 nodes, ACED estimates the trajectory that was followed over time. We use a window of $T = 20$ samples, thus each time the algorithm is run, we estimate $\{\boldsymbol{x}_i^t\}$ for $t = 1, \ldots 20$ and $i = 1, 2, 3, 4$. In a sliding window manner, ACED repeats by dropping the oldest time and adding one new time, and re-running the estimation for the next window of time. As a typical example, we plot the current T location estimates for node 2 as 'boldface $\times$' and the current ground truth locations as 'boldface $\circ$' in Figure 3.2. We also plot the location estimates from another state-of-the-art method, friend-based autonomous collision prediction and tracking (FACT) [1]. FACT assumes a constant velocity, and hence is unable to track the curved trajectory of node 2 from Test III. Furthermore, ACED is capable of predicting future positions 'lightface $\times$' based on (3.3), which are plotted for node 2 in the Figure 3.2 against the ground truth 'lightface $\circ$', where we define 'near future' as within 0.02 sec into the future.

We use the following as our metric of error for the output of one window of any estimator:

$$RMSE = \left[\frac{1}{N} \sum_{i=1}^{N} \left\| x_i^{T/2} - \hat{x}_i^{T/2} \right\|^2 \right]^{1/2}, \tag{3.4}$$

and we report the average RMSE across all sliding windows across the entire test. Figure 3.3 plots this average RMSE of location estimated for all the nodes during each test by three methods, ACED, FACT[1] and modified MDS[60]. From [60], we used its centralized algorithm to find a globally optimum solution for relative position and velocity; however, it is known to perform sub-optimally in noise [60, 1].

Our results show that the FACT method of [1] diverges over time when motion is not linear. As described, each new run of the algorithm uses as initialization the trajectory estimates from the prior window. In Test III with a circular track for node 2, as FACT estimates a linear trajectory, its initialization from the prior window is poor. Over the course of Test III, its estimates at some point are unable to converge to the global optimum, after which it loses track of the coordinates and is unable to recover, leading to a very high

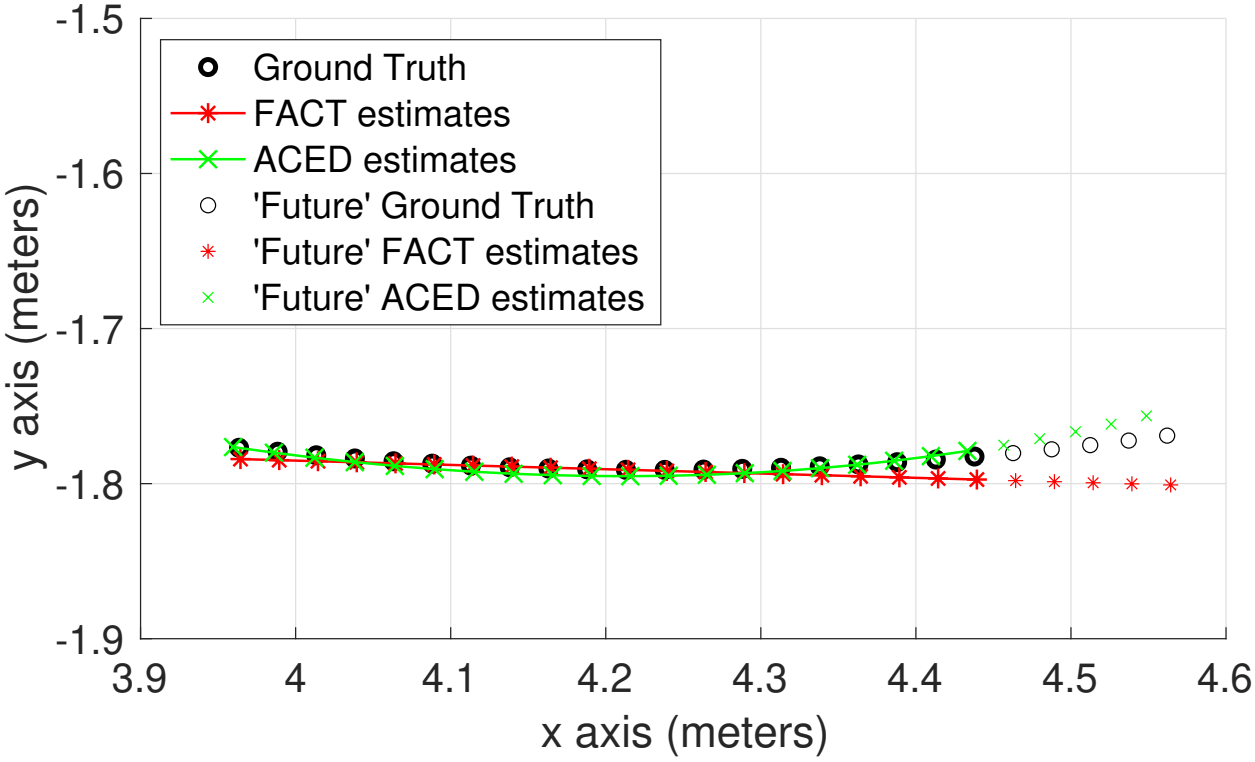


Figure 3.2: Location estimates by ACED and FACT for the node moving with acceleration per window for Test III. ACED's predicted 'future' location estimates, are also plotted against ground truth.

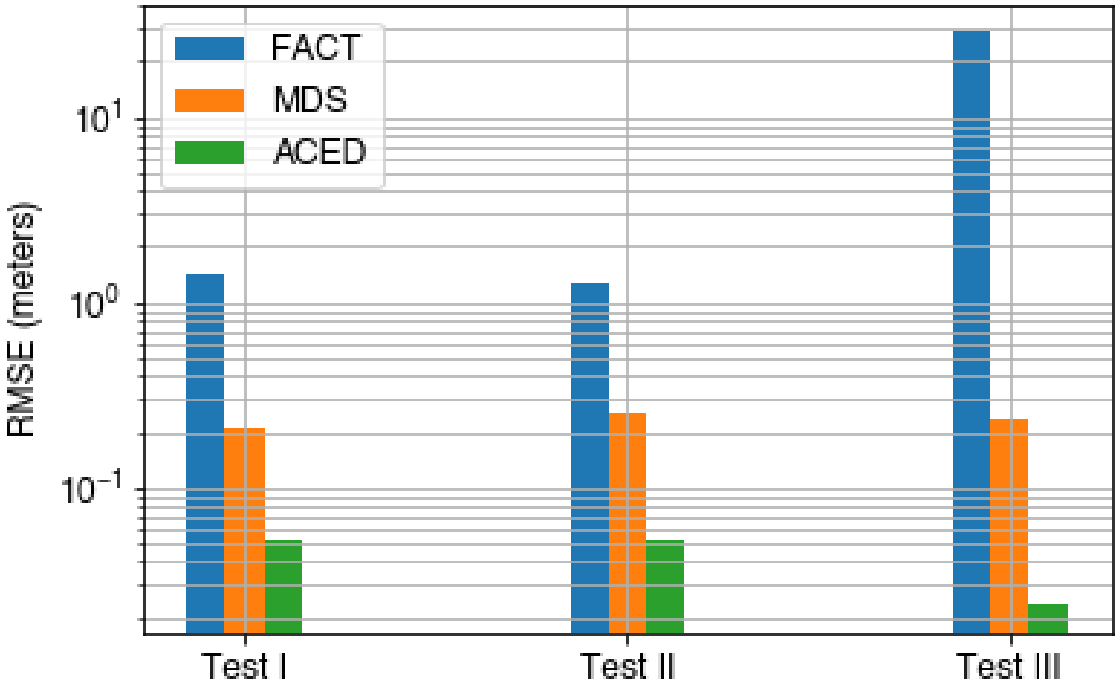


Figure 3.3: Comparison of ACED vs. the modified MDS [60] and FACT [1] for each test, showing RMSE averaged across all nodes.

average RMSE across Test III. ACED provides better tracking in dynamic motion, and doesn't have this convergence problem in our experiments. ACED also compares well to the centralized modified MDS method of [60], demonstrating a lower RMSE by 5-10x.

3.5.2 Interpolated Distance-based Collision Prediction

In order to avoid collision, a node must predict a collision before it happens. In ACED, if the distance $\hat{d}^t_{i,j}$ between nodes i and j in the near future t will fall below a threshold, d_{thd}, this counts as a future collision. Using the relative location estimates from ACED, we extrapolate pairwise distances into the near future. In this experiment, we define the near future as within $\tau = 0.02$ s.

We set the distance threshold d_{thd} to allow a trade-off between false alarms and missed detection(s). Letting r be the radius of one autonomous object, we would set $d_{thd} = 2r + \epsilon_d$ for some $\epsilon_d \geq 0$. By increasing ϵ_d, we would increase the probability of detection of a collision P_D while also increasing the probability of false alarm P_{FA}. A user could set the threshold based on the desired trade-off between the two. Figure 3.4 shows the receiver operating characteristic (ROC) curve, i.e., the relationship between P_D and P_{FA}, compiled with data from across all three tests. ACED is able to provide higher P_D for the same P_{FA} when compared with FACT [1]. Note that even when FACT location estimates diverge, it manages to keep an accurate relative position and velocity for two nodes that are very close, and thus collision predictions are good. However, ACED cuts P_{FA} approximately by a factor of 2 for a constant P_D compared to FACT. Since ACED provides more accurate kinematics whenever nodes are accelerating, it can extrapolate complicated trajectories better, thus providing accurate future inter-node distances and kinematics to predict collisions. The 2^{nd} degree regression coefficients are able to extrapolate the future locations while taking each node's acceleration into account, a trait not achievable by FACT. We also test against the pairwise method of [2], which does not perform nearly as well as FACT or ACED.

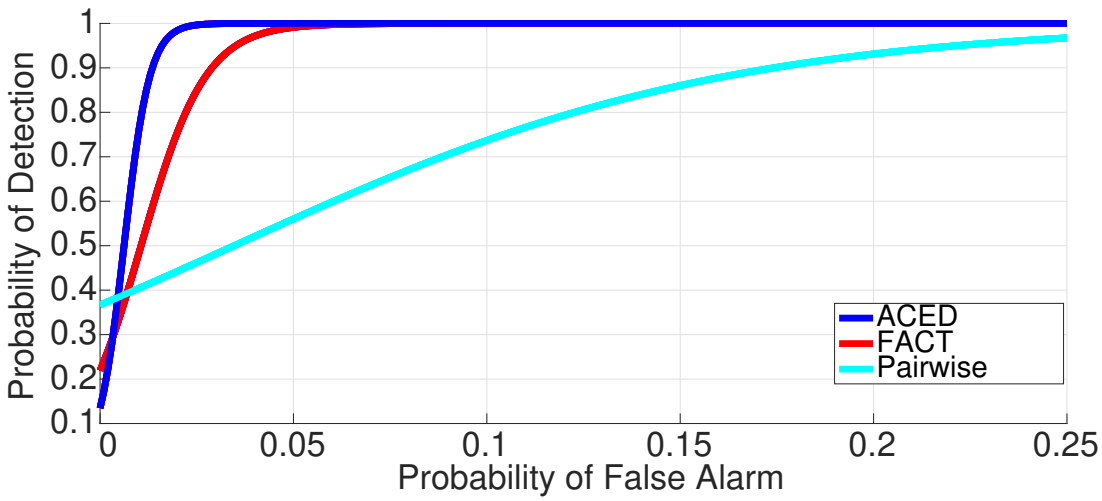


Figure 3.4: ROC plot comparing ACED, FACT [1], and pairwise [2].

3.6 Conclusion

This paper presents ACED, a new approach to estimate trajectories and predict collisions for systems involving mobile devices that are capable of measuring node acceleration and pairwise range measurements.The algorithm does not require a known-location infrastructure or a centralized computation. We test its performance in a network of four prototype mobile nodes mounted on ground robots in three tests. ACED predicts a node's trajectory with an order of magnitude lower RMSE, and collisions with a $> 2x$ lower false alarm probability, than three state-of-the-art infrastructure-free trajectory estimation and collision prediction methods.

Chapter 4

Online learning for dynamic impending collision prediction using FMCW radar

4.1 Introduction

Collision warning systems (CWS) aid in providing safety measures in a variety of applications. In the field of contact sports, physical collisions between players cause 1.6 to 3.8 million concussions or traumatic brain injuries (TBI) annually in the United States, and American football is a most prominent contributor of these TBIs [22]. Such concussions are life-altering and adversely affect players throughout their lives [30]. Traditional protective gear such as helmets can reduce the severity of such concussions, however, they are limited in their utility, as even with the helmets concussions occur. Preventative measures are an important part of protection, therefore, predicting the collisions that are about to occur before they happen could dramatically improve outcomes. One solution is to equip the player with smart wearable device that predicts the impending collisions. A smart helmet could automatically and actively adapt its damping to reduce harm upon impact [9]. Alternatively, it could supply a loud auditory warning to the player that allows them to react. New research suggests that warnings prior to impact allow people to 'prepare themselves, effectively mitigating the consequence of the impact' [53]. Our radar-based collision prediction system has the goal of enabling such active responses from players and their helmets.

Collision prediction systems also find utilization in the field of automation, where frequent collisions encountered by the mobile nodes such as unmanned aerial vehicles (UAVs) and robots cause nuisance and damage. Collision warning systems for vehicles [38], [6], [85] have developed solutions that can be applied to combat the collision problem faced

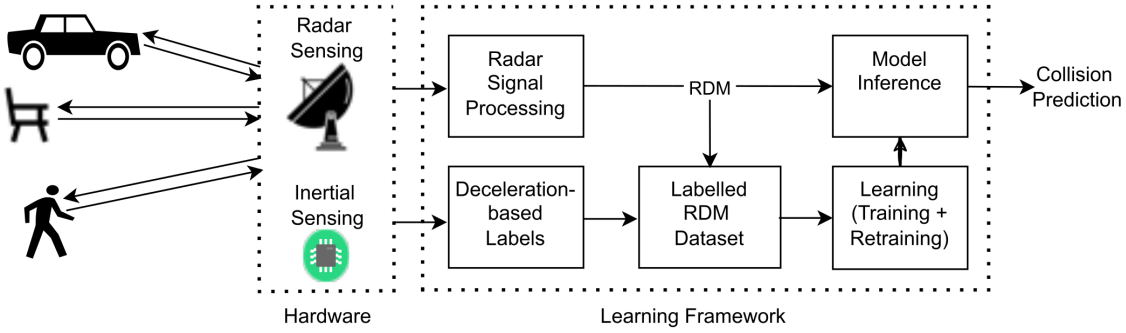


Figure 4.1: Proposed collision prediction system overview: The hardware block present on a mobile node is capable of sensing the environment with the help of a FMCW radar-based sensing and notes its inertial measurements. The learning framework uses measurements to generate radar-Doppler matrices (RDMs) that are fed to a trained model for inferring impending collisions. Simultaneously, the model is periodically re-trained with the latest labelled RDMs to improve results in a changing environment.

in automation. For example, centralized algorithms such as [24], [18] can provide location information to mobile nodes in a network to locate surrounding nodes and predict impending collisions. As an alternative, distributed and cooperative solutions to obtain locations are also well studied [55]. Similarly, mobile nodes can rely on network-provided information to locate themselves and predict any collisions with other nodes [4]. These localization methods are suitable for nodes that are part of a network. However, every object encountered by the mobile node may not be associated with the same network as the mobile node.

Therefore, in order for a mobile node to predict impending collisions with any kind of objects in the environment, it should be able to perform standalone sensing of all objects in its environment without requiring a network of sensors.

It should be noted that collision prediction is fundamentally different than localization. Collision prediction must take into account not only the present positions of objects in the environment, or positions at any single future time, but all of the positions between now and a future time in order to know whether a collision will occur at any point in that time period. As a benefit compared to localization, however, collision prediction uses only the relative kinematics (such as relative range, relative velocity), rather than absolute coordinate information of the objects. We address the complexity of estimating positions

across a time period; and take advantage of the relative nature of collision prediction in the methods developed in this paper.

We consider a mobile node that is envisioned to be equipped with sensing and processing capabilities that allow it to make standalone measurements from the environment and self-sufficient decisions about impending collisions. The sensors are required to be low cost, compact, lightweight, and capable of processing information to predict collisions with a low false positive rate. There are several sensor options available for CWS, each with their advantages and limitations [74]. In the case of sensing within a network of nodes, paired communication with other tagged devices is achieved using ultra-wide band (UWB) [66] or radio frequency identification (RFID) sensors. However, for sensing of the entire environment for possible collision with both tagged and un-tagged objects, standalone sensing is required. LiDARs (light detection and ranging) are expensive sensors that provide rich information about scenes and targets, but can be susceptible to severe weather. Ultrasonic sensors are affordable and compact and are suitable for very short-range detection. Vision-based solutions do not perform well for long distances, in poor light, and in complex, real-world conditions. Radars have been extensively studied in literature, especially for automotive applications [31] and unlike radio frequency tags, in which each device must receive reflections from other tagged devices for pairwise ranging, radars are capable of sensing all objects with electromagnetic properties different from air that are present in the field of view, without requiring additional sensors on the surrounding objects. Moreover, FMCW radars are inexpensive, compact sensors that can measure the relative kinematics (range and range-rate) of the objects in the field-of-view in real-time. Building on this standalone sensing capability of FMCW radars, a performance study of FMCW radars is presented in [15] where the authors analyse the use of an 'impact parameter' in predicting collisions. It must be emphasized that in practice, the accuracy of range and velocity measurements from radars suffers due to system noise and unwanted scattering and reflection, typically referred to as 'clutter', which is a function of the material properties of the other objects in the environment that we are not concerned about the robot colliding with, for example, itself, or anything that would not cause problems if hit. Traditional collision prediction methods require static filtering and pre-processing techniques which may not be effective in the environment in which it is deployed. Traditional methods cannot adapt to the type and style of play of an individual sports player. Learning-based solutions, on the other hand, can offer flexible and

data-dependent prediction capabilities in order to efficiently utilize FMCW radars for dynamically changing environments. In the smart helmet application, they could also learn player-specific mobility patterns that do and do not lead to collisions.

Furthermore, the learning-based solutions need to be robust against degrading performance when there is shift in the input data characteristics, therefore it is necessary that a learning-based model's parameters be continuously adjusted for new or dynamic environments. The machine learning paradigm of online learning is a continuous learning process to learn more efficiently with the data arriving incrementally, to perform the same learning task over time, by the model that is already deployed. When a new labeled data sample arrives, online learning allows the existing model to quickly update its parameters to produce the best model so far. However, incrementally updating with the newest data leads to catastrophic forgetting [56], where learning only a small amount of new information can overwrite established knowledge and cause complete loss of ability to operate on previously learned tasks. Incremental batch learning solves the problem of online catastrophic forgetting by utilizing a series of batches of new labeled samples. After a batch has been received, the model loops over the batch until it is adequately learned, and then the model can be tested on information in that batch and previous batches. However, due to limited resources and requirement for fast on-device learning, only the relevant samples that carry new information must be used for learning incrementally. The field of active learning [64] provides methods for selecting those most relevant labelled samples thereby reducing the size of information that is necessary for decision making. Taking advantage of the on-going research in the field of online incremental learning, we present a functional and practical real-time algorithm that adapt its parameters with the dynamically changing environments.

