



Bachelor Thesis

Computer Science and Engineering 300 credits

Computer Engineer 180 credits

Here I go: A prediction model for
e-bike and e-scooter positioning in-
side a CCAM environment

Computer Science and Engineering 15 credits

Halmstad June 18, 2024

Ruben Croall

Douglas Jonsson Lundqvist

Abstract

This thesis presents a prediction model for e-bikes and e-scooters, aimed at enhancing traffic safety and efficiency by sharing their intentions of future possible positions among road users. The research addresses the current automated vehicle technologies which lack communication between road users. The prediction model is based on and tested with a mobility model, adapted for modelling e-bikes and e-scooters in a simulator program primarily used for pedestrians. This implementation has produced the ability to predict future positions and further the development of intention-sharing capabilities in urban traffic scenarios. The model is built upon physical parameters and mathematical models for a controlled and regulated model. Polynomial regression was applied to predict positions based on historical data and the results were evaluated with RMSE metrics, demonstrating the prediction accuracy in different scenarios. The thesis also includes the integration of the prediction model into a hardware setup, a Raspberry Pi. Demonstrating the practical application and retaining the effectiveness of the model in a real-time environment. Gathered from the results, the model can reserve a predicted area every second but also has the capability to work during faster or slower time intervals, depending on the hardware used to enable the model in the protocol. With this, the research highlights the possibility of implementing this in CCAM systems. The results show promising accuracy with a simple controlled model using as little necessary data as possible. The project work contributes to the field of intelligent transport systems by providing a scalable solution to enhance the interaction between VRUs and vehicles, creating a step closer to achieving the Vision-Zero goal of having zero traffic-related accidents or fatalities.

Sammanfattning

Denna rapport framför en prediktionsmodell för elcyklar och e-scooters som siktar på att öka säkerheten och effektiviteten i trafiken genom att dela väganvändares intentioner och reservera framtida möjliga positioner. Arbetet tar upp den nuvarande teknologin som används för automatiserade fordon och dess brister i kommunikation mellan väganvändare. Prediktionsmodellen baseras på -och testas med en mobilitetsmodell som justerats för att modellera elcykel -och elscooter användare i ett simulationsprogram framtaget främst för att simulera fotgängar-dynamik. Denna implementation har öppnat möjligheten att förutspå framtida positioner och att vidare utveckla intentionsdelning möjligheter i stadstrafikmiljö. Modellen är uppbyggd på fysikaliska parametrar och matematiska modeller för en kontrollerad och reglerad model. Polynomregression har applicerats för att förutspå positioner baserat på föregående positions-data.

Resultaten utvärderas med RMSE som mätetal, vilket ska demonstrera förutsägens precision i olika scenarion. I rapporten ingår även integration av denna prediktions-modell i hårdvara, nämligen i en Raspberry Pi. Detta ska demonstrera den praktiska applikationen och samtidigt bibehålla effektiviteten av modellen i en realtids miljö. Från resultaten kan modellen reservera ett förutsagt område varje sekund, den kan också arbeta under snabbare eller långsammare tidsintervall beroende på hårdvara och dess syfte. Med detta betonas forskningens möjlighet att implementera modellen i ett CCAM system. Projektarbetet bidrar till forskningen inom intelligenta transportsystem genom att framföra en skalbar lösning för att förbättra interaktionen mellan VRUs och tyngre fordon, vilket för oss ett steg närmare Nollvisionens mål om att ingen ska skadas allvarligt eller omkomma i trafiken.

Acknowledgements

We would like to extend our deepest gratitude to the people behind Vadere for their amazing efforts put into pedestrian behaviour models, enabling our research to reach the project goal, and most of all, our supervisor, Oscar Amador Molina, for his guidance, support and constructive feedback throughout the project work.

Acronyms

| | |
|-------------|---|
| AD | Automated Driving |
| APFP | Area of Possible Future Positions |
| AV | Automated Vehicle |
| CCAM | Connected Cooperative and Automated Mobility |
| DP | Data point |
| ETSI | European Telecommunications Standards Institute |
| GNM | Gradient Navigation Model |
| ITS | Intelligent Transportation System |
| P2X | Pedestrian-to-everything |
| RMSE | Root Mean Squared Error |
| RU | Road User |
| V2X | Vehicle-to-everything |
| VAM | Vulnerable road user Awareness Message |
| VRU | Vulnerable Road User |

Nomenclature

| | |
|----------------|---|
| α/β | Polynomial coefficient(s) |
| Λ | Accuracy |
| A/B | Edge point(s) |
| C | Actual position |
| P | Predicted position |
| R | Rotation matrix |
| \mathcal{C} | Varied coordinate system |
| \mathcal{I} | Inertial coordinate system |
| ω | Angle between $\overline{\mathbf{CP}}$ vector and x -axis |
| θ | Possible turn angle/search angle |
| i | Current time step |

Contents

| | | |
|----------|------------------------------------|-----------|
| 1 | Introduction | 1 |
| 1.1 | Purpose | 2 |
| 1.2 | Problem statement | 3 |
| 1.3 | Boundaries and requirements | 3 |
| 2 | Background and related work | 5 |
| 2.1 | Intention detection | 6 |
| 2.2 | Intention sharing | 6 |
| 2.3 | Mobility model | 7 |
| 3 | Materials and method | 9 |
| 3.1 | Methodology | 10 |
| 3.2 | Hardware implementation | 12 |
| 3.3 | Result analysis | 13 |
| 4 | Results | 15 |
| 4.1 | Prediction Model | 15 |
| 4.2 | Embedded system | 20 |
| 5 | Discussion | 23 |
| 5.1 | Limitations | 24 |
| 5.2 | Existing studies | 25 |
| 5.3 | Future work | 25 |
| 5.4 | Scalability | 27 |
| 5.5 | Machine learning | 27 |
| 6 | Conclusion | 31 |
| A | Appendix A | 1 |
| A.1 | Figures and plots | 1 |
| B | Appendix B | 7 |
| B.1 | Equations | 7 |

Introduction

In recent years of developing advanced vehicle capabilities, more and more car manufacturers have turned their eye towards AD. It depends on analysing the surroundings using built-in sensors to promote and enhance safer driving [1]. However, the current sensors made today rely solely on detecting what happens in the current environment and they only respond in real-time when an accident is about to occur [2]. For example, suppose they had the function to detect cyclist behaviour and their intentions. In that case, studies have proven results that can significantly decrease the number of vehicle-to-cyclist accidents by increasing the accuracy of how visual cues can be correctly interpreted [3] [2] [4]. In the year 2021, according to police and emergency medical care data collected by the Swedish transport agency, 7928 mild injuries, 3875 medium injuries, and 504 severe injuries were reported for bicycles alone, where of which 23 were lethal [5]. These numbers must be reduced.

Recent studies suggest that enhanced secure driving can be improved by enabling a communication model between e-bikes, e-scooters and vehicles in the Connected Cooperative and Automated Mobility (CCAM) environment [1]. This is a standard communication area that ETSI strive to implement through the Intelligent Transportation System (ITS) [6]. If a similar communication model is implemented, the goal of reaching zero traffic-related serious accidents or deaths, thus reaching the goal of the Vision-Zero initiative set by the EU, could be achieved by the year 2050 [7]. Although substantial steps have been taken regarding AD, there remains a problem in addressing the safety of the VRUs i.e. cyclists, pedestrians etc. Currently, the information that vehicles broadcast to other vehicles is what the built-in sensors pick up from the nearby environment [8]. The current e-bikes and e-scooters produced do not possess either sensors or the smallest communication model that transmits their intentions.

This project work introduces a prediction model specifically designed for e-bikes and e-scooters within a CCAM environment. The significance of this work lies in its ability to enhance traffic safety and efficiency by predicting the future positions of these VRUs. By integrating a mobility model with polynomial regression and implementing it on a hardware setup like a Raspberry Pi, this research provides a scalable solution for intention sharing. This advancement is a critical step towards achieving the Vision-Zero goal of eliminating traffic-related fatalities and serious injuries, highlighting its potential impact on the development of intelligent transportation systems.

The paper is structured as follows: Section 2 includes background, where related areas are explored regarding the project and what conclusions have been drawn based on previous studies. The current state-of-the-art is presented and existing gaps that the project works to fill, are identified. Section 3 navigates through the methodology of this project which presents the prediction model and the embedded system which allows VRUs, explicitly e-bikes and e-scooters, to communicate their intentions to a nearby environment involving vehicles and other RUs. The mentioned system will be developed through iterative stages of phases that include concept development, adjustment, testing, and hardware implementation. Numerous simulations have been done through simulation software that granted continuous data-driven improvements from different simulations for testing and optimising the system. Sections 4 and 5 highlight the results, discussing how the results are looked at, analysed, and interpreted, and what the results have achieved regarding the background. It is also analysed and compared to the project goals and discusses what improvements can be made. Additional discussion on why certain choices were made, how they affect the project objectives, current limitations and what future work can be made are present. Finally, followed up by the conclusion in section 6 of this report.

1.1 Purpose

In 2018, a study by the U.S. Department of Transportation analysed accidents involving two million drivers over a span of two and a half years and found that an estimated 41% of these incidents could be traced back to errors in recognition, while about 33% were due to errors in decision-making [9]. To minimise the number of accidents happening because of less traffic awareness from vehicles of the current whereabouts of VRUs, the car manufacturer Audi has developed a similar system in the same field as this project, Car-to-Everything (Car2X) and Cellular V2X communication. Their models allow cars to communicate with other road users and inform the car when they are in the vicinity, preventing traffic accidents from occurring [10].

Our project aims to create a system that will be able to, through information about position changes, and velocity, predict an area of possible future positions (APFP) where an e-bike/e-scooter might find themselves positioned in, and reserve this space. With this knowledge, traffic accidents can be avoided by communication between VRUs and other RUs through intention sharing, meaning they share their reserved space so other users are aware. The prediction model in the intention-sharing system will be tested on data from simulations with the mobility model called the Gradient Navigation Model (GNM), which accurately

simulates pedestrian movement and their intentions. The manoeuvres coordination of a pedestrian is assumed to be based on decisions rather than physical contact force [11]. After the system is designed, the next step is to implement it on a single-board computer and simulate a GPS interval of one second. In the following section, the project's goals are introduced and what is expected to be achieved in the current state of the area.

1.2 Problem statement

The current technology for detecting other entities in traffic for AV consists of built-in sensors that respond to real-time events in the nearby surroundings. In the area of CCAM, this is known as intention detection. For AVs, detection and reaction is the only form of communication between VRUs and users that are less at risk. The problem with this kind of communication is it solely relies on visual indicators. This flaw in vehicle technology impacts the general safety concerning integrating AVs into society and will slow down the goal of the Vision-Zero initiative. Sharing the intentions of going somewhere in a given radius in traffic is not implemented due to limitations bound by safety reasons, such as; minimum latency and maximum accuracy requirements for messages, and which mobility model is best suited for all RUs [1].

This project aims to address this problem by creating a prediction model, specifically using e-bike and e-scooter dynamics in crowd-behaviour simulations, to predict their next position and reserving an area that in turn could expand on the possibility of sharing intentions among other RUs in a CCAM environment, reducing the risk of accidents and fatalities in traffic. In the section that follows, the boundaries and requirements are introduced regarding the project work and milestones are set to be achieved.

