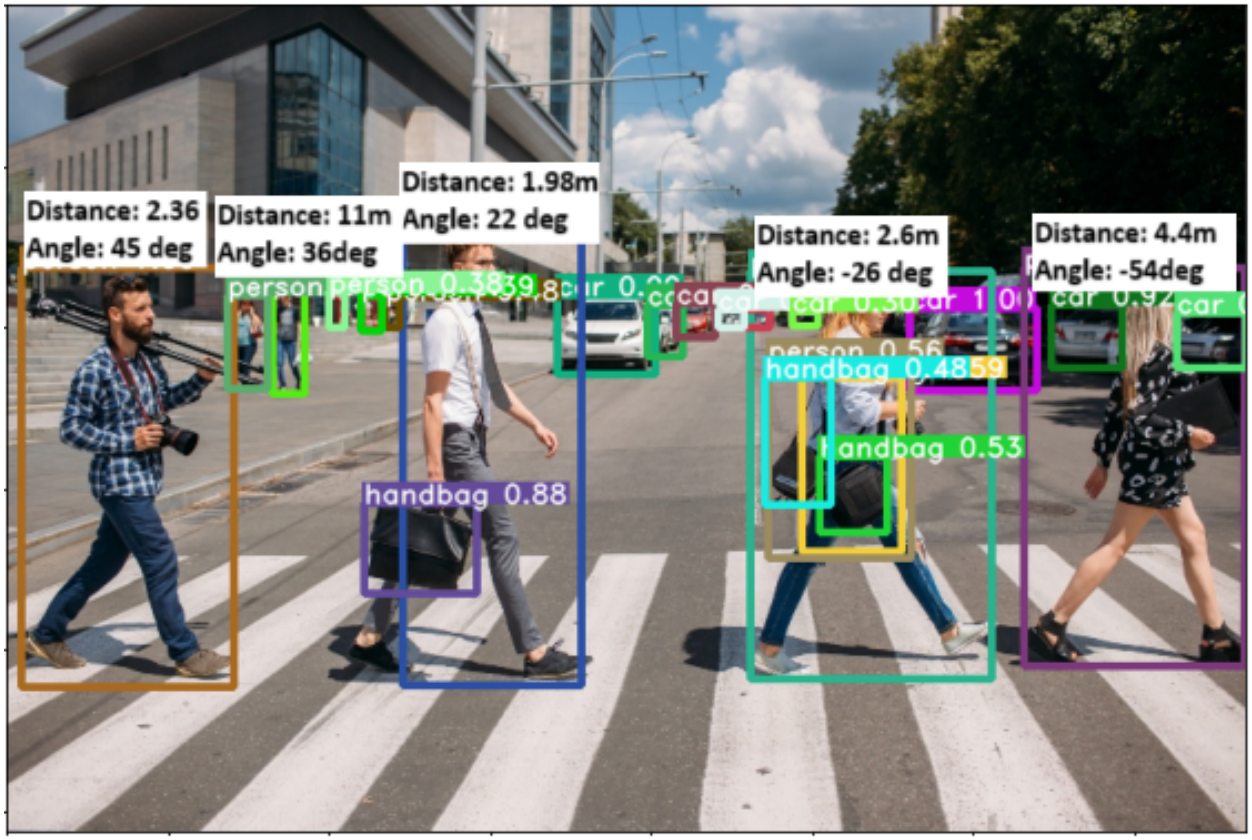
CHALMERS
UNIVERSITY OF TECHNOLOGY



Estimating road-user position from a camera: a machine learning approach to enable safety applications

Master's Thesis in Mobility Engineering

Karan Bharti

DEPARTMENT OF MECHANICS AND MARITIME SCIENCES (M2)
DIVISION OF VEHICLE SAFETY

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's thesis in Mobility Engineering

Estimating road-user position from a camera: a machine learning approach to enable safety applications

KARAN BHARTI

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mechanics and Maritime Sciences (M2)
Division of Vehicle Safety
Chalmers University of Technology
Gothenburg, Sweden 2023

Estimating road-user position from a camera: a machine learning approach to enable safety applications

MMSX30 - Master's Thesis

Department of Mechanics and Maritime Sciences (M2)
Division of Vehicle Safety
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: AI algorithm detecting objects with position estimation for pedestrians[9].

Typeset in LaTeX
Published on odr.chalmers.se
Gothenburg, Sweden 2023

Estimating road-user position from a camera: a machine learning approach to enable safety applications
Master's thesis in Mobility Engineering
Karan Bharti
Department of Mechanics and Maritime Sciences (M2)
Division of Vehicle Safety
Chalmers University of Technology

Abstract

Road user interactions are a crucial aspect of transportation safety, especially for vulnerable road users (VRU). These individuals are more susceptible to accidents and their associated consequences. It includes pedestrians, cyclists, and motorcyclists among other road users. It is of paramount importance to analyze road traffic interactions with a special focus on VRU for developing active safety algorithms, effective transportation policies, and safety measures. To this end, researchers have turned to naturalistic video data as a source of information for analyzing road user interactions. Considering the data volume, manual data reduction would be a challenge in scaling the data analysis. This underscores the importance of robust and efficient pipelines for the analysis of huge amounts of naturalistic video data, using computer vision algorithms, to help in understanding traffic interaction.
In response, this thesis delves into the realm of computer vision involving machine learning to automate video data reduction and improve the analysis of road user interactions. Leveraging lidar's accurate 3D spatial information and cameras' detailed visual data, this thesis aims to develop a machine learning model for extraction of kinematics such as distance and angle of detected VRU with a focus on pedestrians, from video files. The model was trained on lidar output as ground truth for distance and angle estimation.
By developing algorithms capable of extracting position from video data, this thesis aims to streamline the analysis process, reducing manual effort and error-prone subjectivity. This work can help in active safety research to understand road-user interactions and improve traffic safety.

Keywords: Active Safety, Video Data Reduction, Machine Learning, Kinematics Extraction, Distance Estimation, Vulnerable Road Users, Camera Calibration.