In this paper, we extend the nascent idea of utilizing FMCW radars for collision prediction in noisy, cluttered, and dynamic environments. Typical CWS methods [14] implement parametric algorithms where collision risk is measured by calculating node specific parameters, such as its range and velocity. However, these parameters are prone to inaccuracies when measured via a radar in cluttered environments. In order to tackle this, we utilize the research done in the field of computer vision and machine learning. As presented in Fig. 4.1, we propose a learning-based solution to demonstrate the effectiveness of using only the range and radial velocity (range-rate) information obtained from RDMs for predicting impending collision with the help of a convolution neural network (CNN).

The proposed deep learning framework extracts adaptable features from the (continuously changing) environment, thereby eliminating the need for static filtering. Moreover, we apply online learning strategies, as well as automated labelling using accelerometer measurements, in order to make the the CNN classifier adaptable and learn from the recent history of the dynamically changing environment.

Challenges

In order to solve the collision prediction problem using FMCW radar and deep learning in a cluttered and dynamic environment, we face the following challenges:

- The presence of clutter in the radar measurements requires extensive filtering to get accurate measurement of the relative range and relative speed for collision prediction.
- Working with static features for machine learning solutions deteriorates the performance of the such solutions operating in new types of environments or dynamically changing environments.
- The dataset obtained for building a collision prediction system suffers from class imbalance due to the fact that real collisions happen only rarely compared to non-collisions.

Contributions

Overall, we make the following contributions.

- We develop a novel learning-based collision prediction method that uses only radar and inertial data to detect an impending collision.
- We remove the need of static filtering by using the unfiltered, raw radar data for collision prediction, relying on deep learning to extract useful features to predicting collisions.

- We alleviate the class imbalance problem by assigning weights to each class as per their respective frequency in the dataset to achieve high classification accuracy for all the classes.
- We provide a retraining framework by using automated labelling and uncertainty sampling for improving prediction performance in dynamically changing environments.
- We collect, experimentally, a large dataset of experiments to mimic real-world collision scenarios, which will be made public. We use the dataset to validate that our proposed method performs well in comparison with other non-learning collision warning methods and traditional machine learning methods.

4.2 Methodology

In this section we formally define the problem statement, the notation for the measurements, the loss function utilized in the learning framework, the learning model used to minimize the loss function, and the retraining procedure in detail.

4.2.1 Problem Statement

We consider a mobile node which has an attached FMCW radar, an inertial sensor to measure node's acceleration, and a processor for computing. .

The mobile node is moving in an environment with several obstacles. Let us consider the scenario where the radar-enabled node is moving towards an obstacle. Let $d(t)$ be the range and $v(t)$ is the relative velocity between the mobile node and the obstacle at time t. During one measurement time period between t_1 and $t_2 > t_1$, duration $T = t_2 - t_1$, the node collects samples from the radar and inertial sensor. At time t_2 it converts them to a range-Doppler matrix, X_{t_2}.

The goal of our system is to raise an alarm at t_2 if at any time $t \in [t_2, t_2 + \delta t]$, that $d(t) < \epsilon$, where $\delta t > 0$ is the time duration into the immediate future for which we must detect a

future collision. Also, $\epsilon > 0$ is a collision proximity range threshold. In other words, if the measurement indicates that the node will collide with the object within the next δt period of time, the alarm should be raised. In order to achieve this, we create system that can learn a function f that maps from the range-Doppler matrix to a binary decision about whether there will be an impending collision.

4.2.2 Learning Framework

For the collision prediction problem, the inputs we use are RDM matrices, named X, that get mapped via the function f to $\hat{y} \in \{0, 1\}$, where class 0 is encoded as 0 for representing 'no impending collision' and class 1 is encoded as 1 representing 'impending collision'. This makes collision prediction a binary classification problem. Supervised learning-based approaches require a training set of 'measurement-label' pairs $[(X_1, y_1), ...(X_n, y_n)]$ where n is the total number of pairs in one training set, X_j is matrix j and y_j is its collision label, such that $j \in \{1, ..., n\}$.

Loss Function

In order to learn the optimal mapping f between RDM and $\hat{y}$, a classifier for two classes needs to minimize a loss function L for the entire training set of size n, given by

$$L = -\frac{1}{n}\sum_{j=1}^{n}[y_j \log(p_j) + (1 - y_j)\log(1 - p_j)], \tag{4.1}$$

where p_j is the probability of the j^{th} data point belonging to a class as predicted by the mapping. Taking an average over the entire dataset of size n, we get cross entropy loss L, a standard loss function for classification problems.

CNN-based Classifiers

CNNs have become widely popular for image-based machine learning tasks. Compared to the standard MLP architectures, a CNN architecture uses far fewer parameters and can

have much deeper architectures which can equip us to solve more complex problems. There are several CNN architectures available with varying degrees of computational complexity and performance.

For our learning-based solution, we test two CNN architectures: (a) ResNet architecture, since it has shown the top-5 error rate on ImageNet dataset [35] and (b) MobileNet-V3 architecture, since it has been optimized for low-resource devices and low latency applications [36].

4.2.3 Radar Signal Processing

The FMCW radar transmits sequences of a linear frequency modulated (LFM) signal, also called a 'chirp' signal having N samples per chirp, which increases its frequency linearly with time with a slope s within a bandwidth of B Hz. Once the radar receives the reflected signals that bounce back from the target, they are mixed with the transmitted signals to obtain a beat signal. The beat signal is characterized by the beat frequency, f_b, which is equal to $2ds/c$, where d is the range of the object from the radar and c is the speed of light. This frequency is used to infer the range of the target from the radar sensor. A fast Fourier transform (FFT), called the 'range-FFT', is performed on the beat signal to convert it into the frequency domain, thereby obtaining the beat frequencies representing one or multiple objects at various ranges, with a range resolution of $c/2B$ and up to the maximum range of $Nc/2B$. A second FFT, called the 'velocity-FFT', is then performed across a certain M number of chirps that make one RDM, to estimate the relative radial velocity. It is left to the discretion of the designer to define how many of the chirps are going to be processed together into an RDM (due to processor limitations and/or resolution requirements) [28]. Our proposed method implements this 2-D FFT stage and generate a $N \times M$ matrix, which we call the radar-Doppler matrix (RDM), that indicates the amplitude of scattering from each possible relative range and velocity across the possible range of measured range and velocities.

It should be noted that reflection and scattering of the radar signals by objects are a function of the intrinsic properties of the material of the objects, such as dielectric permittivity, magnetic permeability, and electrical conductivity [69]. These properties have been shown to be effective in object distinction and detection [82]. Additionally, the extrinsic

characteristic such as the absolute size of the objects and the size of the objects relative to the wavelength of the radar signals play a crucial role in the amplitude of scattering received in the radar reflections. Therefore, different type of objects produce different amplitude of scattering within RDMs.

4.2.4 Label Creation

Our proposed system automatically generates (with a sub-second delay) labels for the collected radar data using inertial sensing. This labelled training data can be used to automatically retrain the model in order to improve results for the particular environment and user characteristics that are observed during operations.

As a collision between the moving node and an obstacle occurs, the moving node experiences a sudden change in velocity. This change in velocity can be clearly observed as a change in the measured acceleration, for example, as a sharp drop or negative peak in mobile node's IMU measurements, as seen in Fig. 4.2a. During normal operation we obtain these 'moments of collision' by the finding peaks in a node's measured acceleration data.

A collision prediction system should alert the nodes about impending collisions before they happen such the colliding nodes have time to act preemptively to avoid the collision. Considering an alert can be raised instantaneously by a collision prediction system without any mechanical or processing delay, a node receiving this alert, however, will take time equal to its reaction time for any responsive action. To have at least one alert sent by the system to the node before any impending collision, the immediate future δt to check for impending collisions must be at least equal to the reaction time of the node. Thus, if a collision occurs at δt into the future, it can be avoided or ameliorated by issuing one alert at a time that is δt ms before the collision. Depending on the reaction time of the node and the number as well as the frequency of alerts needed for the node (robots or human) in any specific application (vehicular or sports-related), the threshold for δt ms can be adjusted. Based on this approach, for labelling purposes, all the measurements (the RDMs in our case), that happened within δt ms from each moment of collision as measured by IMU are labelled as 'Impending Collision' (shown as magenta colored square markers on trajectory as shown in Fig. 4.2b). The rest of the RDMs are labelled as 'No

Impending Collision' (shown as blue colored circle markers as shown in Fig. 4.2b). The labelled images are then used to train our CNN based model at the subsequent training time to build the next model that provides inference about impending collisions when encountering new, unseen test RDM samples.

4.2.5 Class Imbalance Problem

During system operation, the radar is continuously sensing the environment. Even in the environments that are densely populated with obstacles, the events during which a collision is imminent is considerably lower in frequency than the frequency of events in which the node is not in danger of an imminent collision. With the labelling scheme for the generated RDMs described above, the number of RDMs labelled as 'impending collision' are order of magnitudes lower than the number of RDMs labelled as 'no impending collision', which leads to an imbalanced dataset.

To deal with the problem of imbalanced classes during training, methods such as over-sampling of the less frequent class, under-sampling of the more frequent class, or the use of tree-based algorithms with boosting are suggested [39]. However, increasing the size of the training set without gaining any additional information, discarding of relevant information, and over-fitting are their drawbacks, respectively. Instead, we use the method proposed in [44], in which each class is assigned class weights that is inversely proportional to their respective frequencies, such that

$$w_j = \frac{|C_1| + |C_0|}{2|C_j|}, \tag{4.2}$$

where $|C_j|$ is the number of samples in class $j \in \{0, 1\}$. Assigning a small weight to the cost function for the more frequent class during training results in a smaller error value, and thus, small update to the model coefficients for the more frequent class. A large weight applied to the cost function for the less frequent class will result in a larger error calculation, and in turn, large update to the model coefficients for the less frequent class. This way, we can shift the imbalance of the model so that it could reduce the errors of the less frequent class.

An additional problem encountered for imbalanced datasets is that using the accuracy of the model to assess its performance becomes a less reliable metric because an acceptable accuracy is possible to obtain even if the model has excellent classification performance for more frequent class and poor performance on the other, less frequent class [46]. Thus, instead of using accuracy to pick the best model, we use the $F1$ score as a metric that describes the performance of the classifier on both the classes. $F1$ score is the harmonic mean of the positive predictive value (P, also called precision) and true positive rate (R, also called recall or probability of detection(P_D)) given by

$$F1 = \frac{2PR}{P+R}, \tag{4.3}$$

where P is the fraction of correctly classified positive instances ('impending collision' in our case) among the overall instances that are classified as positive, while R is the fraction of correctly classified positive instances among the overall instances that are truly positive. A high $F1$ score indicates low misclassifications in both of the classes. Therefore, during training epochs, we select the model which has the highest $F1$ score.

4.2.6 Online Learning Framework

Real-world implementation of any collision system experiences a continuous influx of new sequential data in real-time. A static learning-based solution leads to decreased prediction performance if the new data is from different distributions, as a result of, for example, changing user behaviour or environmental condition, than the previous data on which the model was trained. This phenomena is called data drift. It is important to periodically retrain the model to avoid data drift in real time applications. Model retraining is defined as re-running the process that generated the previously selected model, however, on a new training dataset that has experienced data drift. In order to incorporate continuous model retraining, we explore the branch of online learning and suggest the following retraining strategy:

- Continuously collect system's IMU-labelled RDMs from recent history to form a retraining dataset.

- Based on the system requirement for the retraining latency, select a fraction of the collected dataset using the uncertainty-sampling technique [48].
- Retrain the default or current model using this newly sampled subset of data.
- Repeat the above steps for the newest recent history.

4.3 Experiments

4.3.1 Objects and Environment

In order to obtain a large quantity of data in which our collision prediction and warning device experiences actual collisions with obstructions, without having to subject a person to carry the device and experience the same collisions, in this paper we implement the following setup. We use a robotic motion platform as our moving node for our experiments that is comprised of an iRobot Roomba, which is controlled by attached Raspberry Pi-3 module as shown in Fig. 4.3b. The node moves in a laboratory environment as shown in Fig. 4.3a which has other stationary obstacles. The OptiTrack motion capture system [58] tracks and records the ground truth position co-ordinates of all the objects that are tagged with reflective markers. An 3 m $\times$3 m area is barricaded with PVC pipes in order to confine the moving node and maximize its number of collisions with obstacles within the observed area. Within this obstacle-rich area, randomly placed obstacles are present that are filled with a variety of materials. In order to collect a dataset with a variety of radar reflections for our collision prediction method, we choose three different kinds of materials (gravel, soil, and water), as shown in Fig. 4.3c, 4.3d, 4.3e, and 4.3f, to simulate the items that are commonly encountered by a moving objects (vehicles or humans) in the real-world. Cylindrical shaped objects are 0.5m in height and 0.3m in diameter, where as rectangular shaped objects are $0.5 \times 0.2 \times 0.3$m in width.

4.3.2 Sensing hardware

We use AncorTek's SDR-KIT 2400AD2 [73], a low-power and compact software-defined radar for FMCW-based sensing, sitting at the top deck of the prototype. The center frequency of transmitted signal is adjustable within the 24-26 GHz frequency band. AncorTek provides drivers only for Windows, and thus, we use a Windows laptop to receive and store the complex in-phase and quadrature (I/Q) components of the received FMCW radar signal. For the radar's settings, we choose a bandwidth of 2 GHz at a center frequency 25 GHz during all our experiments. Each chirp is of 1 ms duration with $N = 128$ samples per chirp. With these settings, the range resolution and the maximum range that can be measured are 0.075 m and 4.8 m, respectively. Also, the maximum radial velocity that can be measured is 3 m/s. The radial velocity resolution is dependent on the number of chirps processed together at once for one RDM [28]. Lastly, the moving node also has a BNO055 IMU sensor [70] that collects the acceleration data during our experiments at the rate of 60 Hz. It should be noted that the size of the radar sensor used for our experiments is $79 \times 56 \times 13$ mm, which is comparable to the size of a commercially available collision warning radar systems. However, in applications such as smart wearables and smart devices, the size of sensing hardware is encouraged to be miniaturized, for example, as accomplished through Google Soli's 12×12 mm radar chip [49].

4.3.3 Motion and Collisions

We conduct a series of experiments with one mobile node moving in a cluttered environment with eight obstacles that are stationary. The maximum velocity that can be reached by the moving node is 0.5 m/s. The moving node is made to travel in two kinds of trajectories, straight and curved. For the straight trajectory, both the wheels of the moving node are programmed to maintain one constant speed (0.5 m/s), and thus, the node moves in a straight line motion. For the curved trajectory, one wheel of the moving node rotates slower than the other, creating a curvature in the trajectory, and mimicking a curved line motion. Several collisions happen between the mobile node and the obstacles during both styles of these trajectories as given in Table 4.1. When a collision happens, the moving node comes to a complete stop, takes a turn by a random angle, and then starts moving again at the same speed and in the same style of programmed trajectory as before the

collision. Lastly, we do not consider the PVC pipes as 'obstacles' since they are used only to keep the robots inside the experimental area, and thus, all the collisions and the corresponding RDMs between the moving node and the PVC pipes are not included in the final dataset.

Exp.#	Motion type	Number of Collisions	Number of Valid RDMs
1	Straight (S1)	35	5781
2	Straight (S2)	41	6693
3	Straight (S3)	36	5810
4	Straight (S4)	30	4050
5	Straight (S5)	31	4103
6	Curved (C1)	36	6536
7	Curved (C2)	45	7213
8	Curved (C3)	53	6723
9	Curved (C4)	40	6692
10	Curved (C5)	31	6790

Table 4.1: Details of the experiments conducted

4.3.4 Data Characteristics

We run 10 experiments, each one either with curved and straight trajectories and each experiment is of 10 minutes duration. We process $M = 200$ chirps together to make one RDM, thus for our experiments, one RDM comprises of 200 ms duration of complex-valued radar samples. Using a sliding window of 50 ms, one 10-minute experiment generates 1.2×10^4 RDMs. We drop all the collisions with PVC pipes and their corresponding RDMs, as described above, thus obtaining 6×10^3 RDMs per experiment on average. On average, 32 collisions are encountered during one experiment. With the labeling scheme described in Section 4.2.4 and taking $\delta t = 400$ ms, we obtain on average 340 RDMs per experiment that can be labelled as 'impending collision', while rest of the RDMs are labelled as 'no impending collision'. Since only 3% of the RDMs can be labelled as 'impending collision', this dataset is highly imbalanced. As explained in Section 4.3.4, we deal with this issue with the help of Equation 4.2 by assigning weights to each class, as given in Table 4.2.

class j	Number of samples (n_{C_j})	class weight w_j
collision	340	8.5015
non-Collision	5441	0.5312

Table 4.2: Weights for Imbalanced classes in Experiment S1

4.3.5 Radar sensing for traditional CWS

Environments where collisions are likely to happen are characterized by compact spaces with numerous and a variety of objects. Radar systems operating in such environments suffer from additional reflection and scattering from objects which do not cause collisions. For example, scattering from the ground contributes to reflections received at the radar, but in many applications, the node is not going to collide with the ground. Another source of unwanted reflections is the physical object to which the radar is attached, which, for example, in the smart helmet case includes the person themselves.

Overall, we refer 'clutter' as the received radar reflection that are due to any objects within the environment with which the moving node can not possibly collide. Typically these reflections make the largest contribution to the reflections received by the radar and hence, the visibility of the relevant targets in the radar images with which collisions risk must be addressed, get suppressed. Therefore, traditional radar-based CWS systems need to filter out unwanted reflections due to clutter before obtaining range and velocity information.

Classical principal component analysis (PCA) has been extensively used in image and video processing applications as a statistical tool to seek the best low-dimensional approximation of the high-dimensional data. We apply an advanced version of PCA as a model-based filtering method in order to remove reflections that are due to the clutter and extract the reflections that are due to the object from the RDMs.

The data can be constituted as a superimposition of two components: 1) L, which is the low-rank matrix and 2) S, which is a matrix that can be sparse or not. This decomposition can be obtained by robust principal component analysis (RPCA) solved via principal component pursuit (PCP). Due to the correlation between RDMs, the background and clutter are modeled by a low-rank subspace that can gradually change over time, while the objects that are in relative motion with respect to the radar constitute the correlated sparse outliers represented in the sparse matrix.

We demonstrate with the help of Fig. 4.4, the effect of applying RPCA-PCP to RDM images for the three RDMs for various scenarios: (a) radar facing no object in its field of view, (b) radar facing one stationary object in its field of view, and (c) radar facing two moving objects in its field of view. In the left column of Fig. 4.4, the bright vertical reflection at zero velocity is due to clutter. This clutter reflection is removed with the help of RPCA-PCP reconstruction of the signal, thereafter showing only the target in the RDM, as shown in the right column of Fig. 4.4. The post-PCA, filtered RDMs are then processed by the range-FFT and velocity-FFT as explained in Section 4.2.3 to obtain range and radial velocity of the target. It should be mentioned that the amplitude that represent the target gets reduced after the PCA-based clutter removal step. It should also be noted that this clutter removal step is a requirement only for the baseline method. Our proposed CNN-based solution does not require this additional step of clutter removal since it uses the unfiltered RDMs as input for the training as well as for the testing stages.