1.3 Boundaries and requirements

The project is split up into two main goals: designing a prediction model for e-bikes and e-scooters and implementing this on a protocol to enable intentions to be shared through a single-board computer. Initially, a GPS was expected to be mounted on the single-board computer, however, due to time restrictions and material complications, this was not done. Instead, the prediction model will read pre-simulated output data and run as though a GPS was mounted on it. The first goal is the primary one to achieve given it is necessary to advance to the next. Otherwise, the design of a protocol to transmit the predicted intentions from the model cannot be made.

Several milestones have been set to achieve the design and implementation of the prediction model:

1. Define the parameters for e-scooters/e-bikes in the simulator program.
2. Set up data output and simulate different and qualitative scenarios to measure.
3. Create and calculate how to predict future positions with the data output from simulation variables.
4. Figure out how many data points are needed to achieve a prediction error of less than 4 meters in different scenarios.
5. Adjust the model based on the results from more simulations.
6. Design a protocol onto hardware that can use the model and work with data in real-time.

The boundaries within this first part of the project are set to maintain a high degree of control over the model and its values. Rather than using a machine learning program to train and give sequential data from different scenarios, proper control over the program is maintained, making it possible to predict in several typical scenarios first and make it as accurate as possible with each simulation. Additionally, to ensure compatibility with embedded systems and facilitate VAM generation, the model should operate efficiently with limited data.

Also, according to ETSI standards [12, p. 46], the current and the predicted positions should be within the absolute distance of 4 meters between each other. This is viewed as adequate accuracy in traffic scenarios according to ETSI VAM generation standards. Any predictions outside this limit are seen as inaccurate and not suited to be a prediction as well as it is outside the prediction zone to generate VAMs. This could be interpreted in a real-life scenario as a prediction that would be outside of your lane. To consider a prediction safe, confidence to be less than 4 meters from the actual position should be around 95%.

Background and related work

Understanding and predicting human behaviour, especially in the context of mobility and traffic safety, is a complex yet critical area of study. The shared environment between vehicles, e-bikes and e-scooters creates a demand for advanced methods for the current technology intention detection, sharing and mobility modelling in traffic. This thesis project delves into innovative research and provides insight into these three pivotal areas, offering insights into the latest methodologies and their implications for improving traffic dynamics and safety.

In the area of intention detection, innovative approaches to read and anticipate a RUs next move are explored. Through the researched studies, it is delved into how kinematic information and visual cues from cyclists, such as pedalling behaviour and their perception of incoming vehicles, play a crucial role in predicting future positions. Furthermore, the investigation of advancements in cyclists' movements through a weighted distribution of forecasts that account for uncertainties enhances prediction accuracy. The research extends to vision-based methods for detecting intentions via taillights and neural networks.

Intention sharing has started to emerge as a vital component in CCAM, focusing on the efficient transmission of intended actions between vehicles and VRUs. From the gathered studies, the effectiveness of different communication methods is examined, from broadcast methods to decentralised coordination to impact Take-Over-Requests (TORs) in AD systems on driver behaviour. The research highlights the importance of hybrid communication architectures for safety and the potential of V2X communications in protecting VRUs, showcasing the progress made towards enhancing vehicular cooperation and safety.

Lastly, the mobility model section underscores the importance of simulating crowd behaviour and managing pedestrian movement within shared spaces. The GNM model illustrates force-based to intention-based models, offering a more nuanced understanding of crowd dynamics. This approach not only simplifies the mathematical modelling required but also provides a more accurate representation of how individuals navigate crowded environments, considering personal space, reaction time and the influence of surrounding pedestrians and objects.

2.1 Intention detection

The method of recognizing and forecasting a person's actions through analysis of present behaviour, situational factors and other relevant indicators. In one of the articles researched, the authors used kinematic information about interacting road users, with cyclist visual cues. The two visual parameters of pedalling behaviour and the cyclist's view were effective in predicting what the next position of the cyclist was going to be. Their results showed that visual intentions were important in predicting the cyclist's intent [4].

Another study proposes a new method for the prediction of cyclist movements. The method is based on forecasting positions in the form of weighted distributions using a group of forecasts that considers uncertainties. Their method was compared against an existing one without uncertainties, they reduced reliability error by 85% [3].

Besides cyclist behaviour and intentions, there is also a possibility to look at tail-lights and the intention of turning through them. One article proposes a new vision-based method for detecting vehicle tail light intentions, with an enhanced network protocol for transmission. The results they got proved to have higher accuracy in detecting taillight intentions compared with using standard network protocols [13].

Compared to that, one study proposes a system that anticipates the next position of a cyclist through the movement of the head and body when they are about to turn using neural networks. Numerous experiments were done which led to a model consisting only of heat-map images, these yielded the best accuracy in the estimation of the cyclist's next movement [2]. Even though the focus lies on intentionally sharing and transmitting that information between relevant users in traffic, some parts of this research provide insight into the testing of the prediction model in real time.

In section 2.2, the advancement to intention detection is introduced, intention sharing, and what new possibilities it offers to traffic safety.

2.2 Intention sharing

The transmission of an individual's or a system's intended actions or objectives to others within a CCAM context. In one article regarding this, the author's approach to the possibility of manoeuvre coordination goes into detail regarding implicit and explicit forms of communication. The authors bring up four different types of broadcast methods to transmit messages for decentralised coordination in lane-merging scenarios between vehicles. The conclusion of this was that these

methods can make it possible to enable safe and efficient manoeuvre coordination with a roadside unit (RSU) [8].

Another study brings attention to evaluating driver responses to TORs in AD systems during different scenarios, initially focusing on how the timing of the TOR affects driver behaviour in traffic. Their results suggest that a TOR in an AD system can mitigate the risk of critical situations and may be even safer than regular driving [14].

In one similar study, the authors provide a summary of a survey that examines progress and obstacles in cooperative manoeuvres among automated vehicles improved through greater vehicle connectivity. Their study indicates that for safety communications, hybrid communication architectures are more effective than relying on a single type of communication [1].

Regarding communication methods in the area of intention sharing, one article highlights the enhancement and protection of VRUs using Pedestrian-to-Anything (P2X) communications, particularly through the use of the Vulnerable Road User Awareness Message (VAM) as standardised by the ETSI. The findings suggest that adopting a less aggressive approach to message triggering can enhance the effectiveness of standalone VAM [15].

In this project work, the main focus is to create and design the connected prediction model and reach a point where cooperation will be relevant to present along with it. The articles mentioned above are suitable for suggesting a communication method for our model to implement along with the next part of the project. However, to share these VRU's intentions, it is crucial to understand how they behave in traffic. This concept is explained in section 2.3.

2.3 Mobility model

The mobility model is a mathematical model used to simulate crowd behaviour, based on pedestrians' movements. The prediction model foundation will be based on the GNM model, where research has been done to simulate crowd behaviour based on intentions, and not physical force [11]. While the force models usually require four different Ordinary Differential Equations (ODE) [16], this model only needs three to describe the movement and navigation of one pedestrian. This is achieved by narrowing down the complexity of pedestrian navigation through some assumptions;

- Crowd dynamics are driven by individual choices rather than physical interaction.

- People aim to arrive at their destination as quickly as possible, with knowledge about their surroundings.
- The presence of other pedestrians and objects influences their preferred speed, which they adjust after a certain reaction time.

They also take into account reaction time and personal comfort zone as well as avoid pedestrians standing on top of each other. The model very well demonstrates different crowd scenarios such as; bottlenecks, lane formations, stop-and-go waves, and the speed-density relation [11].

This model provides a good base for the research of this project, where the dynamics can be altered but still maintain the behavioural aspect of crowd dynamics since e-bike and e-scooter users follow the same mindset as pedestrians concerning personal space and comfort zones.

Materials and method

This section highlights the methods used in the project to gather the results needed for reaching the project goal, how the results were analysed to further improve the model, and the implementation of the protocol into an embedded system. Below is a flowchart that outlines the steps taken in the methodology.

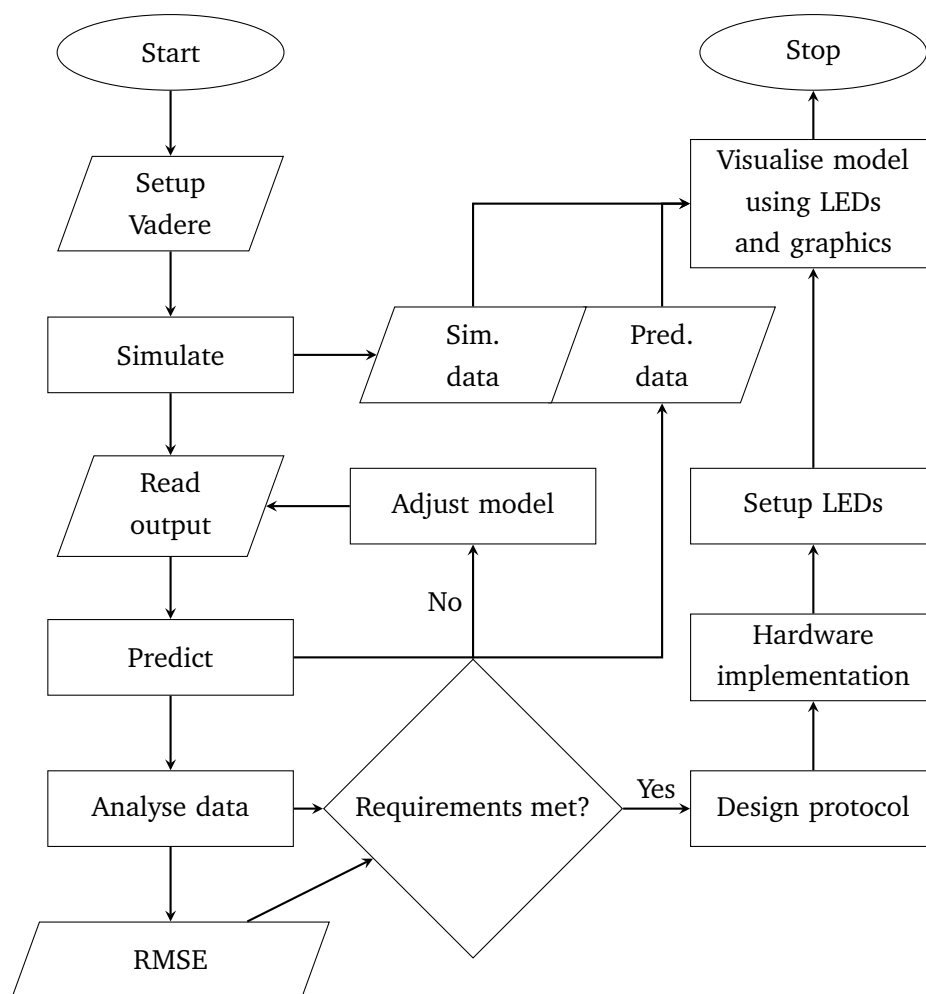


Figure 3.1: Flowchart showing the steps taken to create the prediction model, adjust it until requirements are met, and the hardware implementation.