Contents

Abstract **iv**

Preface **vii**

List of Figures **viii**

List of Tables **viii**

1 Introduction **1**
1.1 Background . . . 1
1.2 Aims and objective . . . 2

2 Theory **4**
2.1 Introduction to active safety . . . 4
2.2 Computer vision . . . 4
2.2.1 Monocular camera . . . 4
2.2.2 Fisheye lens . . . 5
2.2.3 Computer vision algorithms . . . 5
2.2.3.1 You only look once . . . 5
2.2.3.2 Optical character recognition . . . 6
2.3 Lidar . . . 6
2.4 Machine learning regression models . . . 7
2.4.1 Machine learning model score . . . 8
2.5 Model training and evaluation . . . 8
2.5.1 Overfitting . . . 9
2.5.2 Normalisation . . . 9

3 Methods **10**
3.1 Test vehicle . . . 10
3.2 Data collection . . . 11
3.2.1 Region of interest . . . 12
3.3 Data analysis . . . 12
3.3.1 Video data analysis . . . 12
3.3.2 Lidar data analysis . . . 12
3.3.2.1 Ground plane removal . . . 13

3.3.2.2 Transformation of point cloud 13
3.3.2.3 Filtering 14
3.3.2.4 Segmentation and clustering 14
3.3.2.5 Tracking 14
3.3.3 Synchronising video and lidar data 14
3.3.4 Dataset - training and testing 14
3.4 Machine learning models 15
3.4.1 Parameter tuning 16
3.4.2 Overfitting 16
3.4.3 Normalisation 16

4 Results **17**
4.1 Dataset 17
4.2 Model parameters 20
4.3 Model evaluation 20
4.3.1 Polar models 20
4.3.1.1 Without normalising input features 20
4.3.1.2 Normalising input features 21
4.3.2 Cartesian models 22
4.3.2.1 Without normalising input features 23
4.3.2.2 Normalising input features 23
4.3.3 Best performing model 24

5 Discussion **28**
5.1 Region of interest 28
5.2 Machine learning models 29
5.3 Polar vs cartesian approach 30
5.4 Feature engineering 31
5.5 Conclusion 31

6 Future Work **32**

Preface

I would like to express my sincere gratitude to Marco Dozza for his meticulous review of this thesis and his invaluable contributions in shaping the report and for offering insightful recommendations regarding relevant literature. I am also deeply appreciative of Alexander Rasch and Rahul Rajendra Pai for their dedicated supervision and unwavering support. Their expertise and guidance have been instrumental in guiding this thesis from its inception to its successful completion. I am thankful for the consistent weekly meetings, constructive feedback, and their assistance in refining the experimental setup and facilitating the data collection process throughout this journey. I would also like to extend my appreciation to REVERE for providing the lidar sensor, which has been crucial for data collection. Moreover, I am grateful to the project SIMT, e-SAFER, and Voi for their funding support for this project and for providing the necessary resources, including the test vehicle, to enable the successful execution of this research. In this thesis, I have employed AI tools such as ChatGPT for paraphrasing and DALL.E for generating an illustrative image.

Karan Bharti, Gothenburg, August 2023

List of Figures

2.1 Distorted image at the periphery from fisheye lens 5

3.1 Test vehicle used for data collection 10
3.2 Experimentation location (Figure generated using Google Map) . . . 11
3.3 Pedestrain walking in front of test vehicle 11
3.4 XY coordinates as seen from top view of test vehicle 13
3.5 Location of point used as input marked with star 15

4.1 Training dataset in polar coordinates 18
4.2 Testing dataset in polar coordinates 18
4.3 Training dataset in cartesian coordinates 19
4.4 Training dataset in cartesian coordinates 19
4.5 Heat map showing error distribution in distance for non-normalised polar model 25
4.6 Random forest non-normalised polar model distance prediction vs ground truth 25
4.7 Heat map showing angle distribution in angle for non-normalised polar model 26
4.8 Random forest non-normalised polar model angle prediction vs ground truth 26

List of Tables

4.1 Distance prediction performance metrics for non-normalised polar models 21
4.2 Angle prediction performance metrics for non-normalised polar models 21
4.3 Distance prediction performance metrics for normalised polar models 22
4.4 Angle prediction performance metrics for normalised polar models . . 22
4.5 Distance prediction performance metrics for non-normalised cartesian models 23
4.6 Angle prediction performance metrics for non-normalised cartesian models 23
4.7 Distance prediction performance metrics for normalised cartesian models 24
4.8 Angle prediction performance metrics for normalised cartesian models 24

1
Introduction

1.1 Background

In the realm of road transportation, ensuring the safety of road users remains an imperative goal. The dynamics of road environments include intricate interactions between various entities, including pedestrians and vehicles. These interactions are pivotal in shaping the safety landscape and demand in-depth analysis to devise effective safety measures. According to the World Health Organization (WHO) [19], approximately 1.3 million people die each year as a result of road traffic crashes. More than half of all road traffic deaths are among vulnerable road users. In the context of active safety, VRUs refer to those road users who are not inside a vehicle and are therefore more exposed to the risk of accidents. This typically includes pedestrians, cyclists, and motorcyclists[10]. Road traffic injuries are the leading cause of death for children and young adults aged 5-29 years. Road traffic crashes cost most countries 3 percent of their gross domestic product. According to the European Road Safety Observatory [5], almost all fatalities in pedestrian crashes (98 percent) are the pedestrians themselves. One in five of all road fatalities across the EU are pedestrians. This proportion is higher than for other vulnerable road users, namely 9 percent for cyclists, 3 percent for mopeds, and 15 percent for motorcycles. According to a recent report by the UK government [15] on road casualties in Great Britain involving e-scooters, there were 1,434 casualties in collisions involving e-scooters in 2021 compared to around 350 in 2020. Of all casualties in collisions involving e-scooters, 1,102 were e-scooter users which included 10 deaths compared to one in 2020. The rise in casualties highlights the need for active research in developing safer modern transportation.