Baseline Collision Prediction Method

In parametric models for CWS, the time-to-collision (TTC) method is one of the most common methods [34]. For our experiments, we implement the TTC-based baseline method which predicts an impending collision if the measured TTC obtained for every RDM is less than a threshold. We use Equation 4.4 to calculate one TTC value every RDM, which is given by

$$TTC = \frac{d_r}{r_r}, \tag{4.4}$$

where d_r is the range and v_r is the relative range rate between the moving node and the nearby obstacle with which a collision is about to happen and hence, for which an alert has to be raised. An impending collision is predicted every RDM for which the respective TTC is less than a threshold. Using this method for our experiments, we can generate a receiver operating curve (ROC) by computing the probability of false alarm (P_{FA}) and the probability of detection (P_D) for various different threshold values. Note that, if case of two objects, the collision with the closer objects is considered more imminent, therefore, the range for the closer object is selected for the TTC calculations. It should also be noted that the objects are in relative motion with radar, therefore, the calculated range and velocities are relative.

4.4 Analysis

4.4.1 Training Performance of Proposed Method

Our goal is to train a classifier which takes RDMs along with their corresponding labels as input for training and predict the label for unseen RDMs. The cross entropy function specified in Section 4.2.2 is optimized in this process of training the CNN. For training the ResNet-18 CNN as the classifier, Adam [43] is used as the optimizer to update the model's parameters. We also train the MobileNetV3-Small model on the same training dataset and use the RMSProp optimizer [71]. We use D as our training dataset that has all the RDMs from 4 experiments in total: Exp. S1, Exp. S2, Exp. C1, and Exp. C2. The learning rate for the both the models is set to be 5×10^{-4} and both of the models converge in 30 epochs.

We can demonstrate successful training of the ResNet-18 with the help of t-distributed stochastic neighbor embedding (t-SNE) plots of the feature space of the RDM dataset. Fig. 4.5 presents a 2-dimensional projection of the 512-dimensional features space of each RDM from dataset D. In Fig. 4.5a the feature space is from an untrained ResNet-18 model. After successful training the ResNet-18 model as presented in Fig. 4.5b, we can see that the model is able to linearly separate and cluster the collision and non-collision RDMs in the entire combined dataset.

4.4.2 Inference Performance of Proposed Method

For the inference performance of the trained model on any unseen data, we first provide a visual intuition into the collision prediction system with the help of Fig. 4.6.

In the two sub-figures, the surroundings of two different obstacles named 'Object-2' in Fig. 4.6a and 'Object-3' in Fig. 4.6b are shown, along with the straight and curved trajectories traversed by the moving node in the unseen data from Exp. S3 and Exp. C3, respectively. The blue arrow shows the direction of motion which leads to collision incidents between the stationary obstacles and the moving node. The location of the moving node at the time one RDM gets measured is shown by one dot ($\cdot$). During the inference stage,

an RDM is fed to the trained model and a prediction about the 'impending collision' is made. The steps for which 'no impending collision' is predicted are as represented by blue (·) dots, and the steps for which 'impending collision' is predicted are represented by orange (×) cross. The goal of Fig. 4.6b is to provide a visual representation of the inference performance made by our trained model, which includes correct and incorrect classifications for both of the classes.

We combine the datasets from Exp. S3 to S5 and Exp. C3 to C5 to use as the test dataset T, that has the RDMs that are not seen by the trained models. For this test dataset T, predictions made by the two trained models (ResNet-18 and MobileNetV3) are collected and compared with the ground truth for collisions according to the IMU's deceleration-based labels. We also investigate the performance of the traditional supervised learning methods such logistic regression (LR), k-nearest neighbors (k-NN), and support vector machine (SVM) using the same datasets D and T for training and testing, respectively. Fig. 4.7 shows the receiver operating characteristic (ROC) plot for the all the five investigated learning-based classification models as well as the baseline model. The y-axis is probability of detection P_D (or recall R as described in Section 4.3.4) which measures the fraction of RDMs that are inferred as 'collision' over all the RDMs that were truly labelled as 'collision' during the δt window from moments of collisions, as was done in the labelling scheme of Section 4.2.4. The x-axis represents the probability of false alarm, P_{FA}, which measures the fraction of RDMs inferred as 'collision' over all the RDMs that were truly labelled as 'non-collision'. The ROC plot shows that ResNet-18 has the highest area under the curve (AUC) with 0.98, outperforming all the other classification methods and thereby, being our investigation's suggested classifier for the collision prediction problem. The reported F1-Score is 0.91. The MobileNetV3 gives the second best AUC (0.88). The traditional supervised learning methods have AUC of 0.78 for SVM, 0.78 for kNN, and 0.74 for LR, while the baseline method of combining PCA-based filtering with the TTC model has an AUC of 0.6.

We further investigate the nature of predictions made by ResNet-18 on the test dataset T, by exploring the prediction accuracy as a function of the time to collision. As seen in Fig. 4.8, all the false alarms raised by the ResNet-18 classifier are more frequent near the time of the actual collision. This indicates that most of the false alarms (false positives) are close to the time of the threshold of δt. That is to say that the false alarms are less likely to occur when the moving node is further away from objects on the trajectory. Additionally,

it is reported that the probability of detection is the nearly consistent when the time of collision is less than the threshold of δt, indicating that the classifier is able to detect the impending collisions at the specific threshold of δt as designed for our experiments.

Lastly, we also report the effect of the materials of the obstacles on ResNet-18 model's prediction capabilities. The radar reflections vary with the material of the obstacle but our collision prediction model performed with a P_D of 0.95 or higher, regardless of the material composition of the obstacle as shown in Fig. 4.9. For the test datset T, we conduct an one-way ANOVA test. With the null hypothesis being that the materials of the object do not affect the probability of detection of our model and the significance level or Type-I error of 0.01, we report a p-value of 0.0254. Thereby, we accept the null hypothesis, indicating that our model's performance does not vary with the material of the encountered obstacles.

4.4.3 Online learning Performance

Following the retraining strategy in Section 4.2.6, the ResNet-18 classifier is investigated in its retraining performance on the previously unseen data. First, a default model is obtained by training a ResNet-18 classifier on one of the straight-trajectory datasets Exp. S1. In order to emulate data drift, a new, unseen dataset from one of the, curved-trajectory experiments Exp. C1 is used to obtain the performance of the default model. As presented in Fig. 4.10, the default model's classification performance on the new, unseen dataset gives an AUC value of 0.69. Next, this curved trajectory dataset is considered as the recent history and 80 : 20 train-test split of this recent history is then used as our training set to retrain the default model. The retrained model generates an AUC of 0.95 on the test set if the full train set is used. It should be noted that the bigger the size of a dataset used in learning, the longer a machine learning model takes to iterate over the samples in the dataset. We apply the uncertainty-sampling technique for selecting samples from the training set for retraining. We present results for various sample sizes selected for retraining, for example, using top 10% of samples from the full training set that showed highest uncertainty to form as a subset of training set for retraining generates an AUC of 0.83. Similarly, using top 20% and 40% of samples with highest uncertainty generates AUC of 0.884 and 0.885, respectively. We report that the AUCs obtained by retraining on the data subsets of various sizes of the recent history are higher than the AUC from the

default model that was not retrained. Furthermore, using uncertainty-sampling enables low latency and data requirements for retraining and provides a better performance than a static, default model.

Online learning with traditional machine learning models

We also compare the performance of our method with the traditional machine learning methods that incrementally update their parameters as new data is collected. Working in mini-batches of the new data, the model's trained parameters from the previous learning are not cleared, but are updated with respect to the new data provided, which is conceptually equivalent to training a model from scratch on a combined dataset. For our binary classification problem statement, we use two incremental learning estimators, a linear support vector machine and logistic regression. Additionally, in order to build a similar architecture to our method for results comparison, we apply the uncertainty sampling principle on these estimators as well. Each of these estimators are given a varying percentage of most uncertain samples of the new data and their model parameters are updated. The performance of our method in comparison with incrementally trained SVM and LR are also presented in Fig. 4.10. We report that the performance converges at 0.5 of the most uncertain samples from recent history data for incrementally retraining the models. It which shows AUC scores are converge to their optimal after that using only up to half the size of new dataset as training, which will help in reducing the computational load and training time for the models.

4.4.4 Practical Architecture for retraining

We compare the two CNN architectures' inference latencies for one RDM on Intel-Xeon E52666 (CPU) in Table 4.3. It can be seen that MobileNetV3 achieves a 16 times reduction in the model latency compared to the widely popular ResNet-50 and 6 times reduction compared to our work's proposed model of ResNet-18. The CNN architectures also differ in their number of trainable parameters. MobileNet architecture has 10 times lower number of parameters to train in comparison to the ResNet architecture. Therefore, we suggest MobileNetV3 for solutions where swifter predictions are required to be made while having better performance than any traditional supervised learning methods.

Model	Inference time (sec)	# Parameters ($\times 10^6$)
MobileNetV3-Small	0.0165	2.54
MobileNetV2	0.0608	3.50
ResNet50	0.2545	25.56
ResNet18	0.1032	11.69

Table 4.3: A comparison of inference latencies of the various CNN architectures in processing one RDM for real-time applications.

4.5 Prior Work

4.5.1 Smart Helmets and Collision Prediction

Two of the most relevant fields that deploy preventative measures similar to our problem statement of predicting and avoiding collisions are of smart helmets and vehicular CWS. Smart helmets have gained substantial interest from the research community in the past decade. The goal of smart helmets can be diverse, most commonly used for collision prevention, by either deactivating the paired vehicle if helmets are not in the vicinity [61] or by deciding that a collision has occurred and call for help after the collision event [54]. They do not actively predict impending collisions before they happen. Therefore, there is scope for incorporating the smart helmets with advanced sensors and algorithms to instead predict the collisions before they happen, which is crucial in altering the wearer of these helmets to prepare themselves upon receiving an alert according to the prediction. Efforts for a camera-based collision prediction algorithm is presented in [10], where future positions are 'extruded' and the decision about a future collision is made. Another camera-based helmet for avoiding rear end collisions in real-time is presented in [63]. Most of the work in smart helmets utilize cameras or a combination of vibration sensors. To best of our knowledge, radar-based smart helmets for predicting collisions before they happen is novel.

4.5.2 Vehicular CWS and Radars

Vehicular CWS, on the other hand, are well researched and are adapted as a necessity for the automotive industry. The choice of sensor in the vehicular CWS is most commonly

a combination of camera and radar sensors. Radars as sensors although have been extensively studied in literature, they are not used in CWS as the sole sensors because of their degraded performance due to clutter and noise. In order to minimize the clutter reflections received and noise captured in these radar sensor, filtering techniques such as [8], are actively being researched. However, these filtering methods are not dynamic and need to be updated frequently whenever the environment in which the radars operate change. Instead, deep learning can be utilized to work with measurements from radars that operating in cluttered and dynamic environments. Numerous works using deep learning on radar measurements have resulted in successful learning-based solutions in the field of motion and object detection, classification [42], localization and tracking [84], [50], [5], [51], and human health and activity monitoring [20], [59].

4.5.3 Vehicular CWS and Deep Learning

Although the progress in deep learning have found much application in radar-based sensing, the learning-based algorithms for CWS are still being developed. In [68] and [26], authors present a non-learning method in the former and a CNN-based classifier in the latter to use a combination of camera and millimeter-wave radar and predict collisions. Instead of using sensors, another option is to provide input information is vehicle-to-vehicle (V2V) or vehicle-to-infrastructure (V2I) communication streams for obtaining range and velocity information for collision predictions. Based on V2I communication, in [47] a collision warning algorithm using multi-layer perceptron (MLP) neural network called MCWA is presented and reported to have higher performance compared to non-learning methods. A more advanced architecture using VGG-CNN, called RCPM, for a real-time collision detection using trajectory information from videos is presented in [77]. These attempts pave way for exploring faster machine learning architectures for making swifter predictions for safety critical applications such as collision warning. In our work therefore, we propose a ResNet-18 based collision prediction method that only uses purely independent and ego-centric measurements obtained from inexpensive and lightweight FMCW radar sensors. Thus, our method is an infrastructure-free, self-sufficient solution that operates in cluttered environment and is faster and more accurate than traditional parametric and other learning-based methods.

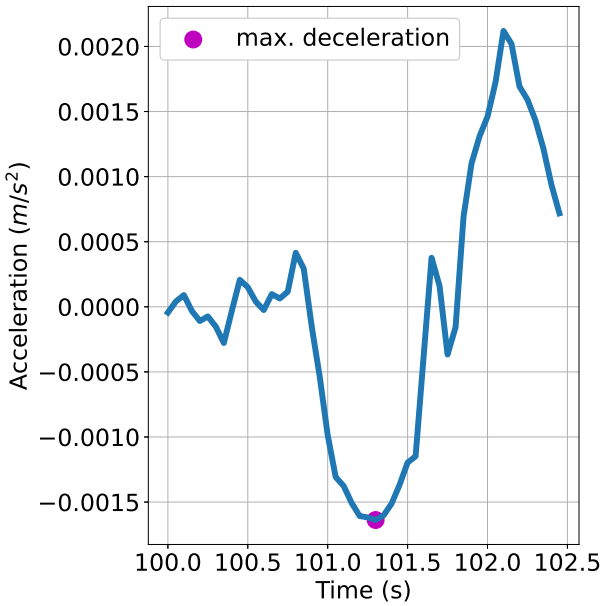
4.5.4 Online Learning and Radars

Finally, the learning-based collision prediction models should additionally use self labelled measurements from recent history and continue to learn. Therefore, a related field to explore is online learning which has seen significant research in building algorithms that continuously and efficiently learn new information, while retaining previously learned information. Most recently, rehearsal-based approaches for incremental learning have been implemented where a model maintains a subset of previous examples that are mixed with new samples to update the model [41], [62], [32]. More specifically for radars, in order to operate in changing environmental conditions, cognitive or fully adaptive radars (FAR) can provide a feedback and optimization mechanism for improved system performance through the subsequent measurements [13], [33]. Another approach is of drift detection with incremental learning, where modification to the learning model occurs only when a change is detected in the environment distribution from which data are drawn, as presented in [3]. In [83], a method is presented that provides extension to its underlying model by adjusting its hidden layer and retraining on selected incorrect samples while retaining previously learned information. We deploy batch learning as the framework for our model to learn continuously with the changing environment.

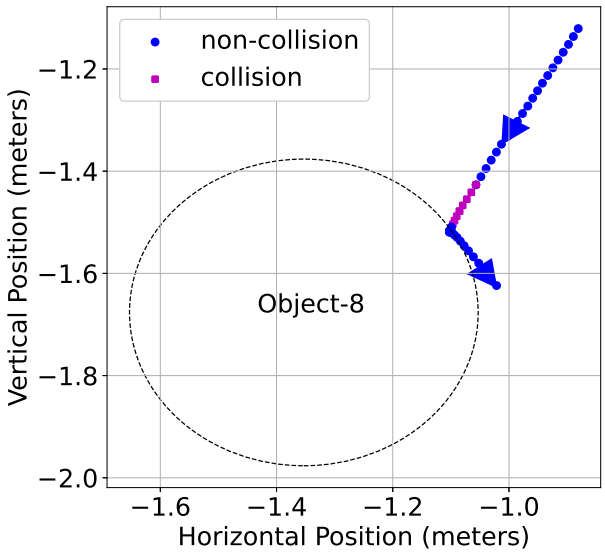
4.6 Conclusions

In this paper, we present a deep learning-based methodology for predicting impending collisions using radar and inertial sensing. Approaching the problem as a binary classification problem, we study the CNN architectures of ResNet and MobileNet to optimize the cross-entropy loss function using labelled range-Doppler matrices as input and collision prediction labels for impending collisions as the outcome. We report ResNet-18 provided an F1 score of 0.91 and outperformed traditional supervised learning methods. We also presented a retraining framework for incorporating online learning capabilities to the ResNet-18 architecture in order to make it adaptable to the changes in the input data distribution due to the dynamic environmental conditions such as changes in the style of motion and object materials. Lastly, the system is designed to automate the labelling process by using the commercially available inertial sensor to provide real-time

acceleration measurements which are used in labeling of the present radar measurements for the continuous, online learning scheme.



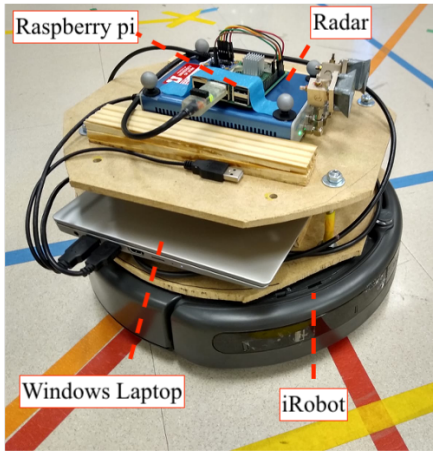
(a) Selecting moment of collision using the maximum deceleration in moving node's IMU data



(b) Using moment of collision to label steps on the path as impending collision

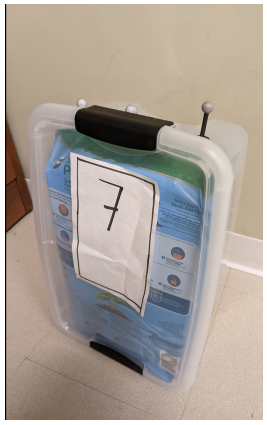
Figure 4.2: Demonstration of the label creation scheme using moving node's deceleration measurement from its IMU

(a) The lab setup with obstacles, moving node, and PVC pipes as boundary.



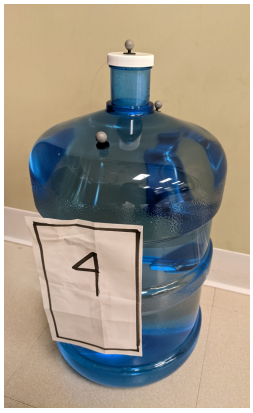
(b) The moving node with a FMCW radar and a Windows laptop to run the radar GUI, attached with Raspberry-Pi to control the trajectories and collect measurements from the IMU sensor.