3.1 Methodology

To create and test the prediction model, data was gathered from simulations in the Vadere crowd behaviour simulation software [17]. This simplified the progress significantly because hundreds of VRUs can be looked at in custom-made scenarios, and their output data is easy to access and adjust. To more accurately mimic e-bikes and e-scooters in Vadere, variables for acceleration, minimum, -and maximum speed, speed distribution mean, and standard deviation were adjusted, based on studies done on e-bikes [18]. Additionally, to reenact VRU behaviour, entities (VRUs) were randomly spawned in a given starting area where each would travel towards a target area, creating a unique individual path for each VRU. Furthermore, the time between each time step i was set to 0.9s to as accurately as possible simulate data gathered from a GPS that reads data at 1 Hz (Vadere can only have a time between time steps of $< 1s$). This is in line with the recommended sampling rate for generating VAMs, although a shorter time between steps could generate more accurate predictions, high sampling rates can cause collisions in the wireless medium [15].

Before simulating, work had to be done on developing a program in Java¹ which could interpret the data output from the simulations. This program can, through reading .txt files, define arrays that contain VRUs positional data (VRU ID, x -, and y value, velocity, time step). This will be crucial for simplifying the development and testing of our model. Additionally, functions were defined to fetch the Euclidian distance between two points, which can later be used to calculate the RMSE. To predict future positions, two different approaches were tested. Firstly, only the current position \mathbf{C}_n , and the previous position \mathbf{C}_{n-1} were looked at, and through fetching velocities at these positions, while also knowing the time between each time step, the velocity on the x -axis, v_x , and one for the y -axis, v_y could be calculated. Then, by calculating the angle between the x -axis and the vector between these positions $\overline{\mathbf{C}_n \mathbf{C}_{n-1}}$, we could predict a future position \mathbf{P} following the same angle and the velocity given by v_x and v_y . However, since we are only looking at two data points (DP) this model was only accurate at predicting scenarios where we are going straight, and not turning.

The second approach, and more promising, was using polynomial regression [19]. This way, we could choose to look at more than just two DPs to get a more accurate prediction. Our future position on the x -axis and y -axis could then be expressed as

$$x_i(t_i) = \alpha_0 t_i^0 + \alpha_1 t_i^1 + \alpha_2 t_i^2 + \dots + \alpha_n t_i^n, \quad (3.1)$$

¹<https://github.com/Rubbaducky95/Prediction-Model—Bachelor-thesis.git>

and

$$y_i(t_i) = \beta_0 t_i^0 + \beta_1 t_i^1 + \beta_2 t_i^2 + \dots + \beta_n t_i^n. \quad (3.2)$$

Where

$$i = 1, 2, 3, \dots, m \quad (3.3)$$

Where n is the degree of the polynomial, the number of data points looked at is $DP = n + 1$, and m is the number of time steps for that VRU. Since the time interval between the time steps is known, t_i is also known (where i is the next time step). The values for α and β are then solved through matrix multiplication (B.1, B.2), using the Apache common maths library in Java [20] where the data points are first added to a list of weighted observed points and then fitted to the polynomial [21], the points are treated with the same default weight (1.0), although this could be subject to change. This will in turn provide a predicted position with a x -, and y -value.

Since the output from Vadere generates all the actual positions and the prediction should be made like the next position is not known, the Java program² reads the positions discretely, by first giving the prediction method an array of size 1, then 2, and so on until m , where m is the number of positions a VRU has in a scenario. The prediction method uses the last $n + 1$ number of elements in that array depending on how many data points we choose to look at.

With the predicted positions from different scenarios, a VRU predicted path;

$$PredictedPath_{VRUx} = \{\mathbf{P}_1, \mathbf{P}_2, \dots, \mathbf{P}_n, \mathbf{P}_{n+1}\}, \quad (3.4)$$

and actual path;

$$ActualPath_{VRUx} = \{\mathbf{C}_0, \mathbf{C}_1, \dots, \mathbf{C}_n\}, \quad (3.5)$$

where n is the number of time steps, will be looked at independently to calculate their RMSE,

$$RMSE = \sqrt{\sum_{i=1}^n \frac{|\mathbf{C}_i \mathbf{P}_i|^2}{n}}, \quad (3.6)$$

resulting in one RMSE value for each VRU in a simulated scenario. The amount of data points looked at is then incremented to repeat the process. This process will be done for each scenario to compare and figure out how many data points should be looked at when the VRU is facing a variety of different situations.

²<https://github.com/Rubbaducky95/Prediction-Model—Bachelor-thesis.git>

We plan to also look at changes in speed and orientation to decide if the number of data points we look at should be incremented or decremented. ETSI states that we should look for a change of either 0.5 m/s, a distance of 4 metres, or a change in angle of 4 degrees when broadcasting a VAM in a CCAM environment [12, p. 46]. This occurrence will be surveilled, and the amount of data points we will look at will be adjusted. We will need to try different methods of observing and adjusting and decide what works best for our model. These adjustment types of the polynomial model will be tested against each other by looking at how their respective errors are distributed.

When adjustments were finished and the predictions were sufficiently accurate, the APFP was implemented into the model simply by looking at the difference in angle from the current and previous position, and using that same angle θ on both sides of the vector between **C** and **P** to calculate the edge-points **A** and **B** of a partial circle with origo at **C** (See appendix B, B.3-B.8). This predicted angle value could be further improved by implementing it to become velocity-, and acceleration-dependant (further discussed in section 5.2). These points would make out a polygon when lines are drawn from **C**, to the first edge point **A**, through **P**, to the second edge point **B**, and back to **C**. This polygon/partial circle will make out the APFP the VRU reserves. In section 4.2, it is explained how the current model was implemented in an embedded system and how it visually demonstrates the model's functionality.

3.2 Hardware implementation

During the planning of the thesis work, it was initially thought that the second part of the project would consist of implementing the model onto a single-board computer to make it work with the GPS. However, due to hardware issues, instructions were given instead to simulate the GPS and the update frequency interval of one second. After the prediction model was done, the work was started to implement it on a Raspberry Pi 5 [22]. This was chosen over other single-board computers, for example, an Arduino, because it offers a more dynamic and versatile development environment. Making it possible to include multimedia access, media processing and display interfaces. Problems arose when trying to convert the code from Java to Python. The library for the polynomial regression model does not have the same functionalities in Python as in Java. If it were to work in Python, the main part of the prediction system and how the polynomial model behaves would need to be changed in the way it gathers previous data. The Java Development Kit (JDK) was installed for streamlined implementation and seamless access to predicted positions and reserved areas on the Raspberry Pi. After the

model was operational, five LEDs were connected. Three were ultra-bright LEDs representing the size of the area region with various intensities depending on how sharp certain turns are. The two other LEDs were used to tell when it turned left or right. A script was made in Python for this functionality along with drawing the predicted positions and the area for each position.

concluding the methodology chapter, the section 3.3 analyses the results of different parts of the model.

3.3 Result analysis

To get a good estimate of the prediction accuracy, the Euclidean distance between the actual positions \mathbf{C}_i , and the predicted positions \mathbf{P}_i , resulting from our simulations and prediction model was calculated. Then, the use of the RMSE from these distances derived from simulations, and the incrementation of the number of data points (DPs) we are looking at, provides a good picture of how many DPs should be looked at in different scenarios. The closer to 0 the RMSE is, the better the overall predictions are for that VRU.

It was estimated that while we have a constant velocity, or if we are accelerating, we are going straight, and therefore 2 DPs should be sufficient. On the other hand, if we are decelerating, we assume we might turn, and therefore should look at more than 2 DPs, but not too many. This upper limit will be found by looking at simulations from a scenario in which a turn is happening, and testing it for different amounts of data points. The use of RMSE is validated by the fact that our prediction errors are assumed to be normally distributed [23], as per the fact that the polynomial regression model we use to predict is assumed to produce independent, normally distributed errors, with mean zero and constant variance [24].

When the predicted position has been generated and the edge points $\mathbf{A}_{\mathcal{I}}$, and $\mathbf{B}_{\mathcal{I}}$ calculated to create an APFP, the accuracy of the predictions can be measured by checking if the next actual position is inside the previous APFP. This was done through ray casting, where a point is generated outside the APFP a line is drawn from that point to the point to be checked, and the number of intersections with the APFP's edges is counted. If the number is odd, we are inside the APFP, if it is even, the point is outside of the APFP [25].

By then counting the number of times a prediction is inside the APFP throughout a whole scenario, an overall accuracy can be calculated by dividing the number of successful predictions by the overall number of positions for that VRU.

Results

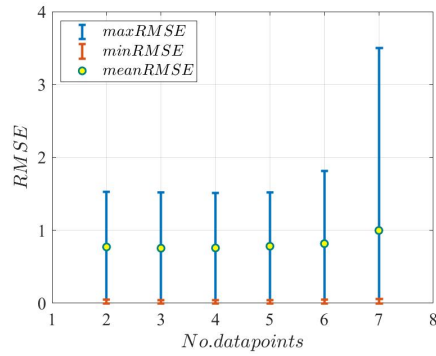
Using the methods described in section 3.1 this chapter describes the outcome of simulating, adjusting, and testing the prediction model up until the point where it is applicable to a protocol implemented onto hardware as described in section 3.2.

4.1 Prediction Model

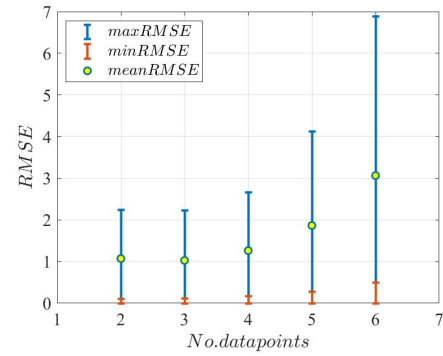
During the first part of the project work, the setup to define pedestrians as e-scooters/e-bikes entities in the Vadere simulation software was managed by basing the dynamic variables off of previous studies done on e-bikes behaviour in traffic [18]. The possibility of gathering data outputs from the scenarios was also managed for the different scenarios (see Appendix A) used to simulate VRU behaviour. With this data, we have defined a prediction model that calculates the next position with the assistance of modelling using polynomial regression.

The errors (distances between actual, -and predicted position) were calculated and the RMSE was utilised to get an understanding of the accuracy of the model. As mentioned in the methodology section, normal distribution validates the use of the RMSE.

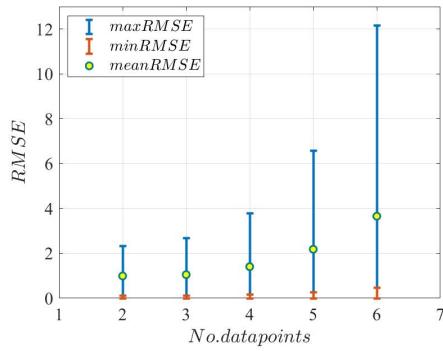
Using a set amount of data points (DPs) in the model will show what amount should be chosen in the different scenarios simulated. Figure 4.1 shows the result of simulating five different scenarios with 100 VRUs in each, where their paths are looked at individually, their RMSE is calculated, and the mean-, maximum-, and minimum RMSE is found among all of them, starting with 2 DPs up to 6 or 7 DPs. Afterwards, when the amount of DPs that should be used when detecting or predicting a turn is clear, different kinds of models will be tested against each other as seen in Figure 4.2.