The integration of artificial intelligence and machine learning in the transportation sector has emerged as a promising avenue for enhancing road safety through a better understanding of road user behaviors. This thesis embarks on a journey to develop a model using machine learning required to augment active safety in road transportation. At its core, this thesis delves into the development of a machine-learning model designed to extract position information from naturalistic video data. Specifically, the focus lies on pedestrians, a subset of vulnerable road users whose behavior holds paramount significance in the context of road safety.

Naturalistic data has been used in the identification of crash causation mechanisms and analysis of road user behaviors. The identification primarily is based on the

kinematic data while the video data has been used to gain a spatial understanding by manual video annotation. However, these videos contain several pieces of information that may not be readily available such as the position and kinematics of the surrounding road users.

The proposed model is engineered to derive essential attributes, namely position, from video data captured by cameras installed on electric scooters. This approach could help to unravel the nuanced interactions between e-scooters and pedestrians but also offers a toolkit for identifying critical events such as near-crashes that are not detected with the kinematics of the ego vehicle. Furthermore, the application of this model could possibly extend beyond event detection. By harnessing the insights gained from the extracted position, further data analysis could help in modeling the e-scooter rider behavior.

1.2 Aims and objective

The primary aim of this thesis is to leverage machine learning techniques to develop an accurate model for extracting positional information from video data captured by fisheye lens cameras installed on a fleet of electric scooters. The overarching goal is to exploit the potentially hidden information within naturalistic data to gain a comprehensive understanding of pedestrian behavior and investigate correlations between pedestrian actions and e-scooter rider reactions, providing valuable insights into potential areas for improved road safety measures and interaction dynamics. Also, advanced data analysis and pattern recognition techniques can help to uncover intricate behavioral patterns exhibited by pedestrians and e-scooter riders. To achieve this, the following specific objectives have been outlined:

1. Acquiring ground truth data: This encompassed configuring the experimental test vehicle with pertinent sensors and data loggers, followed by executing data collection according to a protocol that entailed structured and random walking within the sensor's field of view.
2. Object detection: Employ deep learning algorithm YOLOv7 for object detection and tracking. This will serve as an input feature to kinematic extraction model development.
3. Position extraction model development: Develop a machine learning model capable of accurately extracting key positional attributes, including distance and angle, from video data.
4. Model evaluation: Analyze different model outputs in relation to the ground truth data acquired from lidar to evaluate the precision, accuracy, and overall performance of each model, with the objective of identifying the optimal model suited for our specific purpose.

The research could be formulated into the following research questions:

1. Is it possible to calibrate a camera with a fisheye lens for kinematics extraction using machine learning without knowing camera parameters?

2. Which machine learning-based model would perform best for predicting kinematics when trained on limited data?
3. What challenges and limitations arise when deploying the developed kinematics extraction model in real-world road transportation settings?

The thesis is organized as follows. The theory covers the theoretical background of the work. The methods chapter explains the methodology used to prepare various machine-learning models for the given problem. The results chapter presents the various metrics and performances for each model. In the discussion, the author's inference, conclusion, as well as limitations, and lastly proposed future work are described.

2
Theory

2.1 Introduction to active safety

Active safety refers to systems that help prevent accidents or collisions. These systems can include features such as lane departure warnings, adaptive cruise control, electronic stability control, and automatic emergency braking. In scientific papers, active safety is often discussed in the context of vehicle design and the development of new technologies to improve road safety.[2].

Active safety measures can be considered a type of engineering control and include technology that is designed to produce safer driving behavior, warn drivers of potential oncoming hazards, and automatically intervene in the event that the vehicle's conditions become high-risk [1]. This is in contrast to passive safety measures, which are active during an accident and include features such as airbags and crumple zones.

2.2 Computer vision

As per IBM "computer vision is a field of artificial intelligence that enables computers and systems to derive meaningful information from digital images, videos, and other visual inputs" [4]. Using machine learning and neural networks, computer vision systems can accurately identify and classify objects, and then react to what they "see". In this thesis algorithms such as you only look once (YOLO) and optical character recognition (OCR) have been used.

2.2.1 Monocular camera

It is a type of camera that uses a single lens to capture visual information from the environment. Unlike stereo cameras or depth sensors that utilize multiple lenses or sensors to create a 3D view, monocular cameras generate 2D images. These images offer a flat representation of the scene, capturing colors, shapes, and textures. Monocular cameras are often used in computer vision applications because they are small, lightweight, and easy to integrate into systems.

2.2.2 Fisheye lens

A fisheye lens is a type of camera lens that provides an extremely wide field of view exceeding 180 degrees. It is characterized by its distinct visual distortion, which results in a spherical or panoramic projection of the scene as can be seen in Figure 2.1. This distortion creates a characteristic convex or "fisheye" effect, where straight lines appear curved or bent, and objects closer to the edges of the frame appear larger than they actually are.
Fisheye lenses come in two main types: circular fisheye and full-frame fisheye. Circular fisheye lenses capture a circular image within the frame, with the central area showing the most detail and the periphery displaying strong distortion. Full-frame fisheye lenses cover the entire frame with the fisheye effect, resulting in a more immersive and panoramic view.[6]
In addition to their artistic applications, fisheye lenses are also used in scientific and technical fields, such as astronomy and computer vision, to capture wide-angle images and analyze distorted perspectives.