(c) The obstacle filled with pebbles



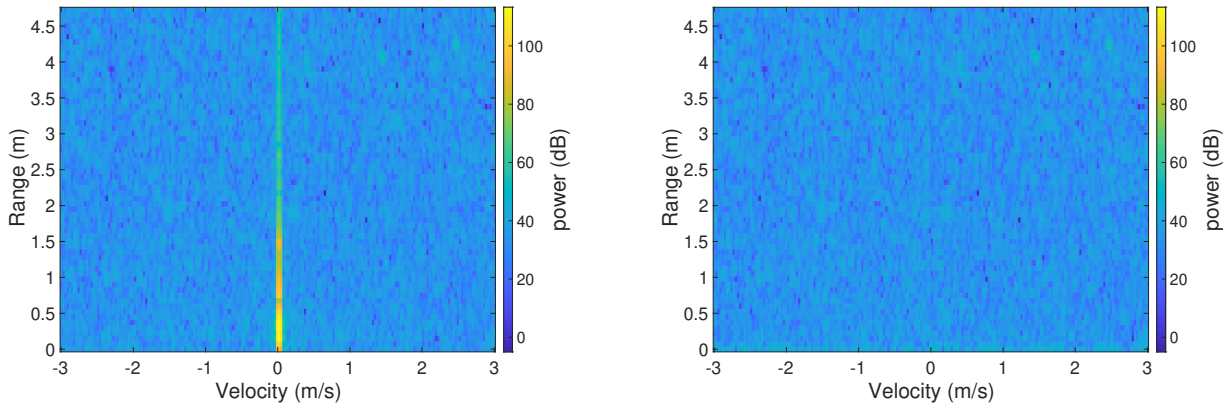
(d) The obstacle filled with soil

(e) The obstacle filled with water, type-I

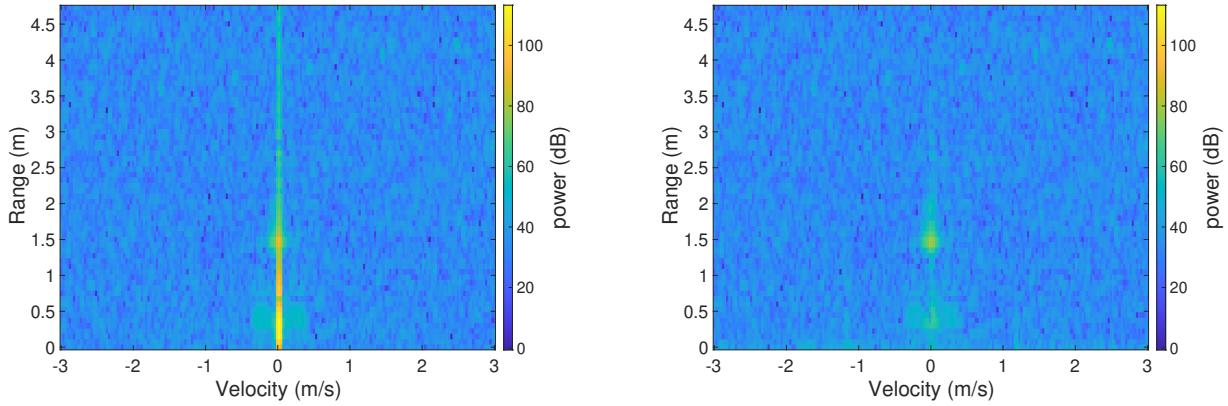


(f) The obstacle filled with water, type-II

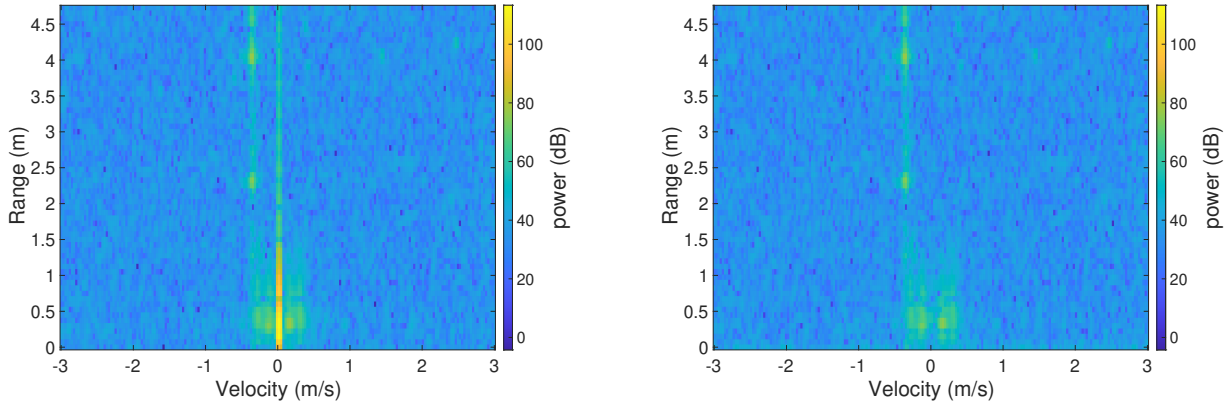
Figure 4.3: Experiment setup: (a) images from the laboratory environment showing (c), (d), (e), (f) the various types of obstacles of different shapes and material, and (b) the moving node having the sensing hardware.



(a) RDM showing no object in view

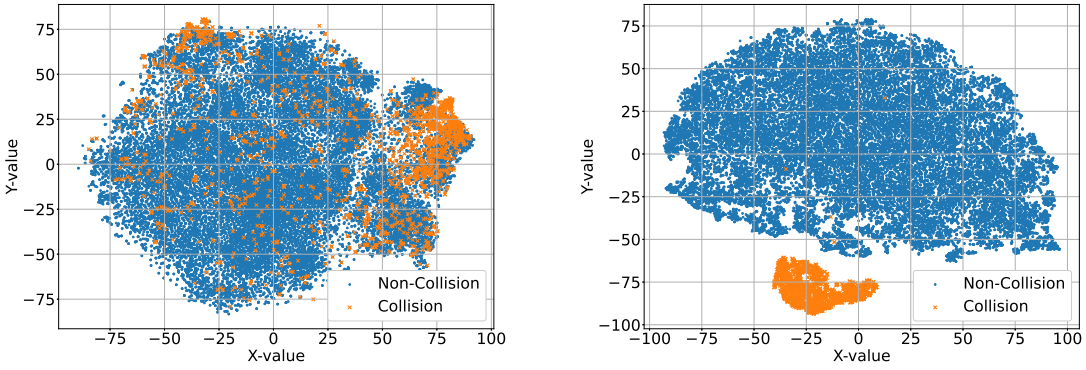


(b) RDM showing one stationary object



(c) RDM showing two moving objects

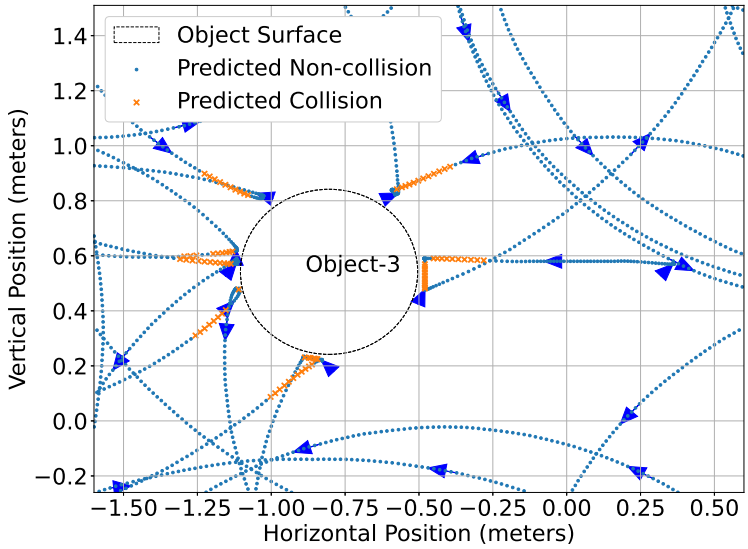
Figure 4.4: Visual demonstration of the effect of PCA-based clutter removal step for the baseline method: Left column shows the RDMs with clutter reflections present and right column shows the RDMs with clutter reflections removed via the PCA for three different scenarios: (a) RDM with no object in the view; (b) RDM with one stationary object in the view; (c) RDM with two moving objects in the view



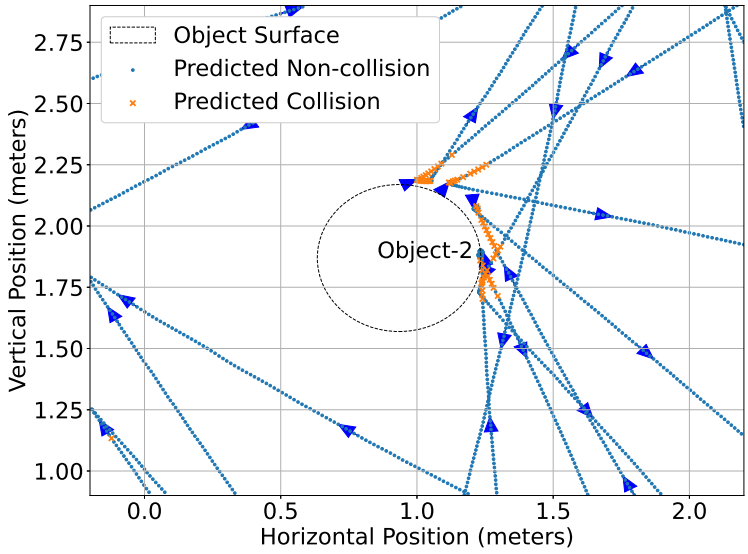
(a) Feature space of the RDMs from an untrained ResNet-18 model.

(b) Feature space of the RDMs after training the ResNet-18 model.

Figure 4.5: Demonstration of successful training of the ResNet-18 model.



(a) Inference results for straight line motion experiment S3 for the moving node near Object-2.



(b) Inference results for curved line motion experiment C3 for the moving node near Object-3.

Figure 4.6: Visual representation of our collision prediction model on straight and curved style of trajectories.

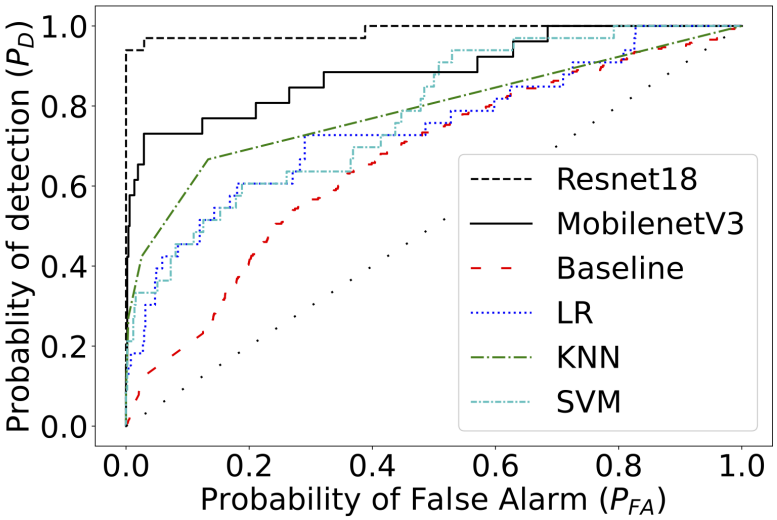


Figure 4.7: Performance comparison in the form of probability of detection versus probability of false alarm for various models for the combined dataset D

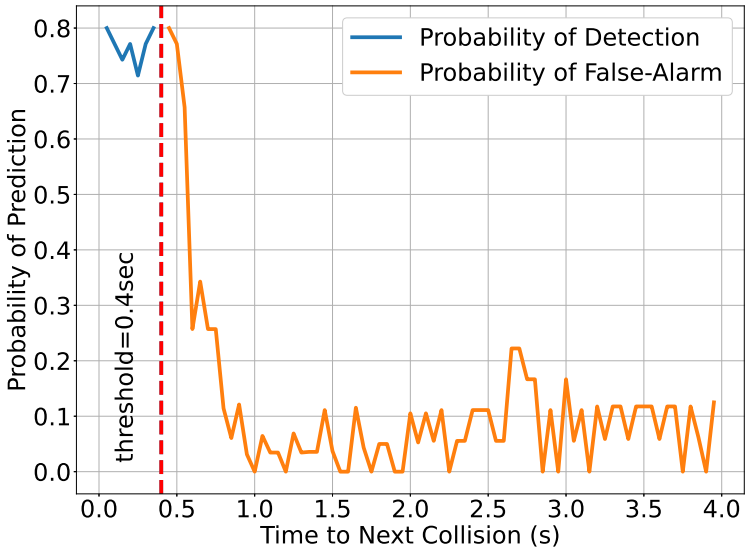


Figure 4.8: Probability of detection and false alarm as a function of time to next collision.

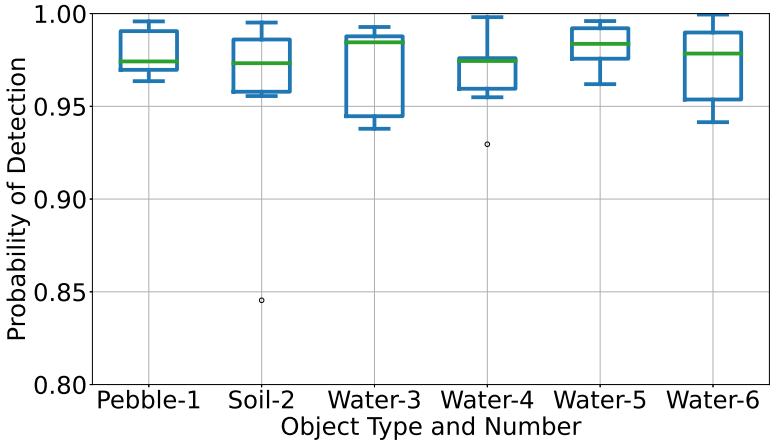


Figure 4.9: Impact of obstacle material on F1 score

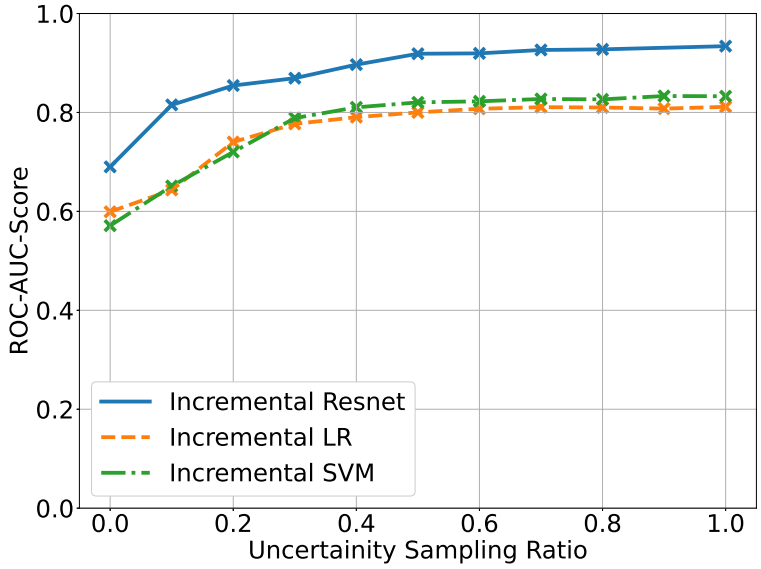


Figure 4.10: Retraining performance of ResNet-18 with two additional incrementally learning models for various sizes based on uncertainty sampling.

Chapter 5

FIELDS: FIngerprint Enabled Localization via Doppler Spread

5.1 INTRODUCTION

Locating stationary and mobile transceivers is one of the most important aspects of wireless systems. Methods that provide robust localization of transceivers in the wireless domain not only enhance the throughput of the communication systems, provide security, and maintain consistent coverage, but can also lead to additional applications, such as navigation and location-based services. However, a wireless channel is a hostile environment, that is, radio waves in the wireless propagation channel undergo a variety of effects such as reflection, diffraction, diffusion, and absorption [29]. These effects are dependent on the channel parameters, the frequency of the radio waves, and the instantaneous location(s) of the transceivers. There are multitudes of wireless channel models that are employed and are being researched in order to realistically characterize these effects through measurements. Thereafter, the correspondingly inferred underlying parameters are then used in designing the localization methods.

The three most commonly utilized measurements in localization methods are the angle of arrival of the signal components, the time of arrival of the signal components, and the *signal strength* (the received power). They are referred in this text as the *location dependent measurements* (LDMs). Each of these category of measurements has seen large body of research for its application in several localization methods [81]. The Doppler frequencies of the received multipath components have also been explored to serve as a valid, although a relatively less common form of measurement, where the difference between

the frequency of the received and the transmitted signals, that is, the Doppler effect, has been shown to provide accurate localization [75], [40], [45].

After one or a combination of such measurements is collected, methods classified as either the *geometric methods* or the *fingerprint methods* are used in order to provide an estimate of the location of the target transmitter. The geometric methods can be seen as the 'traditional' form of localization, the GPS being the most common example, where the location of the transceivers is estimated by exploiting the geometry-based relations of the aforementioned measurement(s). However, the geometric methods suffer heavily from assuming line-of-sight (LOS) conditions and requiring information from more than one transceiver pair, which limit their performance, as is usually experienced in the case of GPS. Furthermore, when the transmission occurs in densely cluttered environments, multipath conditions introduce fluctuations in the collected measurements, which leads to erroneous location estimates.

On the other hand, instead of removing these effects, fingerprint-based localization methods use them to their advantage. The aforementioned measurement(s) are seen as a 'fingerprint' of any particular location and a database of such fingerprints for all the locations is created. Each fingerprint is seen as a measurement that captures *all* the impacts that are induced by the channel on the propagation between the transceivers for a particular location. These impacts not only reflect the desired location-specific information such as the distance and direction of the transceivers, but also capture the location-specific conditions that are otherwise considered as 'non-ideal', such as the amount of non-line-of-sight' (NLOS) and the multipath conditions at one location. For all the locations in any kind of surveyed area, a database of measurements is maintained and the location(s) can be estimated after performing a 'pattern-matching' of the new fingerprint with the ones stored in the database.

In addition to providing accurate location estimates irrespective of the LOS and multipath conditions, an important aspect in the localization methods is of privacy. In Fig. 5.1 we present the two types of scenarios where localization of a node is desired. In cases where the location of a node is provided as a service to an end user, the measurements sent by the end user to the central processor, shall not be able to give any unintended information about the end-user. To ensure this, security features using the encryption techniques can be used such that these techniques, albeit an additional step, secure both

the end-user's reported measurements as well as the proprietary database maintained by location-service provider [37] [80]. Prior to that, however, we can extract only the most relevant information which can provide an accurate location estimate when it is shared with a service provider, while discarding the rest of the signal which might otherwise reveal any private information. In the other scenario, location is provided as a monitoring service, for example, by a central monitor to locate any spurious or unauthorized transmission by an unknown adversary transmitter within an area. In such cases, privacy-preserving method entail estimating the location of the adversary node only through the captured measurements, as the central monitor does not have any other form of record about and from the adversary node. All the aforementioned categories of measurements meet this criteria, that is, once the values are extracted by an end-user as a measurement, discarding the rest of the waveform and extracting the necessary measurements, thus allow a privacy-preserving scheme.

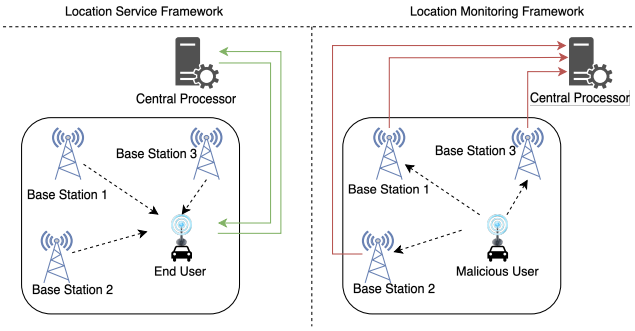


Figure 5.1: Location provided as a service to complying end-users versus location used as a measure to track malicious users.