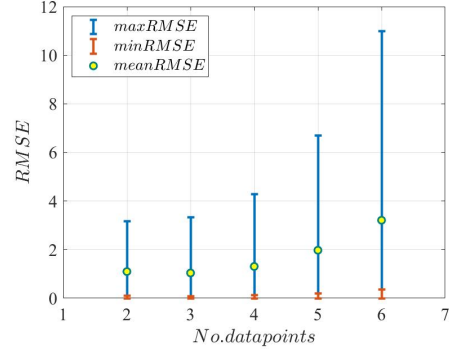
(a) Scenario: Straight Line



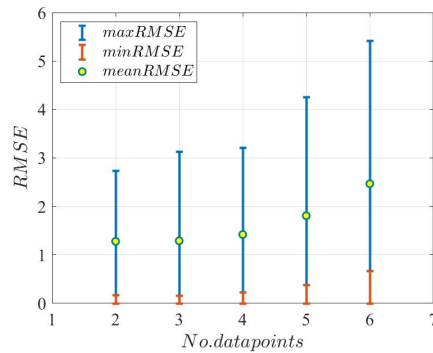
(b) Scenario: Curve



(c) Scenario: S-turn



(d) Scenario: Z-turn



(e) Scenario: U-turn

Figure 4.1: Mean-, Min-, and maximum RMSE from simulating 100 VRUs for different no. data points in five different scenarios.

As can be discerned from Figure 4.1, 2 up to 5 DPs yield good accuracy for a straight line (4.1a), although while 3 DPs seem most optimal for a curve (4.1b), 2-3 DPs yields a similar result in both the S-turn (4.1c), Z-turn (4.1d), and U-turn (4.1e) scenario. Since all the scenarios (see A.1) include parts where the VRU needs to go straight, 2 DPs always yield good results, therefore 3-4 DPs should be chosen when we detect a turn (see A.3). With these results in mind, we adjusted our prediction model to work in two different ways; one is the Check-For-Change (CFC) model, which is dependent on the current and previous DP values; i.e. the previous difference in, velocity, and angle, where we continuously look for a certain difference in these values given by parameters when triggering a VAM generation provided by ETSI [12, p. 46]; a ± 0.5 change in velocity, and/or a change in angle of 4 degrees, then adjust what number of DPs we look at when predicting. If we are accelerating, we look at 2 DPs, if we detect de-acceleration and/or a change in angle, we look at 3 DPs.

The other model is the Mean-model which will use 2 and 3, or 2, 3, and 4 DPs to predict the positions simultaneously, then return the mean x-, and y values of these positions for our new predicted position.

The results in Figure 4.2 showcase the distributed distance errors and the frequency of which they occur in the 50 designated bins for the two different models, where we have simulated all the scenarios with these models and collected all the distance errors from each VRU in each scenario (100 VRUs in each scenario and all their positions). Given the ETSI standards for generating a VAM in an ITS, that recommend the predicted position to be inside a 4-meter range from the actual position, the models demonstrate high accuracy in predicting positions within this recommended range.

Table 4.1 shows the results from Figure 4.2 and indicates that the CFC model (4.2a) generates worse predictions than the Mean Models except in the $1 \leq d < 2$ range, but more frequently generates a prediction that is in the $d \geq 3$ range. The first Mean model (4.2b) uses 2, 3, and 4 data points, and provides better results in the $0 \leq d < 0.5$ range but overall provides similar results as the CFC model. The other Mean model (4.2c) that uses 2, and 3 data seems to perform the most balanced results in these simulations and is the one used moving forward to

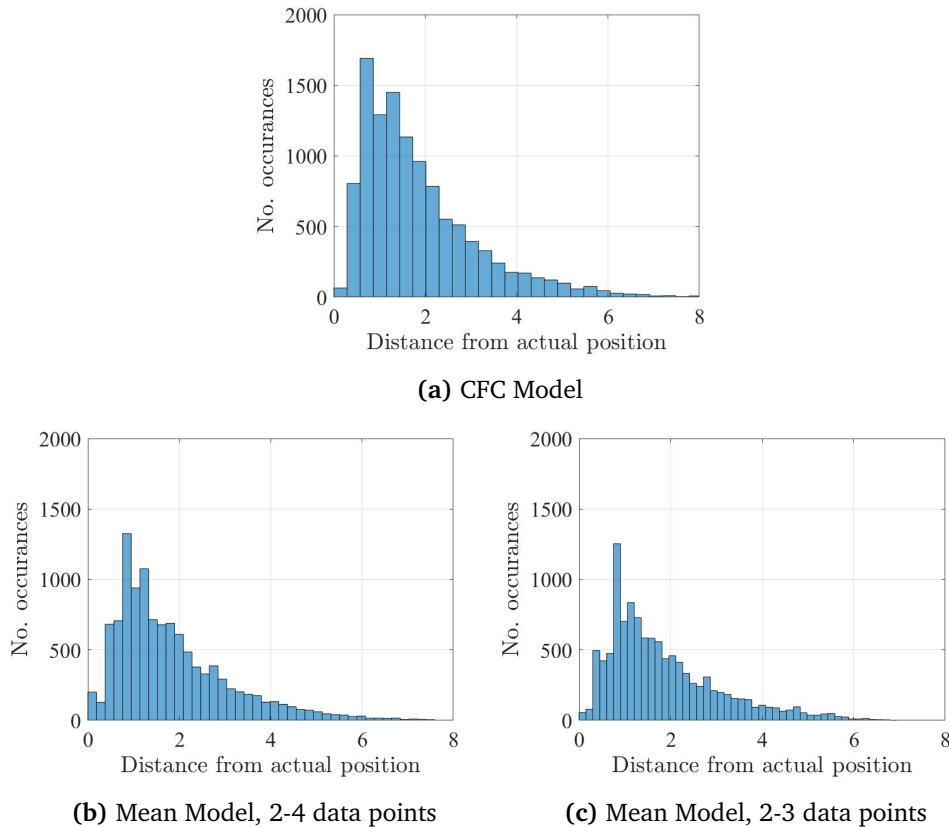


Figure 4.2: Distance error distribution comparing the CFC-, and Mean models.

Table 4.1: The number and share of predicted positions inside a range of different errors (euclidean distances between actual-, and predicted positions) for each model.

| Number and share of occurrences | | | |
|---------------------------------|----------------------|----------------------|----------------------|
| Distance error | CFC Model | Mean Model 2-4 DPs | Mean Model 2-3 DPs |
| $0 \leq d \leq 0.5$ | 709 (6.31%) | 906 (7.95%) | 768 (6.84%) |
| $0.5 < d \leq 1$ | 2467 (21.97%) | 2447 (21.48%) | 2483 (22.11%) |
| $1 < d \leq 2$ | 4156 (37.01%) | 4107 (36.06%) | 4099 (36.50%) |
| $2 < d \leq 4$ | 3043 (27.10%) | 3090 (27.13%) | 3080 (27.43%) |
| $d > 4$ | 855 (7.61%) | 840 (7.38%) | 800 (7.12%) |
| $0 \leq d \leq 4$ | 10375 (92.4%) | 10550 (92.5%) | 10430 (92.9%) |

implement onto the embedded system to visualise predictions in real-time. These three models can make safe predictions with a 92–93% confidence. However, the CFC model is most likely the most prominent for further enhancement, as it can rely on more factors such as velocity and difference in angle (further discussed in section 5.2).

Utilizing the ray-casting algorithm [25] to evaluate prediction accuracy unveiled certain challenges. Specifically, when the VRU follows a linear trajectory, the APFP may become too narrow to discern the prediction's placement within the designated area precisely. In such instances, RMSE proves more effective in measuring accuracy. To address this, a distinguishing method for straight, left-turn, and right-turn movements was developed. Consequently, the ray casting algorithm is exclusively applied when directional changes occur, while RMSE serves as the metric for assessing accuracy during straight trajectories.

Table 4.2 is the result of running each model while checking if our actual position is inside the APFP using ray-casting. The accuracy Λ is calculated by dividing the total number of successful predictions by the total number of positions \mathbf{C} giving us the mean:

$$\bar{\Lambda} = \frac{1}{N} \sum_{id=1}^N \frac{\eta_{id}}{m_{id}}. \quad (4.1)$$

Where N is the total number of VRUs in the scenario, m is the number of actual positions \mathbf{C} for that VRU, and η is the number of times the next current position \mathbf{C}_{i+1} is inside the APFP generated from the prediction \mathbf{P}_{i+1} derived from \mathbf{C}_i . For simplicity, it was decided to interpret signals that foretold the VRU is going straight as a "hit", meaning the next actual position is inside the APFP, therefore the "Straight line" (see A.1a) scenario is not included. However, since directional changes are

Table 4.2: Mean accuracy for each model in different scenarios.

| Scenario | Mean accuracy $\bar{\Lambda}$ | | |
|-------------|-------------------------------|--------------------|--------------------|
| | CFC Model | Mean Model 2-4 DPs | Mean Model 2-3 DPs |
| Curve A.1b | 0.7117 | 0.7250 | 0.7601 |
| S-turn A.1c | 0.7524 | 0.6433 | 0.6894 |
| Z-turn A.1d | 0.6482 | 0.6182 | 0.7011 |
| U-turn A.1e | 0.6469 | 0.5574 | 0.5585 |

of the most importance for VRU safety, this accuracy provides a general idea of the overall accuracy of predicting for the models.

These results show that, given a VRU positional data updated at least every 0.9 seconds, a prediction using that data can somewhat accurately tell where that VRU might be positioned next. The prediction can be calculated using a controlled mathematical model, that does not rely on storing large amounts of data and heavy processing. An APFP can be visualized using LEDs and could be transferred and broadcasted inside of a CCAM in a decentralised approach [1, p. 385], given that the data necessary for reserving an APFP is quite small and can be transferred and processed swiftly. However, there is room for improvement, such as considering counter-steering, and consideration of other factors, such as GPS error, which will be further discussed in section 5.3.

4.2 Embedded system

After successfully porting the model on the Raspberry Pi for demonstration, a Python script¹ was developed with the Pygame library [26] for creating graphics, it was used for visualizing the predicted positions for VRUs and the dynamically reserved area. The script initiates by importing the necessary libraries such as gpiozero and gpiod [27] for controlling the GPIO pins for the connected LEDs. The script first set up five LEDs, three ultra-bright LEDs showing the reserved area and two orange for signalling when turning left or right. For the script to begin working, it first needs a text file with the predicted positions and a reserved area for a VRU when turning left or right. Each file line is parsed to extract these values, which are then stored in a structured list for processing in the script. The script establishes a window where these positions and paths are dynamically drawn in the visualisation component. The coordinates from the text file are made to fit the Pygame window's coordinate system, where the origin is at the top left

¹<https://github.com/Rubbaducky95/Prediction-Model—Bachelor-thesis.git>

corner of the screen. The script scales and translates these positions based on the current focus, which centres around the VRU's current position. Each position and the reserved area is visually represented by dots, lines and shapes, coloured to enhance clarity. The script also displays textual information on the computer screen, such as the current and predicted positions, the direction of movement and any computational errors between predicted and actual positions which are calculated using the Euclidean distance formula.