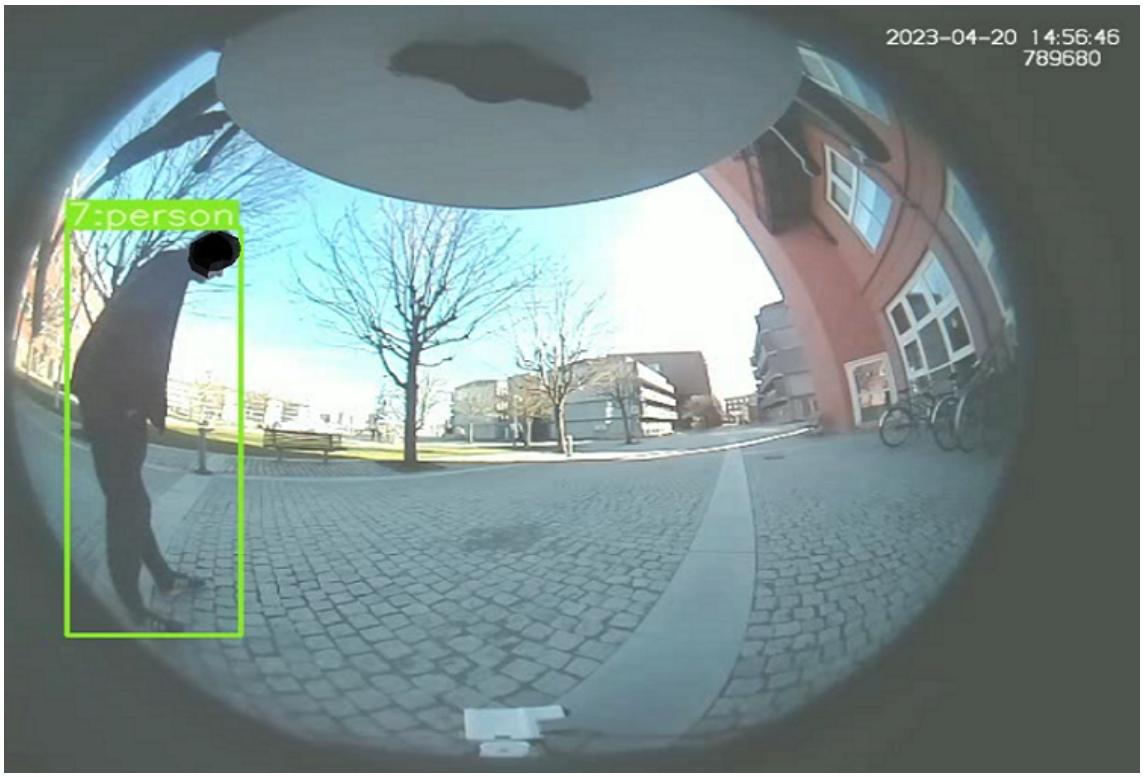


Figure 2.1: Distorted image at the periphery from fisheye lens

2.2.3 Computer vision algorithms

2.2.3.1 You only look once

YOLO is a popular computer vision algorithm used for real-time object detection in images and videos. Unlike traditional object detection methods that involve multiple stages, YOLO follows a single-stage architecture, making it faster and more

efficient. YOLO divides an image into a grid and predicts bounding boxes and class probabilities for objects within each grid cell. This approach allows YOLO to detect multiple objects in a single pass through the network[18].
Key features of YOLO[18]:

1. Real-time detection: YOLO is optimized for real-time object detection, making it suitable for applications such as surveillance, autonomous vehicles, and robotics.
2. Unified approach: YOLO performs object detection and classification in a single step, simplifying the architecture and reducing computation time.
3. Anchor boxes: YOLO uses anchor boxes to predict object shapes and sizes. These anchor boxes help improve accuracy by accommodating different object aspect ratios.
4. Convolutional neural networks (CNNs): YOLO employs CNNs for feature extraction and object detection. These networks can learn hierarchical features from images.
5. Non-maximum suppression: YOLO uses non-maximum suppression to remove duplicate or overlapping bounding box predictions, resulting in cleaner and more accurate detections.
6. Darknet framework: YOLO is often implemented using the darknet framework, which provides an open-source implementation of the algorithm.

2.2.3.2 Optical character recognition

OCR is a computer vision technology that converts printed or handwritten text into machine-readable text. OCR has widespread applications in enabling text-based search in images and scanned documents.
Key components of OCR[14]:

1. Text detection: OCR algorithms identify regions of interest containing text within images. This involves detecting and localizing text elements such as words, lines, and paragraphs.
2. Character recognition: OCR recognizes and classifies characters based on their extracted features. It matches detected patterns against a database of known characters to determine the text content.
3. Post-processing: OCR results often undergo post-processing steps to correct errors and improve accuracy. This may involve spell-checking, context analysis, and dictionary-based correction.
4. Machine learning techniques: Modern OCR systems often use machine learning techniques, such as deep learning, to improve recognition accuracy. Convolutional Neural Networks and Recurrent Neural Networks (RNNs) are commonly used for character recognition.

2.3 Lidar

Light detection and ranging (Lidar) is a remote sensing technology that employs laser light to measure distances and create detailed 3D representations of objects and their surroundings. Lidar has found widespread applications in fields such as

robotics, geology, forestry, and notably, autonomous vehicles [17]. In the context of this research, lidar plays a crucial role in providing accurate ground truth data for the machine learning model designed to predict the kinematics of detected pedestrians. Lidar operates on the principle of emitting laser pulses and measuring the time it takes for the pulses to bounce back after hitting a surface. By measuring the time-of-flight of these laser pulses, lidar systems can accurately determine the distance to objects in their line of sight [3]. The ability to capture 3D spatial information with high precision makes Lidar an ideal sensor for capturing the ground truth data necessary for training and validating the machine learning models.
One of the significant advantages of lidar is its ability to generate a point cloud representation of the environment. Each point in the point cloud corresponds to a specific location in 3D space, allowing for the creation of detailed and accurate 3D maps of the environment [11]. This capability is essential for understanding the positions and movements of pedestrians and other road users, forming the foundation for accurate kinematic prediction.
The utilization of lidar data as ground truth underscores the importance of accurate and reliable data sources in developing effective machine learning algorithms for active safety applications.