In this paper, to capture the location-specific characteristic through the received signals and implement the privacy-preserving feature to localization, we use the *frequency* of signals as the measurement of choice and implement a fingerprint-based method for multipath-rich and mobile wireless systems. We argue in favor of a novel fingerprint, the Doppler spread, which captures the location and velocity specific characteristics due to both multipath and motion in a finer detail than just a single, difference in frequency (the Doppler frequency) measurements. The features of this fingerprint are learned using a deep learning architecture. We use this trained model to test the effectiveness of our fingerprint's learned features for implementing a localization pipeline on a real-world, outdoor radio test-bed.

5.1.1 Contributions

We summarize the major findings that we contribute via this paper as follows:

1. We collect an extensive set of Doppler spread measurements that emulate the kinds of Doppler measurements that could be made of signals from a mobile transmitter by an outdoor, multi-cell wireless infrastructure. These are collected over a large, 2 km by 2 km area in a dense University campus environment. *These measurements, described in this chapter and made publicly available, are by far the largest such Doppler measurements we are aware of in our research field.*

2. We present a database collected over multiple days and under a variety of real-world conditions, to show that the fingerprints contain location and velocity information, and is repeatable for a given location and velocity. We quantify the ability of using these Doppler spread measurements for localization using a deep learning model trained on the database. The results show a median RMSE of 124 meters over the multiple datasets collected. These preliminary results for Doppler spread-based localization show promise for future localization system development.

While other researchers have built Doppler shift-based localization systems, they have not attempted to deploy them on real-world multipath channel measurements. One researcher told us in personal conversation that the method did not work when they tested it outdoors in a multipath environment. Doppler shift corresponds to the frequency difference in one multipath component, presumably the LOS component. Methods that assume only one Doppler shift (and one multipath component) do not do well in real-world situations with many multipath components. Our work relies on Doppler spread, i.e., the cumulative impact of all multipath components spread across a Doppler frequency axis. Our work provides a proof-of-concept that Doppler spread can be useful in localization.

5.2 BACKGROUND

In a wireless channel, the *instantaneous* location of the transceivers determine the affect on the propagation conditions that are faced by radio wave propagating at a particular

frequency within the channel. In a more general sense, localization can be seen as a "parameter estimation" problem, where the channel acts as a data-generating process, which can be represented with the help of an underlying model, and this model has the the *instantaneous* location as the parameter of interest. More directly, the localization process entails the usage of any method that associates the location to a physically-measurable *entity* (that is, the generated data) which gets collected at the transceiver(s). These entities or the measurements, in fact, a function of the location, and are further processed by any of the localization method to estimate the location of transceiver(s). Using the geometric methods, the collected LDMs first provide intermediate measurements such as distance or direction using an underlying propagation model, that are then used in geometry-based algorithms, such as tri-lateration and tri-angulation, respectively, to obtain the final location estimates. In contrast, fingerprint-based localization methods, the relation between the location and the measurements is explored through pattern recognition. Instead of working with an underlying model to map the location and its respective measurements, the measurements are assumed to have a pattern or a distribution, such that two measurements with similar characteristics are concluded to have occurred due to the same location. The main proponent of the fingerprint method is that, instead of minimizing or eliminating the effects on the measurements due to the non-ideal propagation conditions such as NLOS or small-scale fading, the effects are themselves considered as worth exploiting so as to distinguish one location from another. That is to say, each location is assumed to have a unique effect on the measurements, which theoretically makes two dissimilar measurements, and thereby two locations, distinguishable. Fingerprinting methods entail the learning from, or the modeling of, these effects in order to dissociate two measurements according to their respective location(s).

In terms of parameter estimation, every single one of the locations $\boldsymbol{y}$ is the considered as the parameter of the data-generating process (that is, the channel). Here on, statistical inference techniques can be applied to *infer*, that is, to provide an estimate of the parameters (the locations) from the associated data. For example, by using techniques such as the maximum likelihood estimation (MLE) or the maximum a-posteriori probability estimation (MAPE), a relation between the parameter $\boldsymbol{y}$ for the collected fingerprint data $\boldsymbol{d}$ can be obtained such that the likelihood function of the parameter, $\mathcal{L}(\boldsymbol{y}; \boldsymbol{d})$ or the posterior distribution of the parameter, $p(\boldsymbol{y}|\boldsymbol{d})$, is maximized, respectively.

The steps for fingerprint methods can be summarized as:

- *Measurement*: Each base-station or the transceiver is responsible for locally obtaining the measurements that is, the 'fingerprint' or FP for all the locations that are visited within a fixed area. The measurement can in the form of any of the individual LDMs, a combination of multiple LDMs, or some function of the LDMs.

- *Database creation*: During the offline phase, a central unit receives the measurement(s) from all the participating base-station(s) and a 'map' or a database is created. Using this database, the relation between a location $\boldsymbol{y}_l$ and its corresponding fingerprint data is learnt, for example, in the form of the posterior distribution $f(\boldsymbol{y}_l|\boldsymbol{d}_l)$ where f represents the probability density function of $\boldsymbol{y}_l$ given $\boldsymbol{d}_l$ as the measurements, or in the form of machine learning models, such as the discriminative methods (support vector machines) or the generative methods (neural networks).

- *Decision making*: During the online phase, the central processor can match an 'online' fingerprint with the stored fingerprints from the database and make a decision about its location based on a chosen metric. Location estimates are obtained through maximizing the match, for example, maximizing the correlation or maximizing the likelihood of the 'online' fingerprints with respect to the fingerprint entries stored in the database.

5.3 METHODOLOGY

5.3.1 Notations and Problem Statement

Consider an area in 2-D having multiple locations, where an instantaneous location $\boldsymbol{y}$ is defined as a vector of four variables, that is, $\boldsymbol{y} = [p, q, u, v] \in \mathbb{R}^4$, where p and q represent the location coordinates, while u and v represent the velocity components in 2-D. With total of K base-station(s), a data-point is represented as $\boldsymbol{d}$, which is a vector of size $1 \times K$. Next, we introduce the fingerprint extraction process.

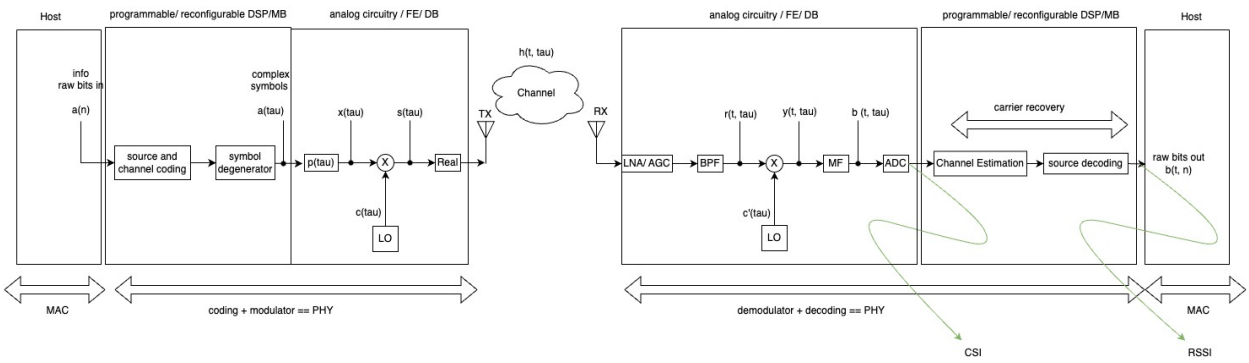


Figure 5.2: Propagation channel

5.3.2 Propagation Channel

A typical propagation channel employed in the transmission of information from a transmitter radio device, such as software defined radios (SDR), to a receiver radio device is presented in the Fig. 5.2. At the transmitter side, a base-band (complex) signal $a(\tau) = Ae^{j\angle A}\delta(\tau)$ undergoes pulse shaping through the filter $p_T(\tau)$, to produce $x(\tau)$. This base-band signal modulates the carrier signal $c(\tau) = \exp(2\pi f_c \tau)$ with f_c as its the center frequency produced by the local oscillator (LO) of the device. The desired, up-converted pass-band signal $s(\tau) = x(\tau)\exp(j2\pi f_c \tau)$, is finally transmitted as $\mathcal{R}\{s(\tau)\}$. Considering the multipath propagation conditions, the received signal $r(\tau, t)$ is the result of convolution between the impulse response of the channel with the transmitted signal, $\mathcal{R}\{s(\tau) \circledast h(\tau, t)\}$. Down-converting it through the receiver radio's $c'(\tau) = \exp(2\pi f_c' \tau)$ and low-pass filtering, we obtain $b(\tau, t)$ as:

$$b(\tau, t) = \sum_{m=1}^{M(t)} \frac{A}{2} e^{j\angle A} e^{j\phi_m} e^{-j2\pi f_c \tau_m(0)} e^{j2\pi f_m^D(t)\tau} e^{j2\pi f_{off}\tau}, \tag{5.1}$$

where, $f_{off} = f_c - f_c'$, is the offset between the oscillators of the transmitter and receiver radios, and f_m^D is the Doppler shift on the m^{th} multipath.

5.3.3 The Proposed Fingerprint

Finally, at the k^{th} receiver, Fourier transform of the down-converted signal is performed to obtained the spectrum which is the desired fingerprint, that represents that spread in the Doppler frequency as,

$$B(f,t) = \sum_{m=1}^{M(t)} \frac{A}{2} e^{j\angle A} e^{j\phi_m} e^{-j2\pi f_c \tau_m(0)} \delta(f - f_{off} - f_m^D(t)) \tag{5.2}$$

5.4 DATA

We perform as series of outdoor experiments in a densely populated city-wide area and obtain multiple datasets of fingerprints of all the locations traversed within the area. The final database created is representative of the locations in the surveyed area and thus, can be used in the fingerprinting methods to obtain the location estimates.

5.4.1 Measurement Platform

We perform a series of large-scale, over-the-air experiments on the platform for open wireless data-driven experimental research (POWDER) [17]. During one experiment, a single mobile node, acts as the transmitter while it traverse on a specific route in the University of Utah campus, facing real-world multipath environments caused due to the day-to-day traffic and crowd conditions. The base-stations buildings act as receivers for our experiments and they simultaneously record the IQ values of the received signal during the entire duration of every experiment. The simultaneous reception is conducted with the help of an online tool called SHOUT, where a central system collects all the IQ values reported throughout the experiment from all the base-stations into one file for each experiment [78]. The SHOUT system also stores the reported GPS coordinates from the mobile node throughout the duration of the experiment into one separate file. Note that in this text, the 'mobile node' and 'bus' are used interchangeably.

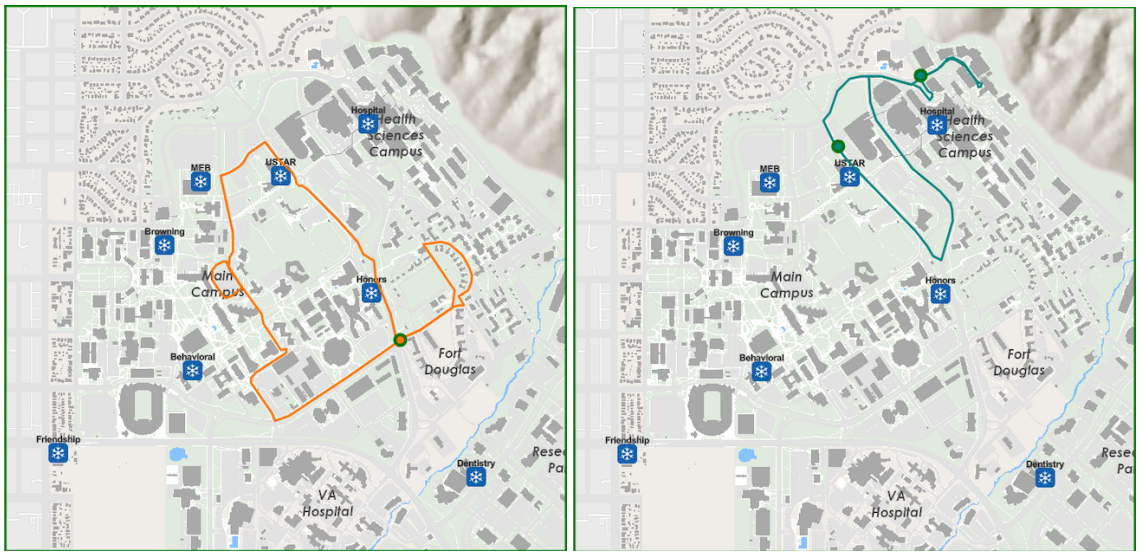
5.4.2 Hardware

On the transmitter side, the SDR on a bus is USRP B210. The AD9361 RFIC transceiver used in the B210 provides an adjustable and programmable output power ranging from -4 dBm to 6 dBm. However, the final power level of transmission is not constant or is fixed for our experiments, and is instead determined by the RF front-end and the antenna used with the SDR. For our experiments, the power level of the transmission is amplified by a gain of 79 dB. On the receiver side, each base-station has a SDR as the USRP X310. The RF front-end amplifier is set to amplify the received signal by a gain of 30 dB. All the receiver SDRs have an external clock such that their time and the frequency of operation are synchronized with the help of the White Rabbit Synchronization system to achieve an frequency accuracy of up to $\pm$25 ppb [23]. It is noted that due to the mobility of the bus, the transmitter SDR is not synchronized externally through the White Rabbit. Thus, with a limited accuracy as $\pm$2.5 ppm of the internal frequency reference in B210, the operational frequency of $f_c = 3.515$ GHz has an offset, known as the carrier frequency offset (CFO), of up to 8.78 kHz [25]. Thus, there exists a mismatch, or the offset, of $f_{off} <= 8.78$ kHz between the frequency of operation for the bus SDR and the frequency of operation for the synchronized base-station SDRs.

5.4.3 Transmission Parameters

From the bus SDR, we continuously transmit an unmodulated, complex sinusoidal wave at a frequency $f_c = 3.515$ GHz. The maximum speed of the bus is expected to be $\vec{v} = 21$ mps. Thus, the maximum Doppler shift at the central frequency of $f_c = 3.515$ GHz is expected to be $f^D_{max} = f_c \vec{v} \cos\beta_m(t)/c = \pm 246.22$ Hz. In order to capture the effect of this speed in a time-varying multipath environment such that M Doppler shifts are resolved through our fingerprint, the sampling rate of all the SDRs involved is maintained to its lowest possible value of $f_s = 0.22$ MHz, which safely meets the Nyquist criteria of $f_s >= 2(f^D_{max} + f_{off})$. Furthermore, in order to resolve the fingerprint to its finest resolution, we set the number of IQ values collected at each receiver during one observation to be $N_s = 2^{17} = 131072$, which is the maximum possible N_s using SHOUT. Thus, the finest possible frequency resolution for our fingerprint is $f_s/N_s = 1.6784$ Hz, which can capture the Doppler shift on a multipath due to velocities as low as 0.0508 mps. Finally, each

t^{th} observation at time instance t is processed to produce one Doppler spectrum as its fingerprint, according to Eq. 5.2.



(a) Orange route (b) Green route

Figure 5.3: Two types of trajectories

5.4.4 Experiments

We work with a total number of $K = 5$, synchronized base-station buildings (named Behavioral, Honors, Hospital, SMT, UStar) as the receivers for all of our experiments. During one experiment, the mobile bus node is serving as the transmitter, while it traverses any one of the pre-specified routes. As shown in Fig. 5.3, two different routes (orange and green) are covered in our experiments. There are six experiments for the orange route and 18 experiments for the green route. In total, 23 experiments are conducted over a span of six days. A total number of $T = 10,400$ data-points are thus collected as one database $\boldsymbol{D}$. It should be noted that the size of the database is $T \times K$, instead of $L \times K$, where $T > L$ since for the l^{th} location y_l, more than one observation can be recorded in the database $\boldsymbol{D}$. Also, depending on the hour and date of the experiment, the available bus, and thereby the hardware configuration of the SDR used in an experiment, can vary between experiments, according to the schedule of buses during that hour and day of experiment. Thus, the frequency offset between the frequency of operation for the bus's transmitter SDR and

Date	Bus	Route	Dataset	# of FPs
30 Jan'23	B1	green	D1	500
			D2	500
			D3	200
	B2	green	D4	200
			D5	200
2 Feb'23	B3	orange	D6	500
			D7	1500
6 Feb'23	B4	green	D8	500
9 Feb'23	B5	green	D9	500
			D10	500
			D11	500
14 Feb'23	B6	green	D12	500
			D13	500
			D14	500
			D15	500
			D16	500
			D17	100
16 Feb'23	B7	orange	D18	500
			D19	500
	B8	green	D20	500
			D21	500
			D22	100
			D23	100

Table 5.1: Summary of the data collected. In total, 23 data-sets are collected.

the synchronized frequency of operation for the base-station SDRs can differ between two experiments. An elaborate description of the entire collected data is provided in Table 5.1.

The number of base-stations are constant $K = 5$ for both kinds of trajectories, however, the green route covered a trajectory of approximately 2.736 kilometers, where as the orange route covered 5.632 kilometers. Effectively, the two types of routes follow a different kind and size of an area. Moreover, for the shorter, green route, there are a total of 19 datasets that collect an overall $N = 7400$ data-points, while for the longer, orange route, there are a total of 4 datasets, collecting $N = 3000$ data-points. Thus, the density of the locations covered in one route differs from the other.

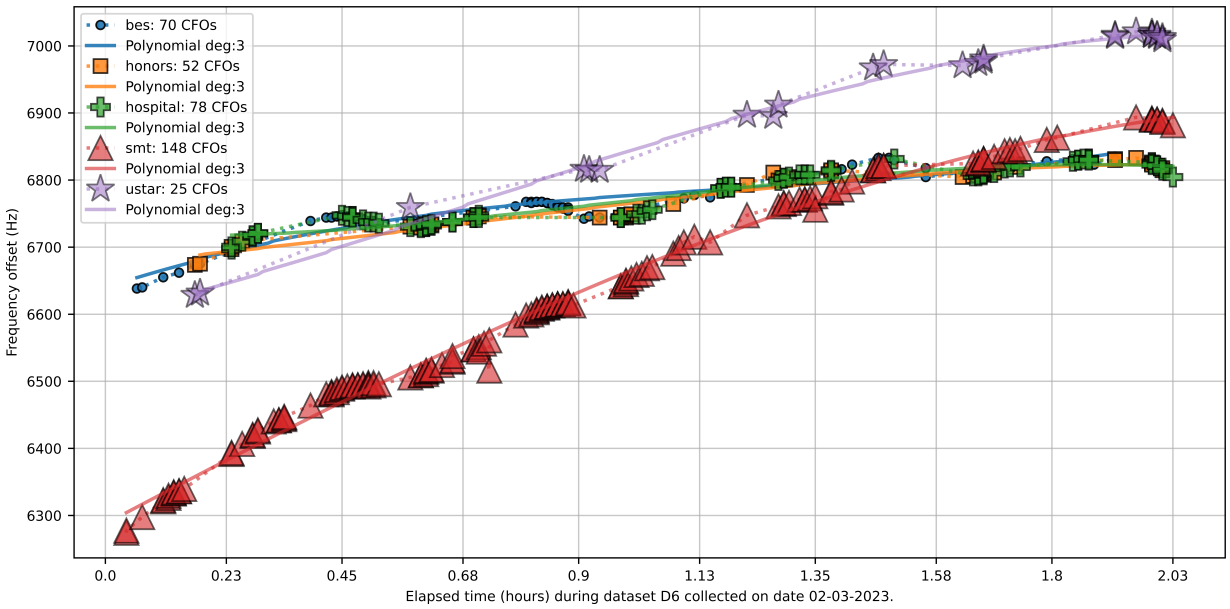
5.5 ANALYSIS AND RESULTS

This section presents the characteristics of the Doppler spread as a location dependent measurement and its performance as a proposed fingerprint for localization.