To enable a live demonstration of the prediction model, the VNC viewer application[28] was initiated to remotely connect the Raspberry Pi interface to a computer (see Figure A.5). This setup made it possible to remotely manage and primarily, display the specific part of the script drawing the predicted positions and the areas. The script controls the LED indicators by adjusting the LED's brightness or binary state depending on the VRU direction (left, right, or straight). For example, suppose the VRU is predicted to turn left. In that case, the left orange LED is illuminated, and the corresponding ultra-bright LED's brightness is adjusted to signify the intensity or angle of the turn, providing a visual cue corresponding to the screen's graphical representation (see Figure A.4 for the script representation). As the script processes each data point, it ensures the updates occur at one-second intervals, simulating the GPS signal. This is done to mimic real-world conditions where the GPS would provide periodic updates on the VRU's position. If this time limit was removed and lower time steps between each position were given, it would be more accurate. Although the visual implementation was done to demonstrate the model's capabilities visually, it also shows its compatibility with running on a single-board computer and returning the predicted positions and reserved areas, filling a crucial part of the goal set at the start of the project.

Discussion

The outcomes of the different simulations performed on the Vadere simulator have led to important insights into the predictive accuracy of the prediction model for e-bikes and e-scooters and their future positions in traffic scenarios. We defined pedestrians as e-scooters and e-bikes with behaviour metrics to mimic as realistic different environments as possible in simulations. The polynomial regression model's performance, as evidenced by the RMSE measurements, presents dynamics across various manoeuvres. The results yield that using two to four data points gives acceptable accuracy for straight paths. The ETSI standards of VAM generation present an adequate accuracy in terms of what ETSI has set as good accuracy. In some scenarios where the distance has been 4 meters or higher from the predicted position, that is deemed as not accurate and poses a safety concern according to ETSI. Complex movements like curves and turns require a more sensitive adjustment of data points to optimise accuracy. The findings support the hypothesis that more complex movements require more refined data input, aligning with the work of ETSI on mobility models.

The CFC model and the mean models represent two different approaches to prediction based on polynomial regression. The CFC model, relying on changes in velocity and angle, showed promising results, especially for predicting positions within a narrower error margin. This suggests that dynamic changes in movement parameters are critical indicators for precise positioning, which are crucial for real-time applications in an automated mobility system. Compared with the mean models, which average two to four DPs or two to three DPs, it showed a broader range of errors, with some predictions exceeding three meters. This variance of data highlights the challenges in using averaged data for predicting movement in dynamic environments. These findings mark the importance of tailored data and model configuration based on the movements being predicted. Future research could explore the integration of additional predicted variables, such as environmental and real-time factors. The results presented can be pivotal for developing similar models in a CCAM environment, especially those requiring high alignment in ensuring safety and efficiency. The model's performances in different scenarios provide a foundation for refining predictive algorithms that could be deployed to handle e-bikes and e-scooters.

5.1 Limitations

Some limitations were noticed with the simulation program *Vadere*, which was recommended for use towards simulating VRUs. Even though it has various settings and parameters that are adjustable for creating different types of scenarios it lacks the additional regard for specific types of VRUs. The behaviour model that was used, GNM, was initially created and intended as a behavioural model for how pedestrians behave. The same behaviour patterns and mathematical movements do not align with the dynamics of e-bikes and e-scooters. They follow different types of traffic laws and behave and move differently as well. The GNM model made it difficult to simulate typical scenarios such as traffic stops, crosswalks, slowing down before sharp turns and not staying too close to the wall. Therefore, it was difficult to create topography in *Vadere* that accurately simulated real-life scenarios that closely matched those for e-bikes and e-scooters. Measures were taken to work around these limitations by increasing obstacle repulsion, exaggerating curves, and broadening roads to force the entities to act like e-bikes and e-scooters. If *Vadere* had a larger set of traffic tools to modify even further, the model could be designed more accurately, as well as access to a behavioural model that has more focus towards e-bikes and e-scooters. Additionally, the amount of VRUs that could be simulated at one time was limited, due to a maximum time limit on spawning entities. The time limit was at 1000 seconds, and to avoid traffic jams, the update frequency of which the entities would spawn was set between 5-10 Hz, depending on the scenario. This would limit the maximum amount of VRUs to 100-500 VRUs. However, even 5 Hz would prove problematic in some cases, therefore it was decided to run all simulations at 10 Hz, resulting in 100 VRUs for each simulation.

Another issue that sprung forward during the project work was that the GPS we were going to work with would not be able to function correctly with the Raspberry Pi. Because of this, the goal of testing the model on an e-bike was changed instead to visually show how the model works with the Raspberry Pi using LEDs. If we had made it work with the GPS and mounted it on an E-bike, it would have been possible to see how well the model would have performed in real-traffic scenarios compared to the simulation runs on *Vadere*.

Despite the reliability of mathematical methods like polynomial regression, it also has limitations. The most prominent issue is that it becomes less accurate if it uses a degree higher than four or five [24]. As mentioned before, the model could benefit from adding a specified weight or a function that assigns weights to recent data points with higher importance, thereby ensuring the model retains a high accuracy by using a low degree in the regression model. Also, if the GPS had

worked with the current hardware setup, it would have been of interest how well the model would have performed with real-time data and discover possible limits. Further comparison with the results towards existing studies is discussed in the next section 5.2.

5.2 Existing studies

The results demonstrate the potential of the prediction model for e-bikes and e-scooters, focusing on predicting future positions with a high degree of accuracy. This aligns with previous research on kinematic information and visual cues in predicting future positions. Previous studies have shown that visual parameters such as pedalling behaviour and a cyclist view contribute to predicting future positions. The project work correlates with these findings through the polynomial regression, combined with physical parameters, which can predict future positions.

However, there are areas where the model needs improvement, particularly in predicting complex manoeuvres. These limitations were also noted in previous research, where the accuracy of predictions decreases for complex movements. Approaches such as utilizing vision-based methods for detecting taillight intentions via neural networks have shown higher accuracy for these scenarios. Incorporating such techniques could potentially enhance the model for complex movement. Analysing the results regarding the hardware implementation, and visualising predicted areas via LEDs is a practical step towards interaction between VRUs and other RUs. This approach is consistent with existing studies emphasising the importance of efficient transmission of intentions. For example, research on V2X communications and decentralized coordination has shown hybrid communications architectures enhance safety and virtual cooperation. The practical demonstration of the model on a Raspberry Pi highlights its capability to operate in a real-time environment, which is crucial for application in intelligent transport systems. This implementation supports findings from previous studies that suggest real-time intention sharing can enhance overall traffic safety.

5.3 Future work

The current approach for the prediction model involves using the mathematical model polynomial regression to predict future positions based on a set number of past data points. To evolve this model, future work is proposed that can enhance the model, make it more efficient in its output and further real-world implementation to make it work better with real-time data.

To improve the model further, the RMSE data from simulations could be used to incorporate an estimated error ξ , for different scenarios a VRU could find themselves in.

$$x_i(t_i) = \alpha_0 t_i^0 + \alpha_1 t_i^1 + \alpha_2 t_i^2 + \dots + \alpha_n t_i^n + \xi_{xj}, \quad (5.1)$$

and

$$y_i(t_i) = \beta_0 t_i^0 + \beta_1 t_i^1 + \beta_2 t_i^2 + \dots + \beta_n t_i^n + \xi_{yj}. \quad (5.2)$$

Where

$$j = 1, 2, 3, \dots, \quad (5.3)$$

and

$$i = 1, 2, 3, \dots, m, \quad (5.4)$$

where m is the number of time steps for that VRU, and each j represents a specific scenario (curve, u-turn, etc.). If the model could learn what scenario the VRU is approaching, this could be implemented.

Moreover, it was noticed in the last stages of developing the model that it could be more dependent on the VRUs velocity and acceleration. If a certain number of data points were observed to see if they were de-accelerating or accelerating, the predicted positions could be adjusted accordingly. Because as of right now, the model assumes constant velocity. This could be achieved by modifying the polynomial regression model to not only account for positions and velocities but also include acceleration terms. Additionally, a weight could be added to the observed points [21], fitting the polynomial better by prioritising crucial turning points. Introducing a time-based weighting scheme that decreases for older DPs and signifies the most recent DP could generate more accurate predictions. This could lead to a higher-degree polynomial or a separate system of equations modelling position, velocity and acceleration. The same could be done for the possible turn angle θ , which is used to determine the width of the area of possible future positions (APFP), making it dependent on more factors such as velocity and acceleration.

Another problem that has been consistent for the prediction model is to predict positions when turning or in curves more accurately. Modifying the model to be more precise in turn would greatly increase its output. Tackling this problem though requires taking into account the phenomenon of counter-steering, which occurs when cycling (or riding a scooter), where a person steers a little bit in the opposite direction before turning. Attempts were made during the project work, but a specific predictive curve algorithm designed for predicting curves could be a solution to improve predictions in curves. Angular velocity components and circular motion physics must be taken into account. The model developed in this

project should also be tested against real-life data, to analyse how it performs compared to the simulations. Additionally, to consider GPS errors for real data, the model could use a margin that checks if the VRU could move to the position provided and if not, disregard the input data and wait for a new position.

Adding these functionalities to the model could enhance the model's prediction accuracy to further stay within the boundaries of the ETSI standards for VAM generation inside an ITS [12, p. 46], but will also increase its complexity consequentially restricting scalability. In section 5.3 it is given what future improvements can be made.

5.4 Scalability

The current model, along with the implementation in the embedded system, highlights the project's potential for scalability. Other intention-sharing models demand constant connectivity within a CCAM network [8, Figure 1], resulting in high network usage. Our model focuses on calculating each user's intention locally, meaning each user calculates just their own intentions. This approach would possibly allow each user to broadcast their intentions to multiple users simultaneously, rather than relying on continuous connectivity to a central network and calculating multiple other road users' intentions based on sensor data (intention detection), increasing the processing time and the risk of collisions in the wireless medium. This approach also creates a bridge between "Day 1" and "Day 3" of the evolutionary path to a fully functional CCAM [1, Figure 2]. The current model and its hardware implementation could potentially allocate resources based on network demand, significantly reducing overall bandwidth requirements and enhancing the system's scalability. However, if the model could be converted to other programming languages, C for example, the model could operate on faster types of embedded systems. Offering a faster computation time in predicting positions. This computation time will however depend on what type of GPS is used and how small the time interval i.e. update frequency is. In section 5.5, one scalable aspect regarding machine learning is discussed and how that can be implemented in the model.