2.4 Machine learning regression models

Machine learning regression is a subfield of machine learning that focuses on building predictive models to establish relationships between input variables (features) and a continuous target variable. The goal of regression is to learn a function that can predict the target variable's value for new input data. In this thesis, these models are an integral part in predicting the kinematics of detected pedestrians based on the input feature. The four regression models which have been tried in this thesis are as follows:

1. Decision tree regressor: The decision tree splits the data based on features into segments that contain similar output values. Each leaf node represents a prediction. Splits are made to minimize the Mean Squared Error (MSE) or other similar metrics[13].
2. Random forest regressor: Random forest is an ensemble of multiple decision trees. Each tree is trained on a random subset of data, and their predictions are averaged or voted to produce the final prediction. This reduces overfitting and increases model robustness[13].
3. Support vector regressor (SVR): SVR is based on support vector machines for regression. It finds a hyperplane that best fits the data while considering a margin around it. Kernel functions are used to transform the data into higher dimensions for better separation[13].
4. Linear model with polynomial transformation: Linear models assume a linear relationship between input features and output. Polynomial transformation involves creating new features by raising existing features to a different order, allowing the model to capture more complex relationships. Linear regression with polynomial transformation fits a linear model to this transformed data[13].

2.4.1 Machine learning model score

In the context of a machine learning regression model, the score typically refers to the coefficient of determination (R^2) of the model's predictions. R^2 is a statistical measure that indicates the proportion of the variance in the dependent variable (target) that is predictable from the independent variables (features) in the model. In other words, it measures how well the independent variables explain the variability in the dependent variable [13].

2.5 Model training and evaluation

The training process involves training the models on a subset of the collected data called training set, while the evaluation process quantifies predictive accuracy and generalization capability on unseen data called validation set, using various performance metrics. The training set is used to train the model by learning the underlying patterns in the data, while the validation set helps in fine-tuning model parameters and preventing overfitting. The following steps are taken during the training process:

1. Feature selection: The input variable to the model is called feature or independent variable. They are to be decided first for training any machine learning model.
2. Feature engineering: Various techniques such as filtering cluttered data to avoid overfitting, normalising the input to avoid biases based on the magnitude could be employed to improve model's generalisation capability and improved performance.
3. Model fitting: Each regression model has its own strengths and weaknesses. Based on a particular problem, various models could be trained on decided features to predict the output they have been trained on.
4. Hyperparameter tuning: The hyperparameters of each model could be tuned to find the optimal configuration that maximizes performance without overfitting. For example, parameters like maximum depth, number of trees, kernel type, and regularization factors were some of the common parameters to be tuned for better results.

The evaluation process is essential for assessing how well the trained models generalize to unseen data. This step helps ensure that the models perform well not only on the training data but also on validation data. The following steps are taken during the evaluation process:

1. Test set evaluation: The models were evaluated on a validation set that was not used during training. The predictive performance of each model is measured by calculating metrics such as mean squared error (MSE), mean absolute error (MAE), and root mean squared error (RMSE).
2. Cross-Validation: To obtain a more robust estimate of model performance, k-fold cross-validation is employed. The dataset is divided into k subsets (folds), and the models were trained and tested on different combinations of folds. This technique helps in assessing the models' performance under various scenarios.
3. Comparison of models: The performance metrics obtained from different models were compared to identify the model that performs best for predicting

desired output.
4. Visualising results: Visualisations such as scatter plots and predicted vs. actual plots are created to visually realise the performance of various models and identify regions where models are performing best and worst.

2.5.1 Overfitting

Overfitting in machine learning refers to where a model learns to perform very well on the training data but fails to generalize effectively to new, unseen data. In other words, an overfit model fits the training data too closely, capturing not only the underlying patterns but also the noise and randomness present in the data. As a result, the model's performance on the training data is excellent, but its performance on new data is significantly worse. Such models usually show large variance or sensitivity towards input. There could be many reasons behind this, some common reasons are listed below:

1. Model complexity: Using a model that is too complex (e.g., high-degree polynomial regression, deep neural networks) can result in overfitting.
2. Insufficient data: When the training dataset is small, the model may overfit because it tries to fit noise due to a lack of representative examples.
3. Noise in data: If the training data contains noise or outliers, the model may fit these irregularities.
4. Too many features: Including too many irrelevant features or variables can lead to overfitting.

2.5.2 Normalisation

Normalisation in machine learning refers to the process of transforming numerical features to a common scale. The goal of normalization is to bring all features to a similar range to avoid biases because of the magnitude of certain input features. This is particularly important for algorithms that are sensitive to the scale of the input features, such as gradient descent-based optimization algorithms and distance-based algorithms. Normalisation is essential for the following reasons:

1. Equal treatment of features: Normalization ensures that all features contribute equally to the learning process. Without normalization, features with larger scales can dominate the learning process.
2. Faster convergence: Gradient descent algorithms converge faster when features are on a similar scale. This can lead to quicker model training.
3. Improved model performance: Algorithms like K-Nearest Neighbors (KNN) and support vector machines that rely on distance metrics can perform better with normalized features.

3
Methods

This chapter discusses the entire pipeline in detail from setting up the experimental vehicle, data collection, data analysis, and feature engineering to building various machine learning models.

3.1 Test vehicle

Our test vehicle as shown in Figure 3.1 is an e-scooter from a Swedish e-scooter rental company called Voi which refers to them as "Voiager 3X" equipped with the following equipment used in the thesis:

1. Monocular camera: It has a resolution of 720*532, a diagonal field of view (FOV) of around 220 degrees, and a frame rate of 30 frames per second (fps).
2. Lidar: It is VLP16 from Velodyne Lidar with vertical FOV of 30 deg, horizontal FOV of 360 with a rotating laser emitter, and receiver assembly. It has a range of 100m and a frame rate of 10 fps.
3. Data logger: It is based on the Raspberry Pi board.

Lidar is placed in a holder with a pitch of around 17 degrees which is roughly the inclination of the steering column of the scooter [12].