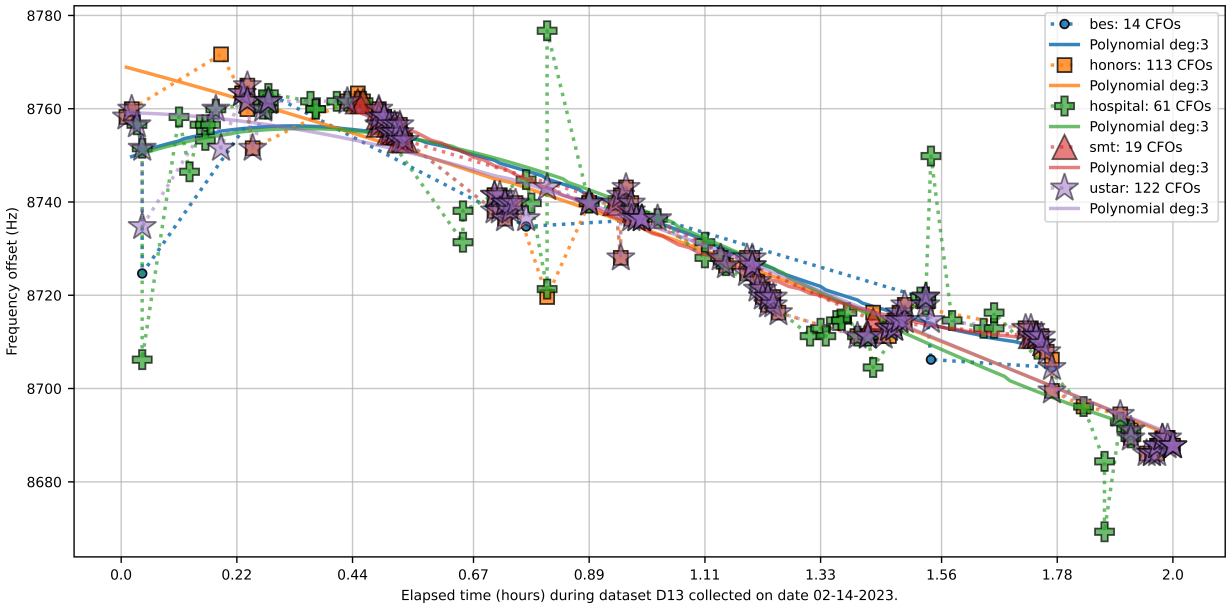
5.5.1 Data Pre-processing

The carrier frequency offset (CFO) between the oscillators of transmitter and receiver, f_{off}, causes the desired fingerprint to shift in the frequency domain. This offset is undesirable because it is a measurement effect unrelated to node location or velocity. Thus, as a pre-processing step of removing this effect, we approximate the offset between the mobile node and each base station with the help of the measured fingerprints themselves. At the instances when there is no motion between the transceivers, f_D should be 0 Hz, thus, the fingerprint from (5.2) will show the instantaneous offset. That is, the CFO f_{off} is the value of the frequency bin of the peak detected in the fingerprint.

We collect all such measurements when the mobile node was stationary and the received signal is above a noise threshold for each base station. As an example of this pre-processing step, we show in Fig. 5.4 all the detected peaks from the fingerprints captured during two different datasets `D6` and `D13`. It is emphasized that that the CFOs are not consistent over time, and that they are related to the conditions of the dataset collection, such as temperatures of the day, trajectory followed, and signal to noise ratio in the signal. We approximate f_{off} by performing a 3^{rd} degree polynomial fit on the collected CFOs, such that the spectrum of the fingerprint can be shifted by the interpolated f_{off}, for both type of measurements, non-stationary and stationary, according to the time of the fingerprint during the experiment, and producing the frequency-corrected fingerprint in the database.



(a) Route Orange CFO



(b) Route Green CFO

Figure 5.4: Frequency offset trend

5.5.2 Fingerprint Characteristics

Fingerprint as a Function of Location

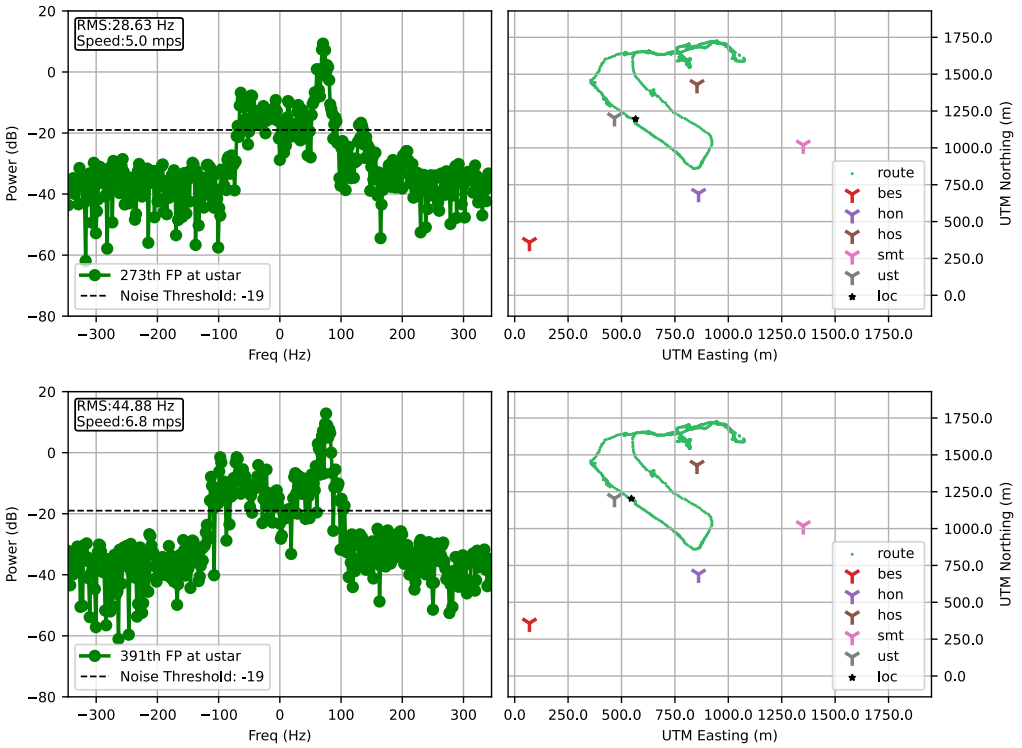
For a particular instantaneous location $\boldsymbol{y}_l$ at any particular time t and with M as the instantaneous number of multipath components experienced by the narrow-band signal, the fingerprint is calculated using (5.2) at a base station k. Examples of the fingerprint are shown in Fig. 5.5 for the base station `ustar` for a number of varying conditions, such as different locations, routes, and the instances of time. During the experiment for dataset D20 on the green route, it can be seen in Fig. 5.5a that the two fingerprints captured for a *nearly* same location, however, at different instances, that is, $n = 273$ and $n = 391$, have similar characteristics or *shape* of their Doppler spread fingerprint.

For different location, a different number of location-specific multipath components are experienced by the signal, thus producing a different Doppler spread shape. This is shown in Fig. 5.5b, having examples of the fingerprints captured during dataset D6. It shows similar shape of this location's fingerprint at two different instances, that is, at $n = 49$ and $n = 136$.

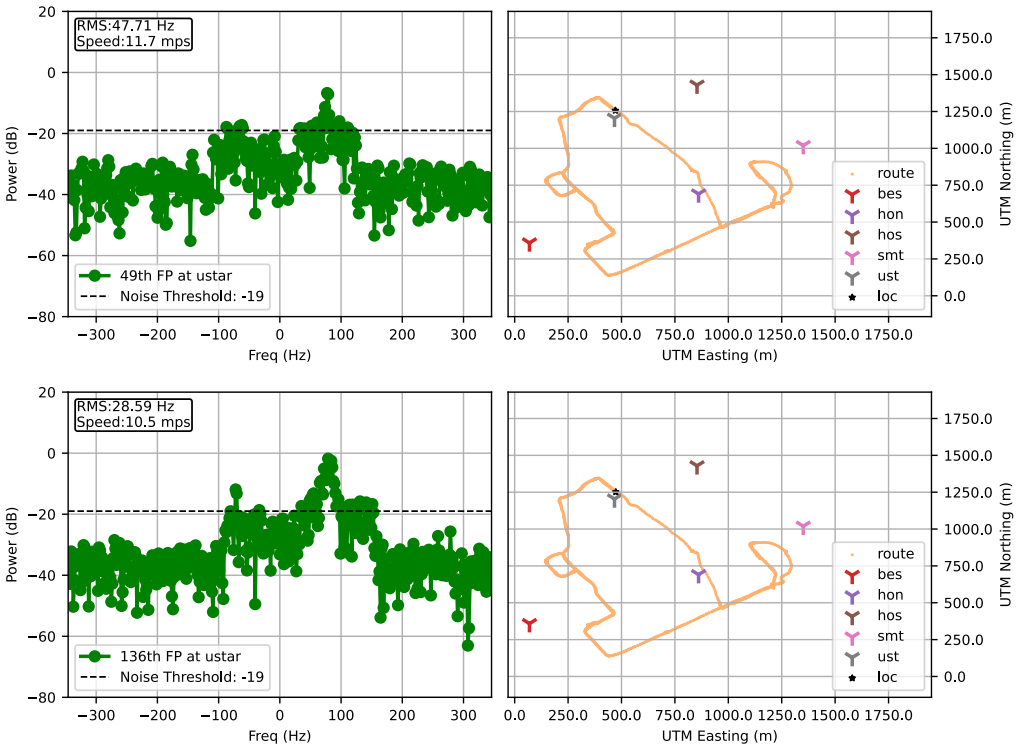
It is noted that for our fingerprint, the range of frequencies is maintained to be within ± 246.22 Hz, which is according the maximum speed of the mobile node that is assumed to be 21 mps.

Fingerprint as a Function of Speed

As presented in (5.2), the obtained fingerprint is a function of velocity of the mobile node, where the shift in frequency due to the relative motion between mobile node and base station, or the Doppler shift, gets *spread* over a range of M frequency bins, as the m^{th} multipath component induces a copy of the Doppler frequency scaled by a factor of $\cos(\beta_m(t))$, where $\beta_m(t)$ is the angle between multipath m and the direction of motion of the node. The full Doppler spread is the sum of the Doppler shifts from all multipath components, weighted by their amplitudes.



(a) Experiment 02-16-2023 14-55-52



(b) Experiment on 02-03-2023 10-39-20

Figure 5.5: Fingerprint captured at `ustar` for the same location during two routes, at different times.

The width of the Doppler spread is often quantified by the root mean square (RMS) Doppler spread. Its value (in Hz) is obtained from the Doppler spectrum as,

$$\sigma_{f_D} = \sqrt{\frac{\int (f_D - \mu_{f_D})^2 \Phi_B(f_D)\, df_D}{\int \Phi_B(f_D)\, df_D}} \tag{5.3}$$

where, μ_{f_D} is the mean of the Doppler power spectrum, and is calculated using distribution of the power spectrum $\Phi_B(f_D)$ of the received signal as,

$$\mu_{f_D} = \frac{\int f_D \Phi_B(f_D)\, df_D}{\int \Phi_B(f_D)\, df_D} \tag{5.4}$$

Using Eq. 5.3, we demonstrate that the maximum σ_{f_D} that we can observe in practice is proportional to the instantaneous speed $\vec{v}$ of the relative motion of the nodes. From our collected datasets, consider D13, as is shown in Fig. 5.6. For the same location, the number of multipath components stays consistent over time and an increase in the speed for the same location produces a wider shift in the frequency spectrum, thus leading to a higher RMS Doppler spread value.

However, it is noted in (5.3) that the RMS spread is also dependent on the distribution of the Doppler spectrum $\Phi_B(f_D)$. This Doppler spectrum may or may not contain multipath components with Doppler shift close to the maximum positive and negative Doppler frequency. In particular, even for a mobile node, if only one multipath component arrives, regardless of what its Doppler shift is, its Doppler spread will be observed to be close to zero.

Doppler spread which can also be affected by environmental conditions unrelated to the node's motion, that may or may not be known to our measurement system. Movement by entities other than the mobile node, for example, other vehicles driving nearby can impact one or more multipath components. A driving vehicle other than our node can induce a Doppler frequency shift in a multipath component that is up to double its speed relative to the node. It can lead to varying numbers of multipath components even at a single location, such as intersections or areas of dense traffic, and the Doppler spread value within a fingerprint is a function of both speed and location. This can be demonstrated with the help of Fig. 5.7a and Fig. 5.7b.

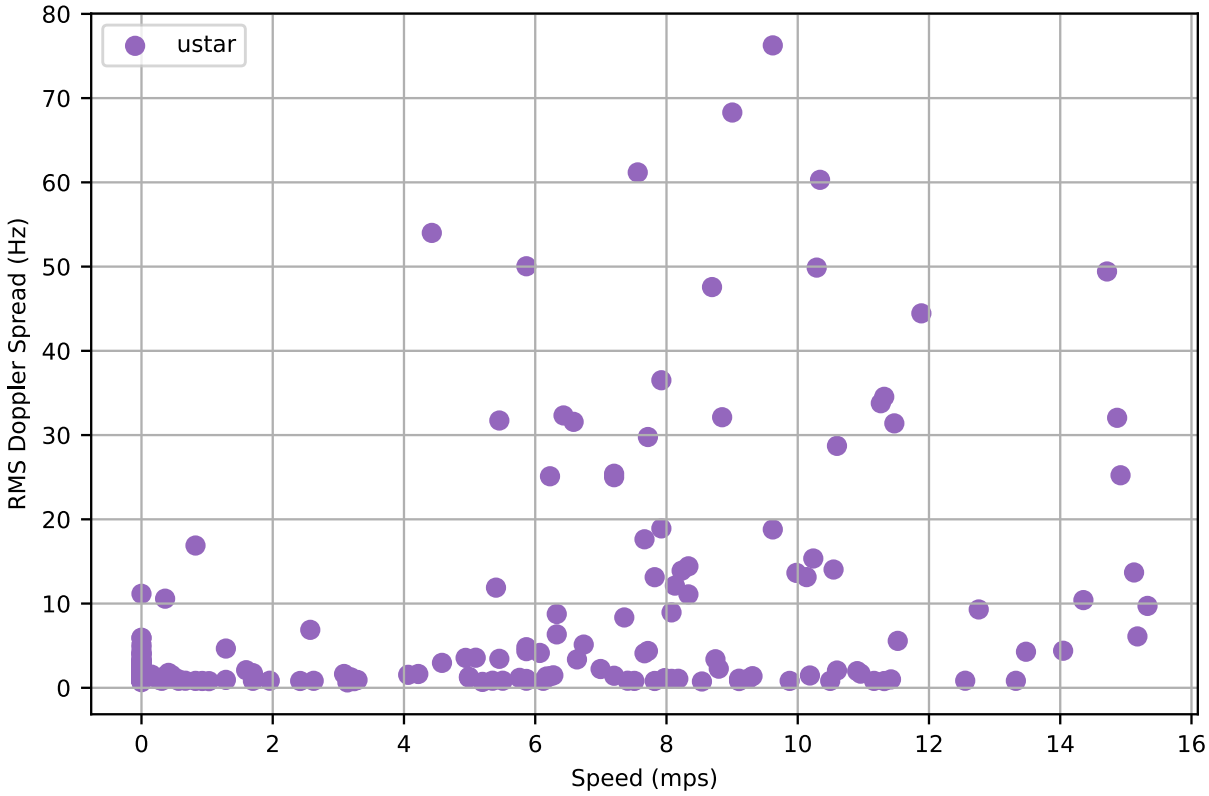
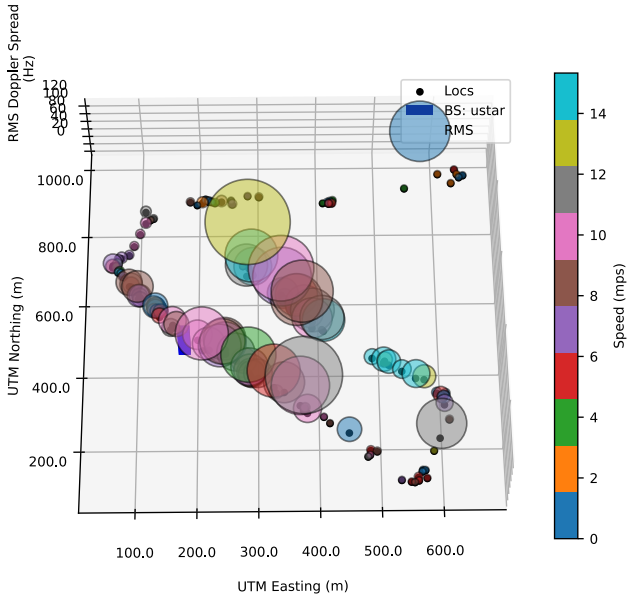
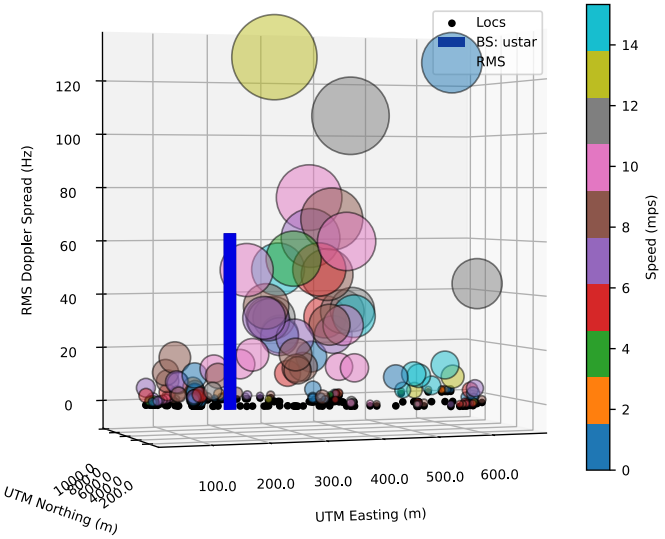


Figure 5.6: RMS Doppler spread as a function of speed, values at `ustar` from dataset D13.



(a) RMS Doppler spread as a function of location.



(b) RMS Doppler spread as a function of both speed and location.

Figure 5.7: RMS Doppler spread values at `ustar` from dataset D13.