5.5 Machine learning

Though the prediction model is fully regulated and controlled through mathematical models and physical parameters, it could also be combined with deep learning techniques. The model's current structure and how it handles data as input and output, the Long Short-Term Memory (LSTM) neural network could be

rewarding to group with [29]. The LSTM model is a form of an extension of the recurrent neural network (RNN) which can be of use for sequential, time series, text recognition and other types of data. It is designed to address and handle the vanishing gradient problems that typically occur in RNNs when processing long data sequences. A key feature of LSTM is the constant error carousel (CEC), it helps maintain a gradient across long sequences, allowing the model to learn long-term dependencies more efficiently. Deep learning models are well suited for capturing temporal dynamics and variance, such as VRU movement. Instead of manually defining parameters and relationships, deep learning models can automatically learn these features directly. Given the model's promise in handling sequential data over different periods would be a great reason for them to match [30].

LSTM was chosen over other models like RNN, Convolutional Neural Networks (CNN) and Multi Layers Perceptron (MLP). It behaves the same as RNN except it does not possess the same ability to handle long-term data. CNN and MLP are powerful models but are more applied towards object detection, speech recognition, image classification etc. They focus more on extracting certain features from data and what it does. The selection of deep learning models depends heavily on the data and the task. Understanding what different models do better in certain areas makes the choice easier to make and work with.

Research of possible future enchantments is made for a better view of how the model can advance and give more precise predictions with neural networks based on a deep learning model. Given how the current prediction model is structured in Java, it is recommended to be done in Python because most of the primary libraries there are used for machine learning. However, there are ways to accomplish this by using the API framework DeepLearning4j with nd4j [31]. They contain a set of tools (nd4j included) for running deep learning on Java. Nd4j contains a mix of numpy and TensorFlow operations for Java for initializing neural networks and machine learning. Since these libraries originate from Python, it is recommended to use them there instead [32].

Another issue that could arise depending on how the deep learning technique is implemented with the model is whether it can run on an embedded system and how well. First of all, it must be written in a language that the embedded system supports. Secondly, factors like real-time requirements and how quickly the model can make predictions become factors that must be taken into account. Depending on the model's complexity as a whole after it is combined with a deep learning technique can be difficult to assess how well it would perform on an embedded system. The prediction model would require less time to produce results in real-time, these factors must be considered before it is applied [33]. The

evolution from simpler, physics-based models to more sophisticated deep learning approaches underscores the problem of accurately modelling human behaviour. Traditional models focus on specific behaviours, which LSTM can learn autonomously. Despite these advancements, the current models can be combined with traditional approaches to cover human behaviours across different traffic scenarios from various cities around the world, acknowledging that it requires understanding for extensive research to perfect pedestrian path predictions [34].

Conclusion

This project has developed and implemented a predictive model for e-bikes and e-scooters for possible usage in a CCAM environment. Key findings are as follows:

- The polynomial regression model, combined with the GNM, predicts future positions of e-bikes and e-scooters with a reasonable degree of accuracy that the ETSI deems sufficient. The results demonstrated that the CFC model yielded the most potential.
- The model successfully calculates and visualizes the APFP, which is crucial for intention sharing, enabling interaction between VRUs and vehicles and possibly preventing accidents.
- The practical implementation on a Raspberry Pi 5 demonstrated the model's capability to operate in a real-time environment, providing a basis for further integration into intelligent transport systems and how the model could be further adjusted towards hardware usage.

For future work, the focus should lie on improving the current prediction model. One feature that could increase the model's accuracy is assigning different weights to the observed data points. The model could reduce the influence of less significant data by prioritising important data points in various situations. This could help the model adjust predictions dynamically based on the observed behaviour of e-bikes and e-scooters, thus improving the overall prediction accuracy. Another area of importance for future work is the integration of a GPS to enable real-time data collection and processing. The current model, which operates on simulated data, needs affirmation that it can operate on real-time GPS data. This would allow the model to adjust predictions in real time, accounting for dynamic changes in the environment and the movement of VRUs. Integrating GPS data will also help validate the model against real-world scenarios, providing valuable insights for further refinements. Real-time GPS integration will enable the system to update predictions at set intervals, improving its responsiveness and accuracy in live traffic conditions.

This bachelor thesis work advances the academic understanding of predictive modelling for e-bikes and e-scooters, providing practical insights that can influence the development of future mobility solutions for these VRUs. The demonstrated potential of this model to enhance safety and efficiency in traffic scenarios marks a crucial step towards achieving the Vision-Zero goal.

References

- [1] ‘A survey on cooperative architectures and maneuvers for connected and automated vehicles,’ *IEEE Communications Surveys and Tutorials*, vol. 24, pp. 380–403, 1 2022, ISSN: 1553877X. DOI: 10.1109/COMST.2021.3138275.
- [2] A. D. Abadi, Y. Gu, I. Goncharenko and S. Kamijo, ‘Detection of cyclist’s crossing intention based on posture estimation for autonomous driving,’ *IEEE Sensors Journal*, vol. 23, pp. 11 274–11 284, 11 Jun. 2023, ISSN: 15581748. DOI: 10.1109/JSEN.2023.3234153.
- [3] S. Zernetsch, H. Reichert, V. Kress, K. Doll and B. Sick, ‘A holistic view on probabilistic trajectory forecasting - case study. cyclist intention detection,’ *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2022-June, pp. 265–272, 2022. DOI: 10.1109/IV51971.2022.9827220.
- [4] A. Mohammadi, G. B. Piccinini and M. Dozza, ‘How do cyclists interact with motorized vehicles at unsignalized intersections? modeling cyclists’ yielding behavior using naturalistic data,’ *Accident Analysis and Prevention*, vol. 190, Sep. 2023, ISSN: 0001-4575. DOI: 10.1016/J.AAP.2023.107156. [Online]. Available: <https://research.chalmers.se/en/publication/536328>.
- [5] Transportstyrelsen. ‘Statistik över vägtrafikolyckor.’ (2024), [Online]. Available: <https://www.transportstyrelsen.se/sv/vagtrafik/statistik/olycksstatistik/statistik-over-vagtrafikolyckor/> (visited on 22/04/2024).
- [6] M. Arndt. ‘Automotive intelligent transport systems (its).’ (2024), [Online]. Available: <https://www.etsi.org/technologies/automotive-intelligent-transport> (visited on 29/01/2024).
- [7] ‘Eu statistics and national programs - we live vision zero.’ (2024), [Online]. Available: <https://www.welivevisionzero.com/statistics-and-national-programs/#:~:text=The%20measures%20relate%20to%20areas,the%20year%202050%20%2D%20VISION%20ZERO> (visited on 24/01/2024).
- [8] D. Maksimovski, A. Festag and C. Facchi, ‘A survey on decentralized cooperative maneuver coordination for connected and automated vehicles,’ *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems*, 2021. DOI: 10.5220/0010442500002932.
- [9] N. H. T. S. Administration and U. D. of Transportation, *Traffic safety facts crash • stats critical reasons for crashes investigated in the national motor vehicle crash causation survey*, 2015.

- [10] A. Malhotra. ‘Car2x c-v2x – connected vehicle pilot projects from the us.’ (Feb. 2024), [Online]. Available: <https://www.audi.com/en/innovation/future-technology/autonomous-driving/car-to-x.html>.
- [11] ‘Gradient navigation model for pedestrian dynamics,’ *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 89, p. 062 801, 6 Jun. 2014, ISSN: 15502376. DOI: 10.1103/PHYSREVE.89.062801/FIGURES/10/MEDIUM. [Online]. Available: <https://journals.aps.org/pre/abstract/10.1103/PhysRevE.89.062801>.
- [12] *Intelligent transport systems (its); vulnerable road users (vru) awareness; part 3: Specification of vru awareness basic service; release 2, 2*, ETSI TS V2.2.1, European Telecommunications Standards Institute (ETSI), Feb. 2023.
- [13] L. Du, W. Chen, C. Li, B. Tong, D. Zhang and B. Liu, ‘A vision-based taillight intention detection method for intelligent and connected vehicles,’ *Proceedings - 2022 Chinese Automation Congress, CAC 2022*, vol. 2022-January, pp. 3695–3698, 2022. DOI: 10.1109/CAC57257.2022.10056012.
- [14] L. Pipkorn, E. Tivesten and M. Dozza, ‘It’s about time! earlier take-over requests in automated driving enable safer responses to conflicts,’ *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 86, pp. 196–209, Apr. 2022, ISSN: 1369-8478. DOI: 10.1016/J.TRF.2022.02.014. [Online]. Available: <https://research.chalmers.se/en/publication/529042>.
- [15] J. M.-P. ´. Erez, O. Amador, M. Rydeberg, L. Linn ´, L. Olsson and A. Vinel, ‘Towards cooperative vrus: Optimal positioning sampling for pedestrian awareness messages,’ Dec. 2023. [Online]. Available: <https://arxiv.org/abs/2312.14072v1>.
- [16] G. Köster, F. Tremml and M. Gödel, ‘Avoiding numerical pitfalls in social force models,’ *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 87, p. 063 305, 6 Jun. 2013, ISSN: 15393755. DOI: 10.1103/PHYSREVE.87.063305/FIGURES/18/MEDIUM. [Online]. Available: <https://journals.aps.org/pre/abstract/10.1103/PhysRevE.87.063305>.
- [17] B. Kleinmeier, B. Zönnchen, M. Gödel and G. Köster, ‘Vadere: An open-source simulation framework to promote interdisciplinary understanding,’ *Collective Dynamics*, vol. 4, 1st Jan. 2019. DOI: 10.17815/CD.2019.21, published.
- [18] M. Koucky, ‘Elcyklar i trafiken - jämförande studie mellan elcyklar och konventionella cyklar i vardagstrafik,’ 2017. [Online]. Available: <https://urn.kb.se/resolve?urn=urn:nbn:se:trafikverket:diva-11714> (visited on 08/02/2024).

- [19] E. Ostertagová, 'Modelling using polynomial regression,' *Procedia Engineering*, vol. 48, pp. 500–506, Jan. 2012, ISSN: 1877-7058. DOI: 10.1016/J.PROENG.2012.09.545.
- [20] 'Math – commons math: The apache commons mathematics library.' (Mar. 2024), [Online]. Available: <https://commons.apache.org/proper/commons-math/> (visited on 26/02/2024).
- [21] 'Math – commons math: The apache commons mathematics library. 13 curve fitting.' (Mar. 2024), [Online]. Available: <https://commons.apache.org/proper/commons-math/userguide/fitting.html> (visited on 06/05/2024).
- [22] 'Raspberry pi - online documentation.' (Mar. 2024), [Online]. Available: <https://www.raspberrypi.com/documentation/> (visited on 24/04/2024).
- [23] T. O. Hodson, 'Root-mean-square error (rmse) or mean absolute error (mae): When to use them or not,' *Geoscientific Model Development*, vol. 15, pp. 5481–5487, 14 Jul. 2022, ISSN: 19919603. DOI: 10.5194/GMD-15-5481-2022.
- [24] E. M. Cramer and M. I. Appelbaum, 'The validity of polynomial regression in the random regression model,' *Review of Educational Research*, vol. 48, p. 511, 4 23 1978, ISSN: 00346543. DOI: 10.2307/1170046.
- [25] 'The point in polygon problem for arbitrary polygons,' *Computational Geometry*, vol. 20, pp. 131–144, 3 Nov. 2001, ISSN: 0925-7721. DOI: 10.1016/S0925-7721(01)00012-8.
- [26] 'Pygame - online documentation.' (Mar. 2024), [Online]. Available: <https://devdocs.io/pygame> (visited on 24/04/2024).
- [27] 'Gpiozero - online documentation.' (Mar. 2024), [Online]. Available: <https://gpiozero.readthedocs.io/en/stable/index.html> (visited on 25/04/2024).
- [28] 'Tigervnc - online documentation.' (Jun. 2014), [Online]. Available: <https://tigervnc.org/> (visited on 26/04/2024).
- [29] G. V. Houdt, C. Mosquera and G. Nápoles, 'A review on the long short-term memory model,' *Artificial Intelligence Review*, vol. 53, pp. 5929–5955, 8 Dec. 2020, ISSN: 15737462. DOI: 10.1007/s10462-020-09838-1.
- [30] F. M. Shiri, T. Perumal, N. Mustapha and R. Mohamed, *A comprehensive overview and comparative analysis on deep learning models: Cnn, rnn, lstm, gru*, Jun. 2023. [Online]. Available: <https://arxiv.org/abs/2305.17473>.
- [31] 'Deeplearning4j - introduction to core deeplearning4j concepts.' (Apr. 2024), [Online]. Available: <https://deeplearning4j.konduit.ai/> (visited on 23/03/2023).