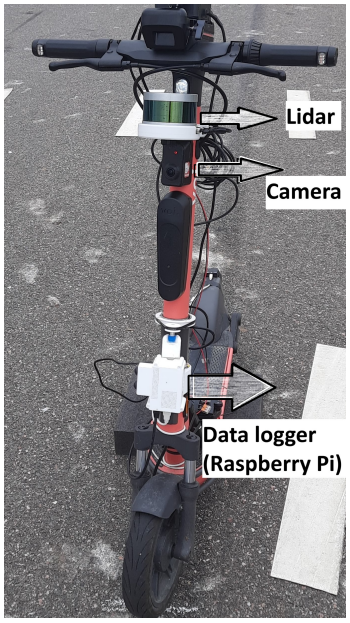


Figure 3.1: Test vehicle used for data collection

3.2 Data collection

Location - 57°42'17.5"N 11°56'18.1"E

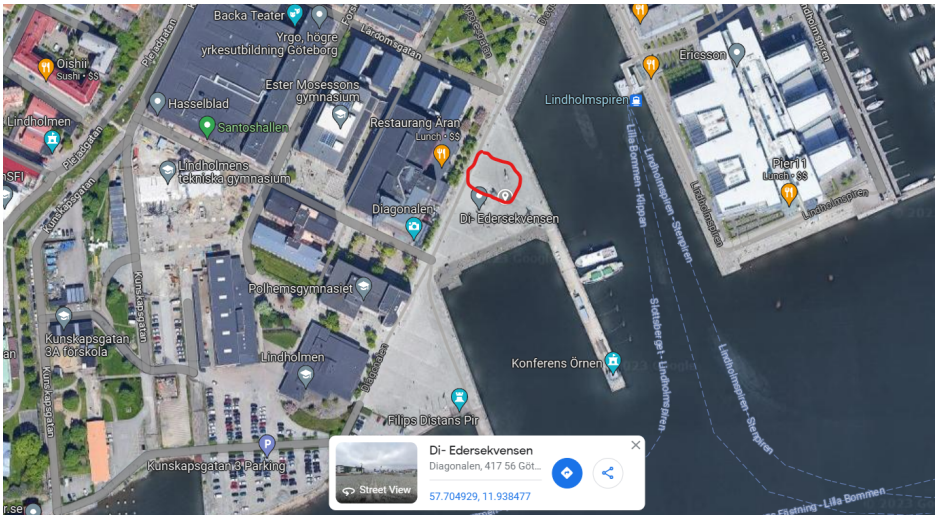


Figure 3.2: Experimentation location (Figure generated using Google Map)

The data collection process encompasses several key steps. First, a GPS time sync is established by connecting the Raspberry Pi to a hotspot. Following this, the lidar and data logger are activated, while the camera is powered on in tandem. The pedestrian then systematically traverses the camera's field of view, covering distances of up to 12 meters through a combination of structured and random movements. This approach ensures comprehensive coverage of the designated grid and captures the entirety of the domain under study. Finally, the lidar is turned off, marking the completion of the data collection process.

Figure 3.3: Pedestrain walking in front of test vehicle

3.2.1 Region of interest

Data is gathered from pedestrians walking within a 12-meter radius from the camera, encompassing an angular range of -90 to 90 degrees (with 0 degrees denoting the camera's front direction), facilitated by the fisheye lens's field of view. This designated area is now referred to as the "region of interest." Notably, YOLO pedestrian detection encountered challenges beyond the 12-meter distance and extreme angles due to image distortion and low resolution. Consequently, the data points near the boundaries are limited within the specified region.

3.3 Data analysis

What we get from the data collection setup is pcap file from lidar and MP4 from the camera along with a log file stating the starting timestamps for the video file.

3.3.1 Video data analysis

The MP4 video captured by the e-scooter is accompanied by a log file containing initial timestamps and uptime data. The uptime data is utilized to compute the time difference between frames with a resolution up to milliseconds. This video is subjected to analysis using the YOLOv7 deep-learning object detection algorithm. Additionally, the Pytesseract library is applied to extract uptime characters from each frame (refer to the top right corner in Figure 2.1) using OCR (refer to Theory for detailed information), which facilitates the computation of timestamps for individual frames. Consequently, the outcome is a video featuring bounding boxes around detected pedestrians, along with the generation of a JSON file that encompasses timestamp details, object categories, IDs, and the coordinates of bounding boxes for identified objects.
The JSON file requires certain postprocessing steps, including interpolation, extrapolation, and addressing incorrect uptime readings. After this process, the bounding box coordinates of the tracked pedestrian are extracted, along with their corresponding timestamps.

3.3.2 Lidar data analysis

The pcap file encompasses systematically organized point cloud data presented as an M*3 matrix. It includes details about the number of frames and timestamps relative to the initial frame. Within the point cloud, our subject of interest (a pedestrian) is identified and tracked. The preprocessing phase involves a series of steps carried out in MATLAB, which encompasses removal of the ground plane, transformations, filtering, segmentation, clustering, computation of centroids for each cluster, and cluster tracking.
The outcome of this scripting process yields a matrix that comprises timestamps alongside the X and Y coordinates (as illustrated in Figure 3.4) of the centroid of the tracked cluster. This data holds significance in determining both the distance and angle of the centroid concerning the lidar within the XY plane.

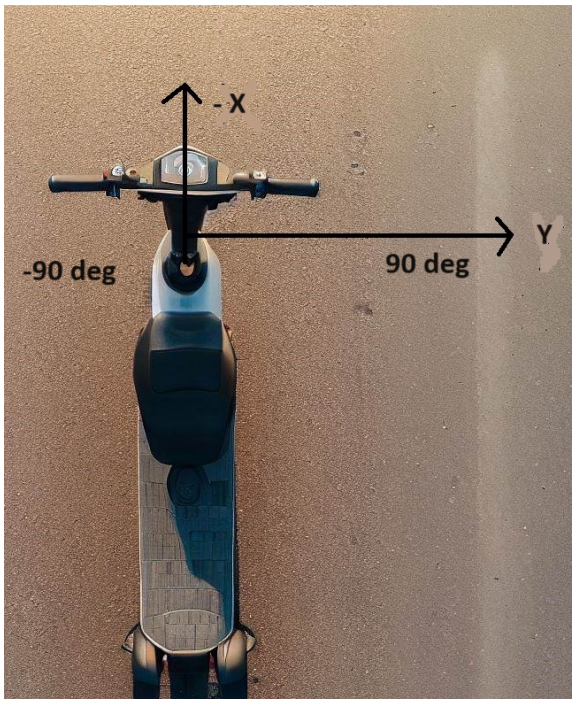


Figure 3.4: XY coordinates as seen from top view of test vehicle

3.3.2.1 Ground plane removal

The computer vision toolbox in MATLAB has a function 'segmentGroundFromLidarData' which involves plane fitting using RANSAC algorithm [8]. The input to this function should be transformed point cloud with ground parallel to the xy plane. A threshold could be additionally defined for segmenting points as grounds that are within the threshold distance from the plane. In this thesis, a threshold of 0.1m has been used.