5.5.3 Parameters for Deep Learning Model

We employ a machine learning framework which utilizes the collected datasets to learn any intricate and any flexible number of features of the fingerprint by tuning the parameters of a CNN-based, deep learning model (ResNet-18). During the training phase, the training dataset is split into smaller batches which are inputted to the model. The model obtains the output and we use the sum of squared errors (SSE) as the cost function for each batch. Next, an optimizer adjusts the model's parameters for every batch during the training phase, and this batch wise propagation of each dataset is repeated for a certain number of epochs such that, in the end, we obtain a trained model with minimal overall loss. We choose the Adam optimizer for our model, and it is trained for a total 100 epochs. The learning rate for the optimizer is set to be 0.001, and it furthermore follows a cosine annealing, such that the learning rate gradually decreases over epochs.

For any dataset, randomly selected 90% of data-points are used in training the model, and in the testing phase, the trained model uses the rest of the 10% data-points and infers their corresponding output (the location coordinates). The number of data-points used in training is an important factor in determining the performance of the model on that dataset. A higher number of training data-points provides more examples of the underlying patterns to learn from and thus, a better generalization performance on unseen, testing data-points. Thus, we split any dataset into a $90 : 10$ ratio between the training and testing data-points, using a significantly larger number of data-points in training than in testing. This ratio is also maintained constant for every dataset used for training the model, irrespective to the total number of data-points in the dataset. To measure improvements on the estimated locations while training, a random 10% data-points are used as the validation data-points. We also maintain an $L2$ regularization penalty of 0.001 to avoid over-fitting.

Once the training phase is concluded, the trained model is used as the location estimator for the testing data-points.

5.5.4 Localization Performance

The fingerprint based localization methods rely on pattern matching of the fingerprint for one location over time. However, the time varying nature of environment and the non-ideal conditions, such as the remaining CFO error and the varying signal to noise ratio (SNR), can lead to significant variations in the structure of the fingerprint used.

Moreover, since the performance of the deep learning model depends on the number and type of of data-points used in training, we present the estimated locations as a function of the type of dataset, number of data-points used in training, and the approximated CFO.

Location Estimates within a Dataset

In the testing phase, the trained ML model takes any single data-point $\boldsymbol{d}$ from the 10% of the samples from the dataset and provides an estimate for the location, as $\hat{\boldsymbol{y}} = [\hat{p}, \hat{q}]$. We present in Fig. 5.8 the location estimation results for the two datasets that differ in the type of trajectories during the experiments, while keeping their number of recorded measurements nearly similar. For this, we use the dataset $D19$ as an example for the orange route, and the dataset $D20$ as an example for the green route, having a total of $N = 500$ measurements collected, as shown in Table 5.1.

The (magenta color) location estimates made by the trained model are shown along with to their (green or red color) ground truth location coordinates provided by GPS during experiments. For a single data-point, the performance of the model is measured with the help of the error metric e, taken as the Euclidean distance between the estimated locations $\hat{\boldsymbol{y}}$ and the ground truth locations $\boldsymbol{y} = [p, q]$, which are provided by GPS during experiments. The error metric, e, is calculated as

$$e(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \sqrt{(\hat{p} - p)^2 + (\hat{q} - q)^2}, \tag{5.5}$$

and is displayed in Fig. 5.8 as a red or green color solid line for a green or red color ground truth location coordinate. The color assigned to a ground truth coordinate and to its respective error is to represent that the bus node is stationary, that is at a zero speed

at this location if it is colored red, and on the other hand, if the location and the error is colored green, the bus node is moving, that is, it has a non-zero speed at this location.

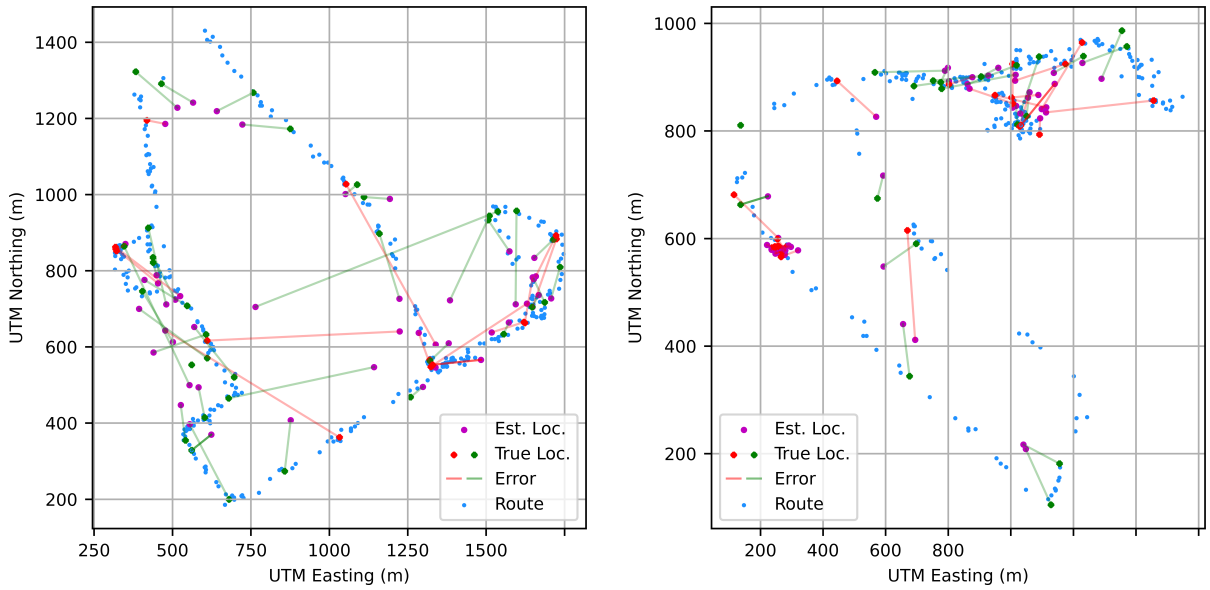


Figure 5.8: Location estimates for the two type of routes taken from dataset D20 and D19. The errors are displayed using the solid straight lines to represent the Euclidean distance between the (green or red color) ground truth coordinates and the (magenta color) estimated coordinates.

It is also noted that the during the trajectories of the two routes, displayed in blue color in the Fig. 5.8, the same number of measurements cover a wider area (1.5 km by 1.2 km) and a longer trajectory (5.623 km) during the orange route, than the area (1.2 km by 1 km) covered and trajectory (2.736 km) in the green route. Thus, in the dataset for the shorter green route, a location can have multiple instances of measurements, thereby, the model trained on multiple data-points for a location, can provide better performance on that location during the online, testing phase. To show this, we present the cumulative distribution of all the errors $e(\hat{\boldsymbol{y}}, \boldsymbol{y})$ calculated during the two datasets in Fig. 5.9. Orange route has larger distance errors pertaining to a sparsely surveyed area, where as, the green route provides smaller distance errors due to more instances of measurements for each locations collected during this shorter route. The median of the Euclidean distance error is reported to be 57.95m for the green route dataset $D19$, a lower value compared to the median error of 165.14m for the orange route dataset $D20$.

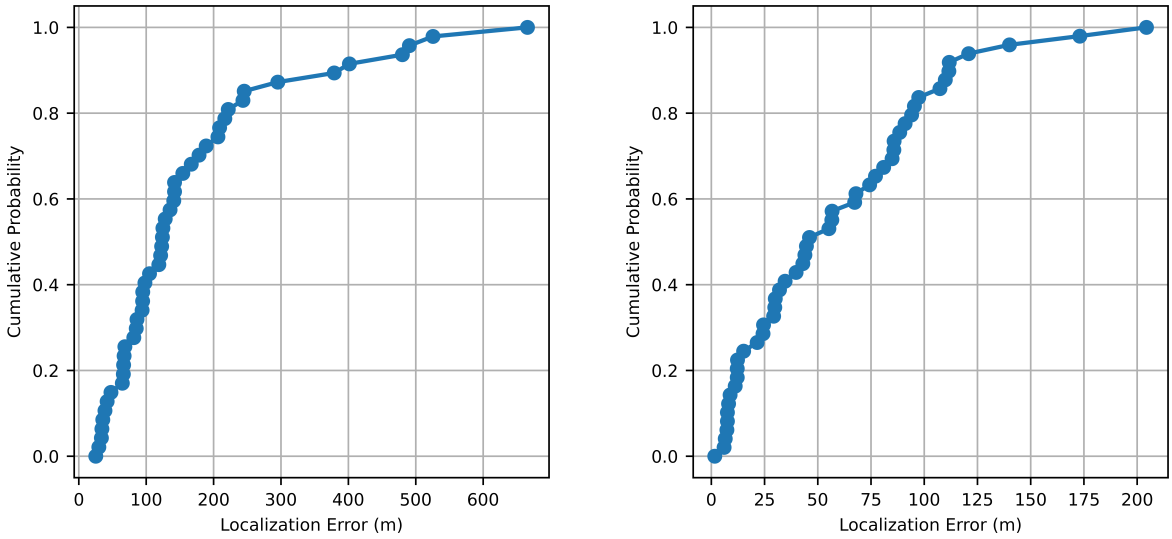


Figure 5.9: Cumulative distribution of the errors of location estimates for the two type of routes taken from dataset D19 and D20.

Location Estimates with Multiple Datasets

In this section, we study the effect of the variations in the multiple datasets on the performance of the deep-learning models. Multiple datasets can be combined together on the basis of certain metrics, such as the similarity of experiments' conditions, to form a database that represents the characteristics of the fingerprints for a certain set of conditions. If however, between two datasets, factors such as the size, time, conditions of the dataset vary a lot, combining the data-points of such datasets into one database and training the model on randomly selected training data-points from this database, can lead to poor performance on the testing data-points. To quantify the performance of the model for a particular dataset, we provide a summary statistics in the form of the root mean square of the error (RMSE) value, in meters (m), as

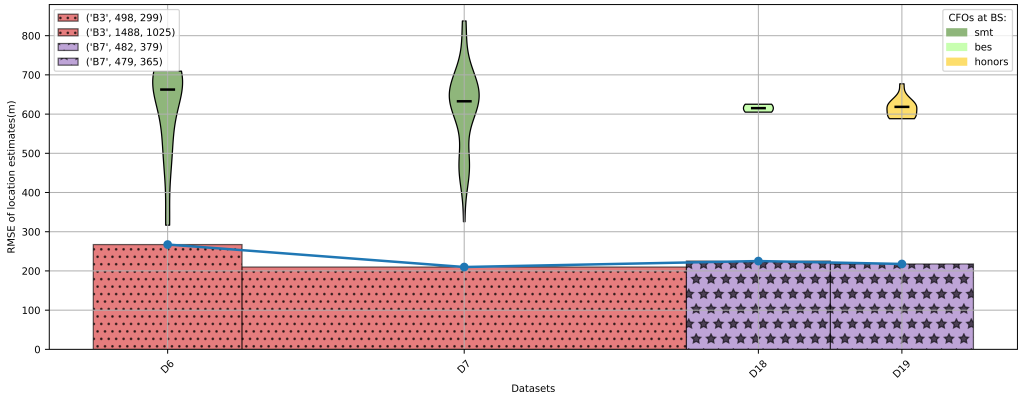
$$RMSE = \sqrt{\frac{1}{N_t}\sum_{l=1}^{N_t}((\hat{p_n} - p_n)^2 + (\hat{q_n} - q_n)^2)}, \tag{5.6}$$

where, N_t is the number of data-points involved in the testing phase of the dataset, which is, a total of the 10% of N. We present in Fig. 5.10, the RMSE values for all the datasets in

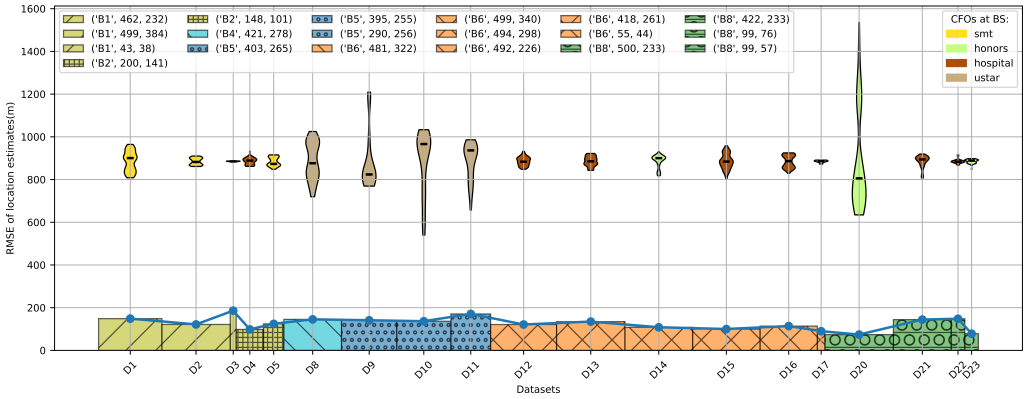
the two types of routes, where the size of each dataset available for training and testing the model is provided as the second entity of the legend of a bar.

For any one route, the datasets vary in the time and day of an experiment, the overall duration of an experiment, the bus node involved in an experiment, the number of stationary measurements for approximating and removing the CFO in an experiment, the variance of CFO over the duration of the experiment. On the basis of locations being similar for one route, all the datasets in a route are used in the training of model. We retain the learned parameters from one dataset, and use the next dataset in the incremental learning of the trained model within one route. The deep learning model is thus incrementally trained and tested on each dataset, with an intermediary testing phase RMSE value to reflect the performance of the model for the data trained. We use the same values for the learning parameters from Section 5.5.3 for the two routes. The RMSE value decreases as the datasets of a particular route are incrementally trained on. This is to say that, as more data-points of similar type of locations are added to the training, the performance is improved, as is shown for the green routes, where the last dataset $D23$ is providing the lowest RMSE for the green routes. However, it is noted that the lowest RMSE value for the orange route is provided by the dataset $D7$, pertaining to the fact that it has half of the entire orange route data-points. Also, recall that $N = 7400$ for the green route and $N = 3000$ for the orange route. Thus, between the two routes, the median RMSE is 124.4m for the green route is lower than the reported RMSE of 221.39m for the orange route. Thus, the performance of the model is directly proportional to the size of dataset.

Performance over different bus crystal oscillators: The trend in the RMSE over the datasets is representative of the various changes between the datasets. We first consider a change in the available bus during two datasets. This change is reflected by the type of the fills in the bar plot of each dataset in Fig. 5.10 and the nomenclature $B1$ to $B8$ in the legend reflects the eight different types of buses that are used in our experiments, as shown in Table 5.1. A change in the bus, that is the transmitter radio's local oscillator crystal, might lead to a different CFO value between the bus transceiver and the base-stations, thereby, the re-shifting of the Doppler spectrum by the approximated CFO can differ between two datasets. However, if a sufficient number of stationary measurements are present, that are used for calculating the CFO according to the pre-processing step presented in the Section 5.5.1, we can capture the CFO trend and its variance during



(a) RMSE of location estimates for the orange route datasets.



(b) RMSE of location estimates for the green route datasets.

Figure 5.10: RMSE of location estimates for all the dataset

an entire duration of experiment, as shown through the example CFO trend in Fig. 5.4, and remove the CFO to obtain correctly shifted Doppler spectrum. Thus, between the datasets, if a change in crystal occurs, it can be correctly compensated if the captured CFO values are from the instances that are spread over the entire experiment. It is also noted that since CFO drifts over time and due to the environmental conditions such as temperature, a larger variance in the captured CFO values is directly proportional to the duration of the experiments, thus, to the size of the dataset. Thereby, a dataset with higher variance in the captured CFO values, provides a better approximated CFO via the 3^{rd}-degree fit of our pre-processing step. For each dataset, we present a violin plot for the distribution of the captured CFOs at the one base-station that has the highest variance in its CFO values. Thus, with datasets having the most number of and a variety of CFOs

captured and removed, we obtain lower RMSE error values, as is shown for the change in bus between in dataset $D3$ and $D4$, between $D17$ and $D20$, and we obtain higher RMSE value between a change of bus between $D7$ and $D18$ pertaining to the lower range of CFOs captured in the latter dataset.

Performance over days: Lastly, in Fig. 5.10, the color of the bar plots reflect the day of the collected dataset, showing a total of 5 days of data collection for the green route and 2 days for the orange route. Within a route, the RMSE value reduces over the course of the day as more datasets are added, when the other confounding factors are not varied. For example, for green route, the six datasets collected on Feb. 14^{th}, the first five datasets are having a nearly consistent size, same bus, and similar CFO distributions, thereby witnessing a downward trend to the reported RMSE values, as well an impressive lowest RMSE for the day through the last reported dataset, despite its small size. That is to say, that model is suitable to incrementally train for the day in consistent conditions. Similar trend is observed during the days of Feb. 3^{th} and Feb. 16^{th} It is noted that for a change of day, the bus crystal is also changed, thus, the effect of a multiple days is studied as the condition of the change in the bus itself.

5.5.5 Future Work

Despite its promising results of the proposed Doppler spread to function as a localization fingerprint, the range of the frequencies within the fingerprint is narrow and the fingerprint is highly sensitive to noise. Due to the time varying and sensitive nature of the proposed fingerprint, it is desired to utilize higher frequencies to obtain wider Doppler spread, depending on the available parameters in the radio device used, the speed of motion to track, and the environment of operation such as indoor vs outdoor conditions. Moreover, working in an environment that has a higher number of multipaths could provide additional location-specific features to the model, thereby, conducting experiments in indoor scenarios through different form of devices, such as UWB radios, is a future consideration. Lastly, for building real world applications, fingerprint databases need to be updated regularly along with the model utilized to learn the features from the database, since the various unexplored factors can adversely affect the performance of the models.

Chapter 6

Conclusions

6.1 Contributions of Dissertation

This thesis explores radio devices and radar sensors in building real-time collision prediction and localization. Collision prediction is presented as a relative localization task, where obtaining a node's position with respect to other actors are sufficient in building infrastructure-free solutions, thus, not requiring costly calibration, anchor placement, and providing ad-hoc solutions to collision prediction problem. We demonstrate that either sensing the ranges in a network of radio or using standalone sensor, independent and effective collision prediction solutions can be implemented that do not require a central processing hub. As shown in our experiments, sensing is performed via UWB transceivers as an example for the former case, and via a FMCW radar in the latter case. In addition to providing physical safety, we also present new method to provide safety to user's identity while sharing only the location-specific information with a monitor which can further assist us in our local placement or our placement in a network. The main contributions of this dissertation can be divided into two topics, *data* and *algorithms*.

6.2 Data

We have provided a total of four, real-world, one of its kind datasets through this dissertation. All the experiments are conducted in a controlled laboratory setting, with reproducible and multiple levels processing and thorough analysis conducted of the measurements. The datasets mimic real-world scenarios, validating the efficacy of the proposed method against traditional collision warning and machine learning approaches and are

compiled into a labeled repository for public use. We also conducted 2 outdoor data campaigns in densely populated urban areas over a live test-bed, and we developed a labeled database which explored the characteristics of proposed Doppler spread for a fingerprint-based localization system. This additional feature enhances privacy while maintaining effective location fingerprinting.