- [32] T. Cap, A. Kreitzer, M. Miranda, D. Vadimsky and J. Picone, 'Iml: A python-based interactive machine learning demonstration,' Institute of Electrical and Electronics Engineers Inc., 2021, ISBN: 9781665428972. DOI: 10.1109/SPMB52430.2021.9672265.
- [33] E. Batzolis, E. Vrochidou and G. A. Papakostas, 'Machine learning in embedded systems: Limitations, solutions and future challenges,' *2023 IEEE 13th Annual Computing and Communication Workshop and Conference, CCWC 2023*, pp. 345–350, 2023. DOI: 10.1109/CCWC57344.2023.10099348.
- [34] A. Bighashdel and G. Dubbelman, 'A survey on path prediction techniques for vulnerable road users: From traditional to deep-learning approaches,' *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, pp. 1039–1046, Oct. 2019. DOI: 10.1109/ITSC.2019.8917053.

Appendix A

A.1 Figures and plots

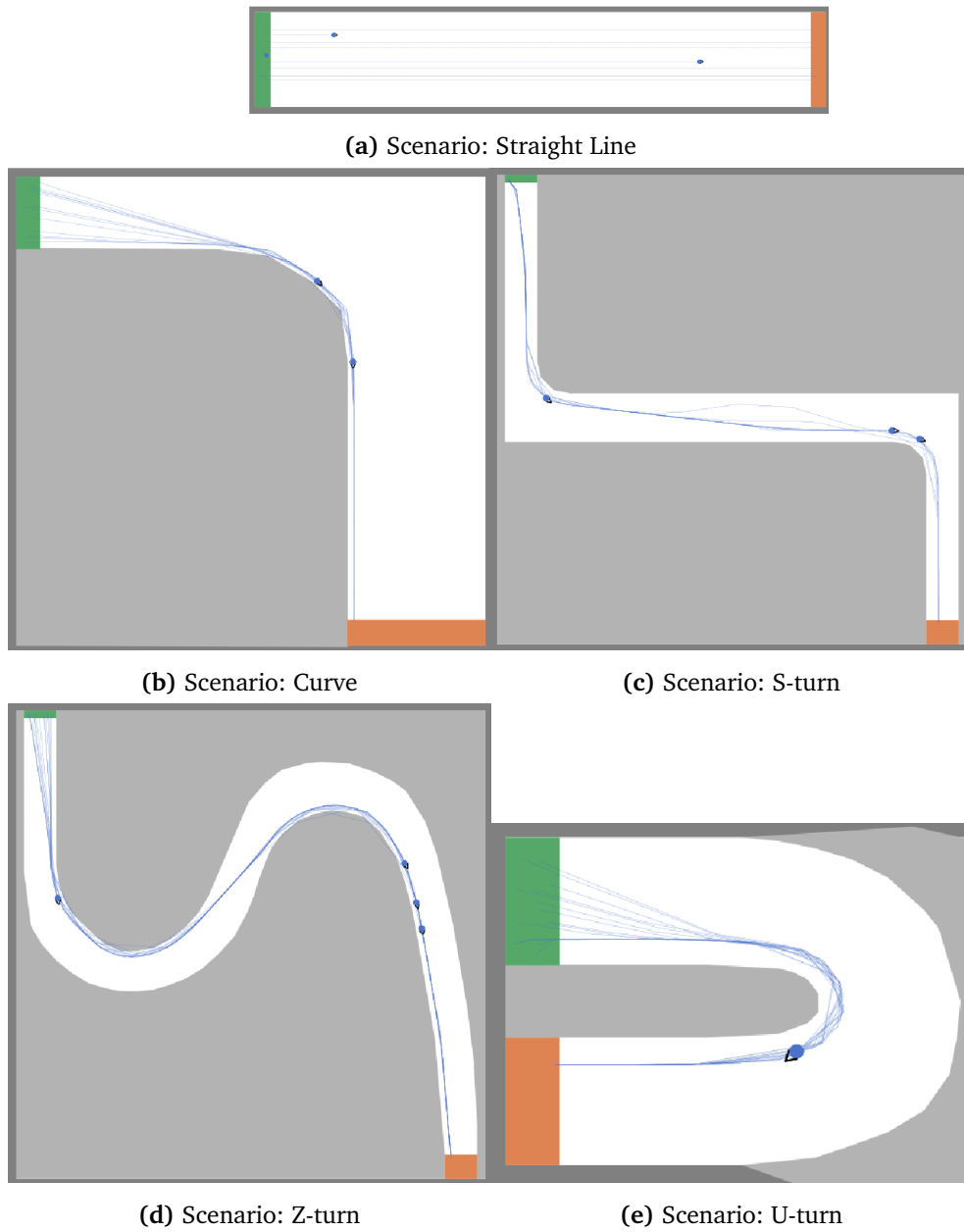
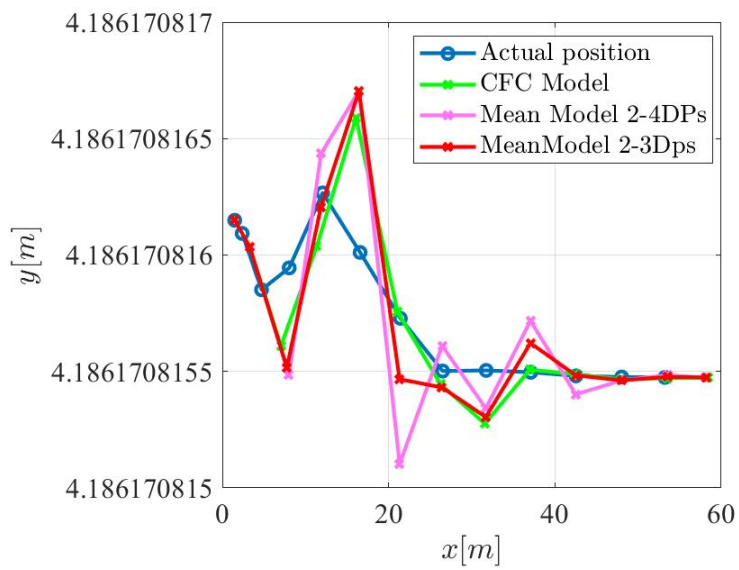
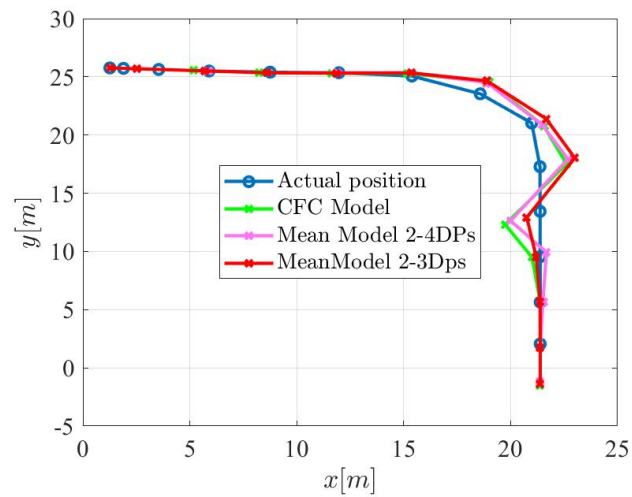


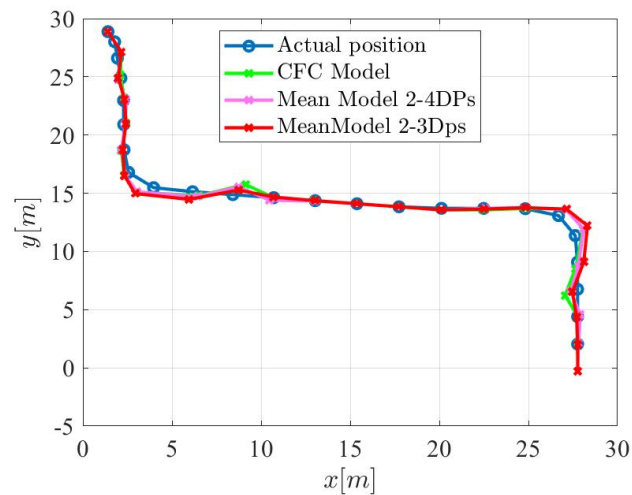
Figure A.1: Topography layout from Vadere for the scenarios used to simulate VRU dynamics in this project.