3.3.2.2 Transformation of point cloud

As previously mentioned, our lidar holder is not positioned horizontally; it is inclined at an angle equivalent to the caster angle, which signifies the angle between the steering column and the ground's normal. This angle is approximately around 17 degrees, although slight variations may occur based on the specific configuration of the test vehicle. While we aim to place the vehicle in an upright position with both tires on the ground, there might be slight variations in elevation due to the ground's contour or the support used for the scooter. To ensure the reliability of our data collection and accommodate these external pitch variations, we employ a different approach. We identify the ground plane through clustering and plane fitting techniques and determine its orientation relative to the global XY plane. This calculated angle is then utilized to transform the point cloud in the pitch direction.

3.3.2.3 Filtering

To reduce the computational effort going forward, all the points outside the region of interest are removed. Denoising and downsampling by 10 percent is done to remove any noise in the lidar data. Noise could refer to outliers that do not have relevance to the desired study.

3.3.2.4 Segmentation and clustering

Function 'pcsegdist' from computer vision toolbox based on MATLAB was used to cluster point spatially. It uses a distance threshold to identify a cluster. This threshold signifies Euclidean distance within this threshold from a certain point will be a part of the same cluster. The threshold used for pedestrian clustering was 0.5m. For faster computations, the parameter 'ParallelNeighborSearch' was set to true as it enables parallel computations [7].

3.3.2.5 Tracking

Following the segmentation of the pedestrian from the point cloud, the script calculates the centroid of every cluster. It identifies the relevant cluster to track, guided by the user's initial input value. The initial input value, representing the coordinates of a point in the pedestrian cluster to be tracked, is manually input by the user. Furthermore, the script continuously monitors the nearest cluster in consecutive frames, considering the distance covered between frames to ensure accurate tracking. A threshold of 0.5 meters between successive frames (100 milliseconds) has been employed for this purpose in the thesis.

3.3.3 Synchronising video and lidar data

In instances where YOLO detects a pedestrian, the X and Y coordinates of the centroid are determined through linear interpolation between neighboring lidar frames that are close to the time of interest. This is necessary due to the distinct frame frequencies of the lidar and camera, as well as their separate activation times for recording. The outcome of this process yields a matrix that encompasses timestamps, locations of bounding boxes, as well as X and Y coordinates pertaining to detected pedestrians. These coordinates serve as ground truth to train machine learning models built around cartesian coordinates.
Using the X and Y coordinates, calculations are performed to deduce both the distance and angle. These calculated values serve as the ground truth for the purpose of training machine learning models, which are built around polar coordinates.

3.3.4 Dataset - training and testing

During the course of this thesis, data collection has been carried out on multiple occasions. A dataset with about 33000 points was obtained after synchronisation of the video and lidar. Each point is an array containing the timestamp, position (both in cartesian coordinates and polar coordinates), and bounding box coordinates. These were then filtered as detailed in 3.4.2 reducing the overall data points

to about 11000. The filtered data has been used for training to avoid overfitting in the model. The filtered data points were randomly split in a ratio of 4:1 to generate training and testing datasets respectively.

3.4 Machine learning models

Supervised machine-learning regression models, including decision trees, random forests, support vector regressors, and polynomial-fitted linear regression, were trained using lidar output as the reference for ground truth. Two distinct approaches were pursued:

1. Polar coordinates model: This model estimates the pedestrian's distance and angle relative to the camera's position.
2. Cartesian coordinates model: This model estimates the absolute position of the pedestrian in cartesian coordinates.

In both approaches, the input parameter consisted of the coordinates of the bounding box, specifically the midpoint of the lower horizontal side of the rectangle as shown in Figure 3.5. Including additional inputs like aspect ratio, height, or width of the bounding box was deliberately avoided. This decision aimed to prevent biases that could arise due to variations in pedestrian physique, walking style, or running pattern.