6.3 Algorithms

In Chapters 2 and 3, we delve into infrastructure-free collision prediction system using the Ultra-Wideband (UWB) signals combined with inertial sensing and offer a cooperative strategy for predicting collisions by utilizing the Multidimensional Scaling (MDS) algorithm. We generate accurate estimates of relative kinematics by using only the noisy, pairwise, inter-node range measurements and node acceleration data. This method, independent of known-location infrastructure, supplements existing technologies by providing precise trajectory prediction robust against indoor multipath effects due to the high-time precision of the UWB radios.

In the second project presented in Chapter 4, Frequency Modulated Continuous Wave (FMCW) radars are proposed for collision prediction in dynamic, cluttered environments. Leveraging range and radial velocity information from radar-Doppler maps (RDMs), a Convolutional Neural Network (CNN) predicts impending collisions. The deep learning framework adapts to dynamic environments, eliminating the need for static filtering, and employs online learning and automated labeling for adaptability and real-time learning.

We further apply these deep learning algorithms in Chapter 5, to introduce a privacy preserving fingerprint-based localization method. The novel proposed fingerprint, which is defined in this thesis as the 'Doppler spread', fills the gap in literature between the usage of the Doppler frequencies and their real-world properties in the multipath environments. These solutions are building on the infrastructure, resources, and progress made via the current trend of learning through data, a direction which is promising for many more innovative products to be built using the concepts and visions pursued in this dissertation.

6.4 Future Work

Despite the real-world implementations and prototypes we have delivered in this thesis, more experiments are needed to fulfill the stringent criteria of real-world applications and attain the level of robustness and reliability expected by industries to facilitate their broader and user-friendly adoptions.

In this respect, for collision prediction, computation performed in the decentralized paradigm of 'edge computing', that brings processing and data storage closer to the location where it is needed, rather than relying on a centralized data center is desired. This further aims to reduce latency, optimize bandwidth usage, and enhance efficiency by processing data locally, at or near the 'edge' of the network, where it is generated. This enables faster decision-making, improved security, and increased scalability for our targeted safety-critical applications working in real-time experiences.

Furthermore, the data-driven approaches would benefit from larger datasets. Expanding the scale of real-world datasets can significantly amplify the effectiveness and accuracy of data-driven approaches. Larger datasets provide a richer and more varying pool of information, which allows algorithms to learn more nuanced patterns and relationships within the data. With increased data volume, machine learning models, presented as the working algorithm for our second and third products, can better generalize to new scenarios and make more reliable predictions or decisions. Moreover, larger datasets can help mitigate overfitting, a common issue where models perform well on training data but poorly on new, unseen data. Additionally, larger datasets enable researchers and practitioners to explore more complex models and techniques, pushing the boundaries of what is possible in various domains. Overall, the availability of larger datasets can catalyze advancements in data-driven approaches, unlocking their full potential for solving real-world problems.

Lastly, for our localization solution presented as the third product, applying a filter, such as Extended Kalman Filter (EKF), cab be applied for handling non-linearities and uncertainties present in the dynamic scenarios, and to track the offenders from the obtained location estimates from our delivered dataset. By continuously updating our estimates based on both model's current output and a dynamic model of the system, the EKF can effectively correct for discrepancies and maintain accurate location estimates even in the

presence of sporadic anomalies. Overall, there are avenues to explore in our learning based approaches, specifically, if not generally as well, to the datasets delivered in this thesis.

References

[1] A. S. Abrar, A. Luong, G. Spencer, N. Genstein, N. Patwari, and M. Minor. Collision prediction from uwb range measurements. *arXiv preprint arXiv:2010.04313*, 2020.

[2] A. S. Abrar, N. Patwari, and J. Decavel-Bueff. Demo abstract: Collision prediction from pairwise ranging. In *19th ACM/IEEE Intl. Conference on Information Processing in Sensor Networks (IPSN 2020)*, April 2020.

[3] M. Akcakaya, S. Sen, and A. Nehorai. A novel data-driven learning method for radar target detection in nonstationary environments. *IEEE Signal Processing Letters*, 23(5):762–766, 2016.

[4] D. Alessandrelli, A. Azzarà, M. Petracca, C. Nastasi, and P. Pagano. Scantraffic: Smart camera network for traffic information collection. volume 7158, pages 196–211, 02 2012.

[5] Y. Almalioglu, M. Turan, C. X. Lu, N. Trigoni, and A. Markham. Milli-rio: Ego-motion estimation with low-cost millimetre-wave radar. *IEEE Sensors Journal*, 21:3314–3323, 2021.

[6] M. Althoff, O. Stursberg, and M. Buss. Model-based probabilistic collision detection in autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 10(2):299–310, 2009.

[7] A. Aravkin, S. Becker, V. Cevher, and P. Olsen. A variational approach to stable principal component pursuit. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, July 2014.

[8] R. M. C. K. Ash, M. On the application of digital moving target indication techniques to short-range fmcw radar data. *IEEE Sensors Journal*, 18(10):4167–4175, 2018.

[9] J. P. Aston, N. Benko, T. Truong, A. Zaki, N. Olsen, E. Eshete, N. G. Luttmer, B. Coats, and M. A. Minor. Optimization of a soft robotic bladder array for dissipating high impact loads: an initial study in designing a smart helmet. In *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, pages 607–614, 2020.

[10] S. Atev, O. Masoud, R. Janardan, and N. Papanikolopoulos. A collision prediction system for traffic intersections. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 169–174, 2005.

[11] A. Banerjee, D. Maas, M. Bocca, N. Patwari, and S. Kasera. Violating privacy through walls by passive monitoring of radio windows. *WiSec 2014 - Proceedings of the 7th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 07 2014.

[12] B. Beck, R. Baxley, and J. Kim. Real-time, anchor-free node tracking using ultra-wideband range and odometry data. *Proceedings - IEEE International Conference on Ultra-Wideband*, pages 286–291, 11 2014.

[13] K. L. Bell, C. J. Baker, G. E. Smith, J. T. Johnson, and M. Rangaswamy. Fully adaptive radar for target tracking part i: Single target tracking. In *2014 IEEE Radar Conference*, pages 0303–0308. IEEE, 2014.

[14] F. Bella and R. Russo. A collision warning system for rear-end collision: a driving simulator study. *Procedia - Social and Behavioral Sciences*, 20:676–686, 2011.

[15] H. M. Bernety, S. Venkatesh, and D. Schurig. Performance analysis of a helmet-based radar system for impact prediction. *IEEE Access*, 6:75124–75131, 2018.

[16] I. Borg and P. Groenen. Modern multidimensional scaling: Theory and applications, volume 2 of statistics in social science and public policy, 1997.

[17] J. Breen, A. Buffmire, J. Duerig, K. Dutt, E. Eide, A. Ghosh, M. Hibler, D. Johnson, S. K. Kasera, E. Lewis, D. Maas, C. Martin, A. Orange, N. Patwari, D. Reading, R. Ricci, D. Schurig, L. B. Stoller, A. Todd, J. Van der Merwe, N. Viswanathan, K. Webb, and G. Wong. Powder: Platform for open wireless data-driven experimental research. *Computer Networks*, 197:108281, 2021.

[18] B. Buchli, F. Sutton, and J. Beutel. Gps-equipped wireless sensor network node for high-accuracy positioning applications. pages 179–195, 02 2012.

[19] R. Chellappa, Gang Qian, and Qinfen Zheng. Vehicle detection and tracking using acoustic and video sensors. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages iii–793, 2004.

[20] C. Chen, A. Markham, N. Trigoni, B. Wang, L. Xie, and P. Zhao. Heart rate sensing with a robot mounted mmwave radar. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2812–2818, 2020.

[21] J. A. Costa, N. Patwari, and A. O. Hero III. Distributed multidimensional scaling with adaptive weighting for node localization in sensor networks. *IEEE/ACM Transactions on Sensor Networks*, 2(1):39–64, Feb. 2006.

[22] D. Daneshvar, C. Nowinski, A. Mckee, and R. Cantu. The epidemiology of sport-related concussion. *Clinics in sports medicine*, 30:1–17, vii, 01 2011.

[23] E. F. Dierikx, A. E. Wallin, T. Fordell, J. Myyry, P. Koponen, M. Merimaa, T. J. Pinkert, J. C. J. Koelemeij, H. Z. Peek, and R. Smets. White rabbit precision time protocol on long-distance fiber links. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 63(7):945–952, 2016.

[24] B. Dil, S. Dulman, and P. Havinga. Range-based localization in mobile sensor networks. pages 164–179, 02 2006.

[25] Ettus. Gpsdo selection guide — ettusknowledgebase, 2016. [Online; accessed 19-April-2024].

[26] L. Fan, Q. Yang, Y. Zeng, B. Deng, and H. Wang. Real-time collision warning and status classification based camera and millimeter wave-radar fusion. In L. Fang, Y. Chen, G. Zhai, J. Wang, R. Wang, and W. Dong, editors, *Artificial Intelligence*, pages 103–114, Cham, 2021. Springer International Publishing.

[27] P. Groenen, R. Mathar, and W. Heiser. The majorization approach to multidimensional scaling for Minkowski distances. *Journal of Classification*, 12(1):3–19, March 1995.

[28] E. Guerrero-Menéndez. Frequency-modulated continuous-wave radar in automotive applications. *Guerrero-Menéndez*, 2018.

[29] K. Haneda, R. Rudd, E. Vitucci, D. He, P. Kyösti, F. Tufvesson, S. Salous, Y. Miao, W. Joseph, and E. Tanghe. Chapter 2 - radio propagation modeling methods and tools. pages 7–48, 2021.

[30] K. G. Harmon, J. A. Drezner, M. Gammons, K. M. Guskiewicz, M. Halstead, S. A. Herring, J. S. Kutcher, A. Pana, M. Putukian, and W. O. Roberts. American medical society for sports medicine position statement: concussion in sport. *British Journal of Sports Medicine*, 47(1):15–26, 2013.

[31] J. Hasch, E. Topak, R. Schnabel, T. Zwick, R. Weigel, and C. Waldschmidt. Millimeter-wave technology for automotive radar sensors in the 77 ghz frequency band. *IEEE Transactions on Microwave Theory and Techniques*, 60(3):845–860, 2012.

[32] T. L. Hayes, N. D. Cahill, and C. Kanan. Memory efficient experience replay for streaming learning. *2019 International Conference on Robotics and Automation (ICRA)*, pages 9769–9776, 2019.

[33] S. Haykin. Cognition is the key to the next generation of radar systems. In *2009 IEEE 13th Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop*, pages 463–467, 2009.

[34] J. C. Hayward. Near-miss determination through use of a scale of danger. *Highway Research Record*, 1972.

[35] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[36] A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. 04 2017.

[37] Z. Hu, Y. Li, G. Jiang, R. Zhang, and M. Xie. Prihorus: Privacy-preserving rss-based indoor positioning. In *ICC 2022 - IEEE International Conference on Communications*, pages 5627–5632, 2022.

[38] J. Jansson. *Collision Avoidance Theory: With application to automotive collision mitigation*. PhD thesis, Linköping University Electronic Press, 2005.

[39] N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intell. Data Anal.*, 6:429–449, 2002.

[40] Y. Kalkan and B. Baykal. Frequency-based target localization methods for widely separated MIMO radar. *Radio Science*, 49(1):53–67, Jan. 2014.

[41] R. Kemker and C. Kanan. Fearnet: Brain-inspired model for incremental learning. *ArXiv*, abs/1711.10563, 2018.

[42] W. Kim, H.-W. Cho, J. Kim, B. Kim, and S. Lee. Yolo-based simultaneous target detection and classification in automotive fmcw radar systems. *Sensors (Basel, Switzerland)*, 20, 2020.

[43] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

[44] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas. Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, 30:25–36, 11 2005.

[45] B. Kusy, A. Ledeczi, and X. Koutsoukos. Tracking mobile nodes using rf doppler shifts. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, SenSys '07, page 29–42, New York, NY, USA, 2007. Association for Computing Machinery.

[46] V. Lakshmanan, S. Robinson, and M. Munn. *Machine Learning Design Patterns*. O'Reilly Media, 2020.

[47] D. Lee and H. Yeo. Real-time rear-end collision-warning system using a multilayer perceptron neural network. *IEEE Transactions on Intelligent Transportation Systems*, 17, 2016.

[48] D. Lewis, J. Catlett, W. Cohen, and H. Hirsh. Heterogeneous uncertainty sampling for supervised learning. 1996.

[49] J. Lien, N. E. Gillian, M. E. Karagozler, P. Amihood, C. Schwesig, E. Olson, H. Raja, and I. Poupyrev. Soli: ubiquitous gesture sensing with millimeter wave radar. *ACM Trans. Graph.*, 35:142:1–142:19, 2016.

[50] C. X. Lu, S. Rosa, P. Zhao, B. Wang, C. Chen, J. A. Stankovic, N. Trigoni, and A. Markham. See through smoke: robust indoor mapping with low-cost mmwave radar. *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, 2020.

[51] C. X. Lu, M. R. U. Saputra, P. Zhao, Y. Almalioglu, P. P. B. de Gusmão, C. Chen, K. Sun, N. Trigoni, and A. Markham. milliego: single-chip mmwave radar aided egomotion estimation via deep sensor fusion. *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, 2020.

[52] A. Luong, P. Hillyard, A. S. Abrar, C. Che, A. Rowe, T. Schmid, and N. Patwari. A stitch in time and frequency synchronization saves bandwidth. In *ACM/IEEE Intl. Conference on Information Processing in Sensor Networks (IPSN 2018)*, pages 96–107, April 2018.

[53] N. G. Luttmer, T. E. Truong, A. M. Boynton, D. Carrier, and M. A. Minor. Treadmill based three tether parallel robot for evaluating auditory warnings while running. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9135–9142, 2020.

[54] M. Magno, A. D'Aloia, T. Polonelli, L. Spadaro, and L. Benini. Shelmet: an intelligent self-sustaining multi sensors smart helmet for bikers. In *International Conference on Sensor Systems and Software*, pages 55–67. Springer, 2016.

[55] E. Mazomenos, D. Jager, J. Reeve, and N. White. A two-way time of flight ranging scheme for wireless sensor networks. volume 6567, pages 163–178, 02 2011.

[56] M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989.

[57] NaturalPoint. Motion capture systems - optitrack webpage.

[58] Optitrack. Motion capture systems, 2022.

[59] T. Rahman, A. T. Adams, R. Vinisha, M. Zhang, S. N. Patel, J. A. Kientz, and T. Choudhury. Dopplesleep: a contactless unobtrusive sleep sensing system using short-range doppler radar. *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2015.

[60] R. T. Rajan, G. Leus, and A.-J. van der Veen. Relative kinematics of an anchorless network, 2018.

[61] M. K. A. M. Rasli, N. K. Madzhi, and J. Johari. Smart helmet with sensors for accident prevention. In *2013 International Conference on Electrical, Electronics and System Engineering (ICEESE)*, pages 21–26. IEEE, 2013.

[62] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5533–5542, 2017.

[63] S. R. Rupanagudi, S. Bharadwaj, V. G. Bhat, S. Eshwari, S. Shreyas, B. S. Aparna, A. Venkatesan, A. Shandilya, V. Subrahmanya, and F. Jabeen. A novel video processing based smart helmet for rear vehicle intimation & collision avoidance. *2015 International Conference on Computing and Network Communications (CoCoNet)*, pages 799–805, 2015.

[64] B. Settles. Active learning literature survey. *Science*, 10(3):237–304, 1995.

[65] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz. Localization from mere connectivity. In *Proc. 4th ACM Intl. Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '03, page 201–212, 2003.

[66] A. Singh and N. Patwari. Collision prediction using uwb and inertial sensing: Experimental evaluation. In *2021 IEEE International Conference on Autonomous Systems (ICAS)*, pages 1–5, 2021.

[67] A. Singh and N. Patwari. Range-based collision prediction for dynamic motion. In *2021 IEEE 18th Annual Consumer Communications Networking Conference (CCNC)*, pages 1–6, 2021.

[68] W. Song, Y. Yang, M. Fu, F. Qiu, and M. Wang. Real-time obstacles detection and status classification for collision warning in a vehicle active safety system. *IEEE Transactions on Intelligent Transportation Systems*, 19:758–773, 2018.

[69] J. A. Stratton. *Electromagnetic theory*. McGrow-Hill Book Company Inc., 1941.

[70] H. Thornburg. Adafruit bno055 absolute orientation sensor, 2022.

[71] T. Tieleman, G. Hinton, et al. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[72] W. S. Torgerson. Multidimensional scaling of similarity. *Psychometrika*, pages 379–393, 12 1965.

[73] UserManual.wiki. Ancortek manual, Apr 2016.

[74] J. Vargas, S. Alsweiss, O. Toker, R. Razdan, and J. Santos. An overview of autonomous vehicles sensors and their vulnerability to weather conditions. *Sensors*, 21(16), 2021.

[75] C. Villien, V. Fleck, E. Ostertag, and P. Raymond. 3-d short-range localization device by low-cost cw-doppler radar. In *IEEE International Radar Conference, 2005.*, pages 557–561, 2005.

[76] A. Viquerat, L. Blackhall, A. Reid, S. Sukkarieh, and G. Brooker. Reactive collision avoidance for unmanned aerial vehicles using Doppler radar. In C. Laugier and R. Siegwart, editors, *Field and Service Robotics: Results of the 6th International Conference*, pages 245–254. Springer Berlin Heidelberg, 2008.

[77] X. Wang, J. Liu, T. Qiu, C. Mu, and P. Zhou. A real-time collision prediction mechanism with deep learning for intelligent transportation system. *IEEE Transactions on Vehicular Technology*, PP:1–1, 06 2020.

[78] K. Webb, S. K. Kasera, N. Patwari, and J. K. V. der Merwe. WiMatch: wireless resource matchmaking. 2021.

[79] P. Wei, L. Cagle, T. Reza, J. Ball, and J. Gafford. Lidar and camera detection fusion in a real time industrial multi-sensor collision avoidance system, 2018.

[80] Z. Yang and K. Järvinen. The death and rebirth of privacy-preserving wifi fingerprint localization with paillier encryption. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, pages 1223–1231, 2018.

[81] Z. Yang, Z. Zhou, and Y. Liu. From rssi to csi: Indoor localization via channel response. *ACM Comput. Surv.*, 46(2), dec 2013.

[82] H.-S. Yeo, G. Flamich, P. Schrempf, D. Harris-Birtill, and A. Quigley. Radarcat: Radar categorization for input & interaction. pages 833–841, 10 2016.

[83] J. Zhang, L. Chen, C. Wang, L. Zhuo, Q. Tian, and X. Liang. Road recognition from remote sensing imagery using incremental learning. *IEEE Transactions on Intelligent Transportation Systems*, 18(11):2993–3005, 2017.

[84] P. Zhao, C. X. Lu, J. Wang, C. Chen, W. Wang, A. Trigoni, and A. Markham. mid: Tracking and identifying people with millimeter wave radar. *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 33–40, 2019.

[85] Z. Zhao, L. Zhou, Q. Zhu, Y. Luo, and K. Li. A review of essential technologies for collision avoidance assistance systems. *Advances in Mechanical Engineering*, 9:168781401772524, 10 2017.