(a) Scenario: Straight Line

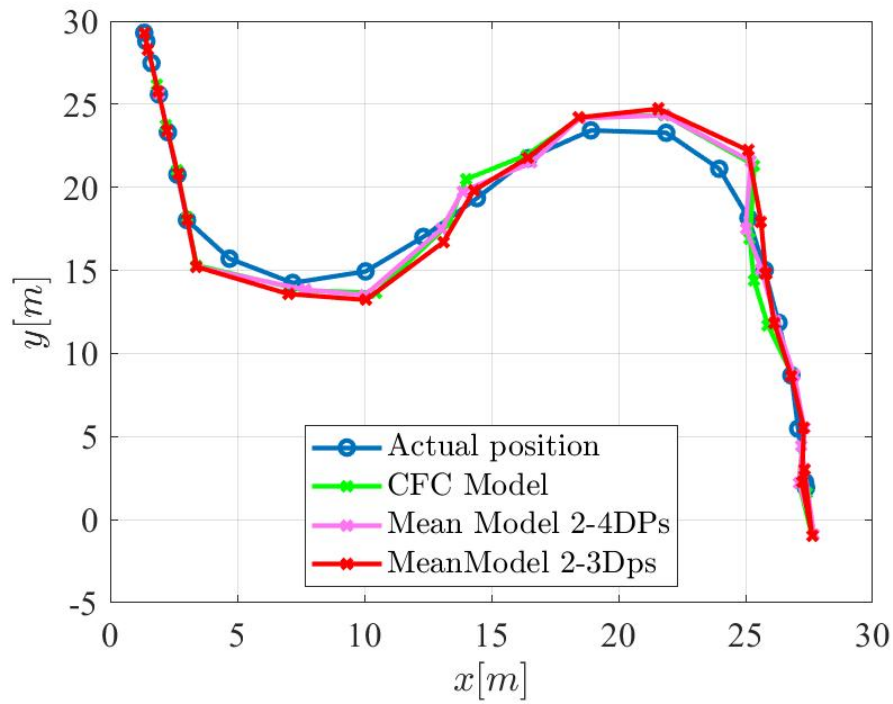


(b) Scenario: Curve

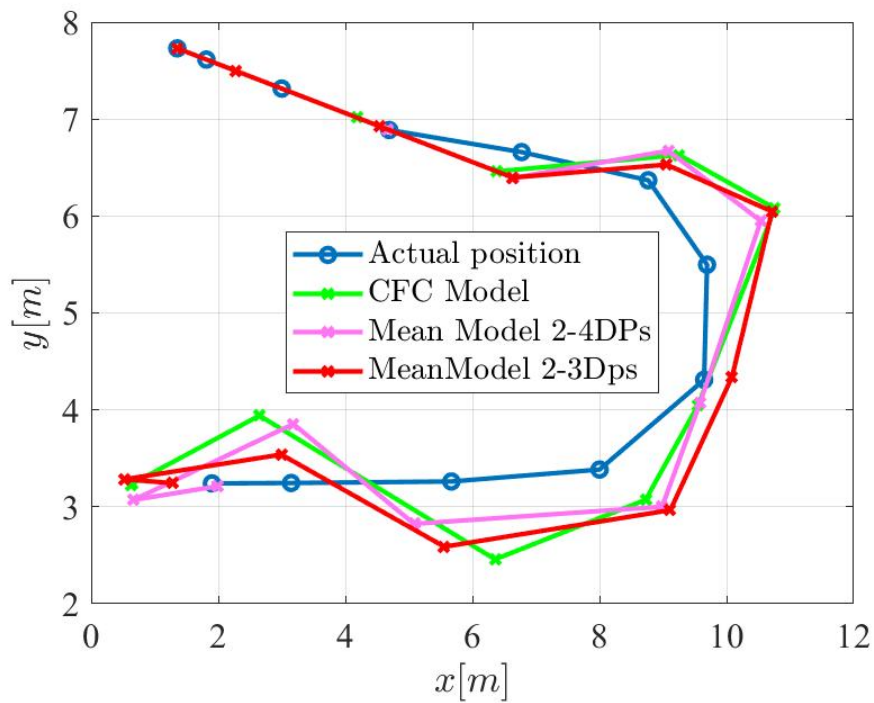


(c) Scenario: S-turn

Figure A.2: Examples of a VRUs actual-, and predicted paths with the different models.



(a) Scenario: Z-turn



(b) Scenario: U-turn

Figure A.3: Examples of a VRUs actual-, and predicted paths with the different models.

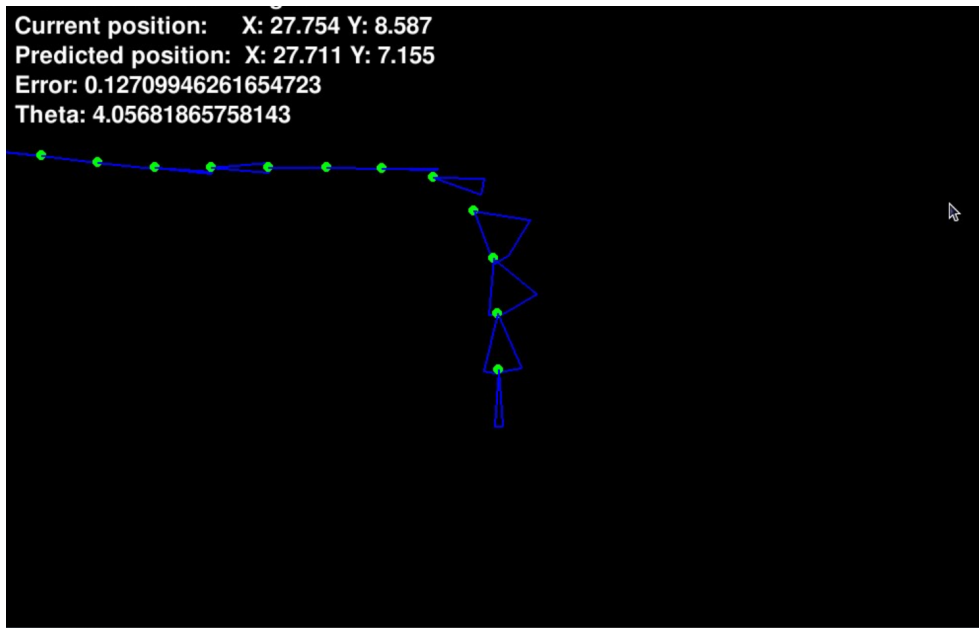


Figure A.4: Demonstration of the VNC viewer sampling predicted data points and area of possible future positions (APFP) in real-time while also displaying the coordinates of current-, and predicted positions, the distance between predicted, -and actual position (error), and the search angle (Theta) which determines the width of the APFP.

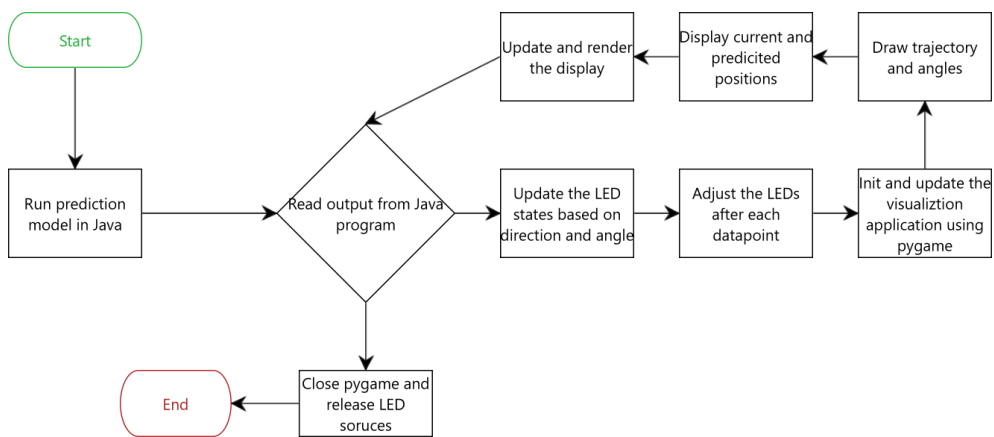


Figure A.5: Flowchart of how the prediction model is implemented in the Raspberry Pi, displaying how it reads data while activating LEDs and displaying the predictions.

Appendix B

B.1 Equations

Matrices for the predicted positions $\mathbf{P} = \{\mathbf{P}_1, \mathbf{P}_2, \dots, \mathbf{P}_m\}$ where $\mathbf{P}_i = (x_i, y_i)$.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} 1 & t_1 & t_1^2 & \dots & t_1^n \\ 1 & t_2 & t_2^2 & \dots & t_2^n \\ 1 & t_3 & t_3^2 & \dots & t_3^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_m & t_m^2 & \dots & t_m^n \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_m \end{bmatrix}, \quad (\text{B.1})$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} 1 & t_1 & t_1^2 & \dots & t_1^n \\ 1 & t_2 & t_2^2 & \dots & t_2^n \\ 1 & t_3 & t_3^2 & \dots & t_3^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_m & t_m^2 & \dots & t_m^n \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \\ \beta_m \end{bmatrix}. \quad (\text{B.2})$$

Where m is the number of time steps for a VRU, $n + 1$ is the number of data points looked at in the actual position array, and α and β are the polynomial coefficients to be solved through matrix multiplication.

Introduce two coordinate systems \mathcal{I} and \mathcal{C} . The coordinate system \mathcal{I} is inertial and it is within this system the VRU moves around. The second coordinate system \mathcal{C} varies from each discrete time step. A vector expressed in \mathcal{I} coordinates will be denoted $\mathbf{v}_{\mathcal{I}}$ whereas a vector expressed in \mathcal{C} coordinates will be denoted $\mathbf{v}_{\mathcal{C}}$. The position of the VRU \mathbf{C} at any discrete time step coincides with the origin of \mathcal{C} with unit axes

$$\mathbf{e}_x = \frac{\mathbf{CP}}{\|\mathbf{CP}\|} \quad (\text{B.3})$$

Where \mathbf{P} is the prediction of the VRU's position at the next time step and where $\mathbf{e}_y \perp \mathbf{e}_x$ according to the right-hand rule. Given the angle ω , any point in the \mathcal{C} coordinate system can be linearly transformed to the \mathcal{I} coordinate system using the rotation matrix

$$\mathbf{R}_{\mathcal{I}/\mathcal{C}} = \begin{bmatrix} \cos \omega & -\sin \omega \\ \sin \omega & \cos \omega \end{bmatrix} \quad (\text{B.4})$$

Where the subscript denotes transformation from \mathcal{C} to \mathcal{I} . By finding an expression for the edge points $\mathbf{A}_{\mathcal{C}}$ and $\mathbf{B}_{\mathcal{C}}$, they can then be projected onto the inertial coordinate \mathcal{I} system using $\mathbf{R}_{\mathcal{I}/\mathcal{C}}$ for a given angle ω . \mathbf{A} and \mathbf{B} are positioned on opposite sides of the vector \mathbf{CP} by the *search angle* θ and can be expressed as

$$\mathbf{A}_{\mathcal{C}} = P \cos \theta \mathbf{e}_x + P \sin \theta \mathbf{e}_y \quad (\text{B.5})$$

And

$$\mathbf{B}_{\mathcal{C}} = P \cos \theta \mathbf{e}_x - P \sin \theta \mathbf{e}_y \quad (\text{B.6})$$

Where $P = \|\mathbf{CP}\|$. Combining equation (B.4) with equation (B.5) and (B.6) yields the following equations that expresses the position of points \mathbf{A} and \mathbf{B} in inertial coordinates

$$\mathbf{A}_{\mathcal{I}} = \mathbf{C} + \mathbf{R}_{\mathcal{I}/\mathcal{C}} \mathbf{A}_{\mathcal{C}} = \begin{bmatrix} c_x + P \cos \omega \cos \theta - P \sin \omega \sin \theta \\ c_y + P \sin \omega \cos \theta + P \cos \omega \sin \theta \end{bmatrix} \quad (\text{B.7})$$

And

$$\mathbf{B}_{\mathcal{I}} = \mathbf{C} + \mathbf{R}_{\mathcal{I}/\mathcal{C}} \mathbf{B}_{\mathcal{C}} = \begin{bmatrix} c_x + P \cos \omega \cos \theta + P \sin \omega \sin \theta \\ c_y + P \sin \omega \cos \theta - P \cos \omega \sin \theta \end{bmatrix} \quad (\text{B.8})$$

These two equations will be used for each discrete time step, given P_i , ω_i and θ_i to provide the position of the edge points for any t_i . The area of the search sector for each t_i is

$$APFP = \theta P^2 \quad (\text{B.9})$$