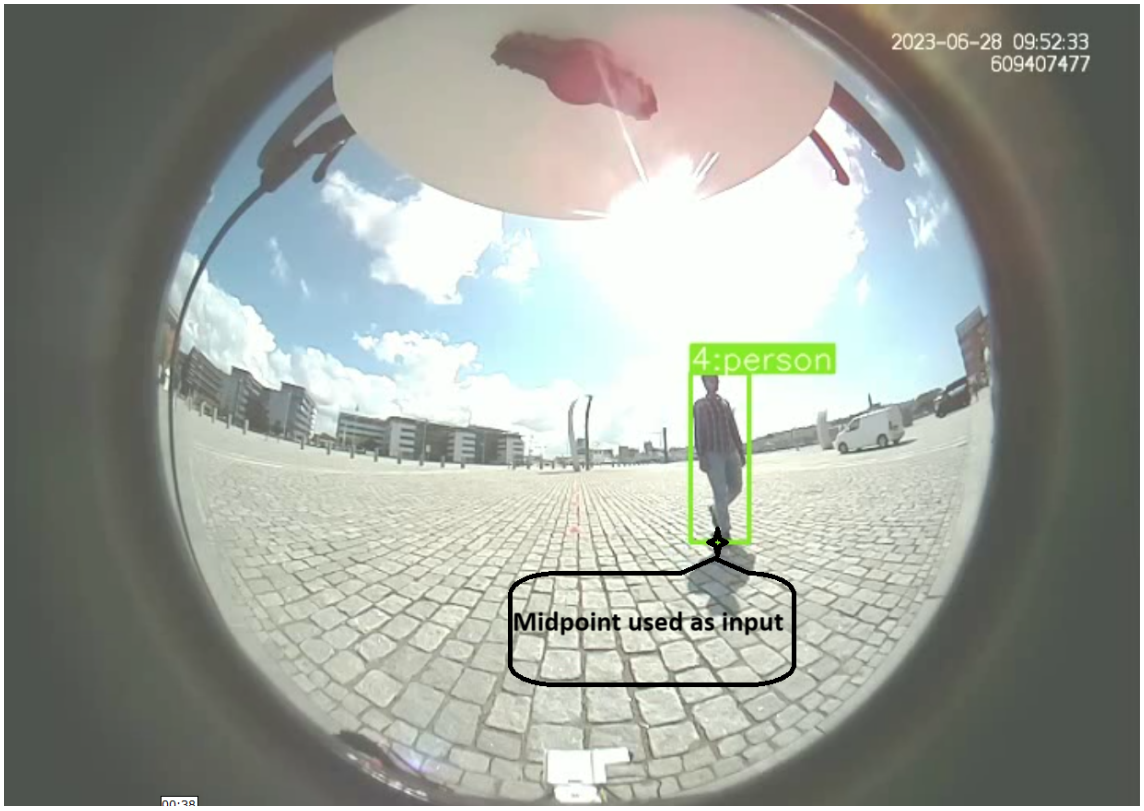


Figure 3.5: Location of point used as input marked with star

3.4.1 Parameter tuning

1. Decision tree: Parameters such as the criteria for determining data split quality, maximum tree depth, and complexity pruning parameters were refined through tuning.
2. Random forest: Parameters encompassing criteria, number of trees, maximum tree depth, and complexity were tuned.
3. SVR: The kernel type, degree for the polynomial kernel, and regularisation parameter underwent tuning.
4. Polynomial regression: The degree of polynomial transformation was tried from 2 to 5.

3.4.2 Overfitting

To avoid overfitting, the following filtering steps have been taken:

1. Rounding of X and Y coordinates from Lidar from 4 decimal points to 2 decimal (1cm resolution).
2. Filtering out repetitive coordinates data (ground truth).
3. Filtering out repetitive bounding box coordinates (input).

This was done to ensure each data point is unique and equal weightage is given to all parts of the grid.

3.4.3 Normalisation

Normalisation refers to the process of scaling the features of a dataset so that they have similar magnitudes. It is a crucial feature engineering that can help improve the convergence during training and performance of machine learning models.
The Z-Score (Standardization) has been used to train the models which makes the mean equal to 0 and the standard deviation 1.

$$x_{\text{scaled}} = \frac{x - \text{mean}}{\text{standard deviation}}$$

4

Results

The focal point of this chapter revolves around the predictive performance of each model, both with and without normalization. Two primary approaches have been undertaken for model development. The first involves training the polar model directly on distance and angle, while the second employs training the cartesian model on X and Y cartesian coordinates to subsequently determine the distance and angle through post-processing. In both scenarios, the input feature corresponds to the location of the lower midpoint of the bounding box coordinates.

4.1 Dataset

Figure 4.1,4.2, 4.3 and 4.4 illustrates the reference data employed for training and evaluating the polar and cartesian models respectively. The data reveals a comprehensive angle range coverage up to 6 meters, while an angle span of -30 degrees to 60 degrees is included for distances up to 10 meters. At greater distances, the angle coverage diminishes further. This phenomenon may be attributed to reduced pedestrian detection accuracy at extreme angles and increased distances, attributed to increased distortion at the edges of the fisheye lens and lower camera resolution respectively. The camera is placed at the origin (marked by a black dot) facing the negative X direction in cartesian plots as shown in Figure 4.3 and 4.4.

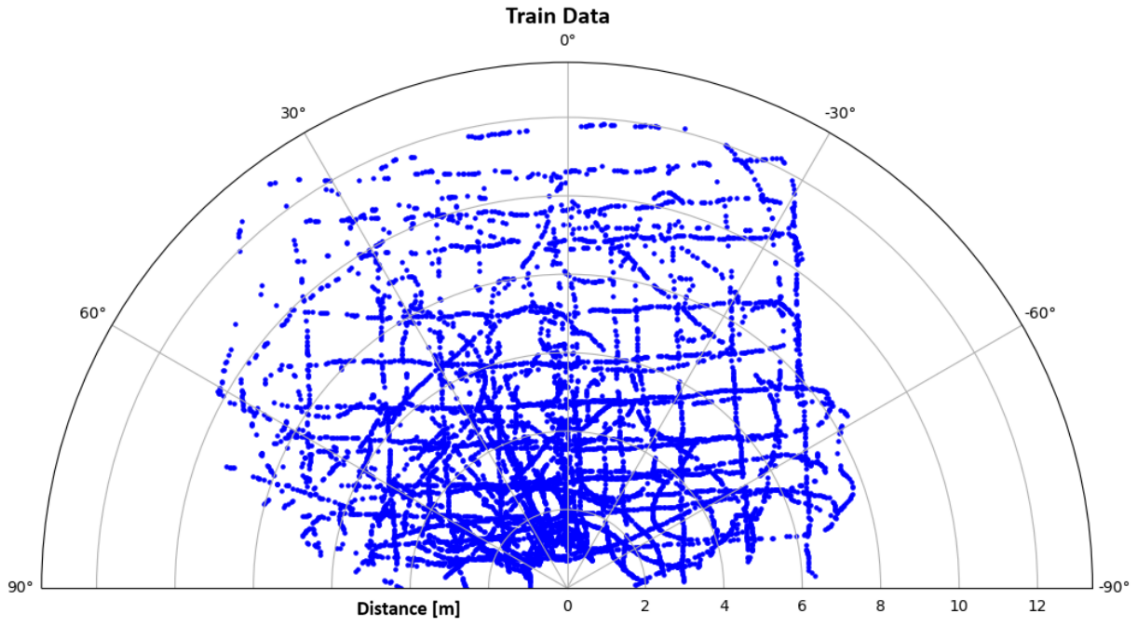


Figure 4.1: Training dataset in polar coordinates

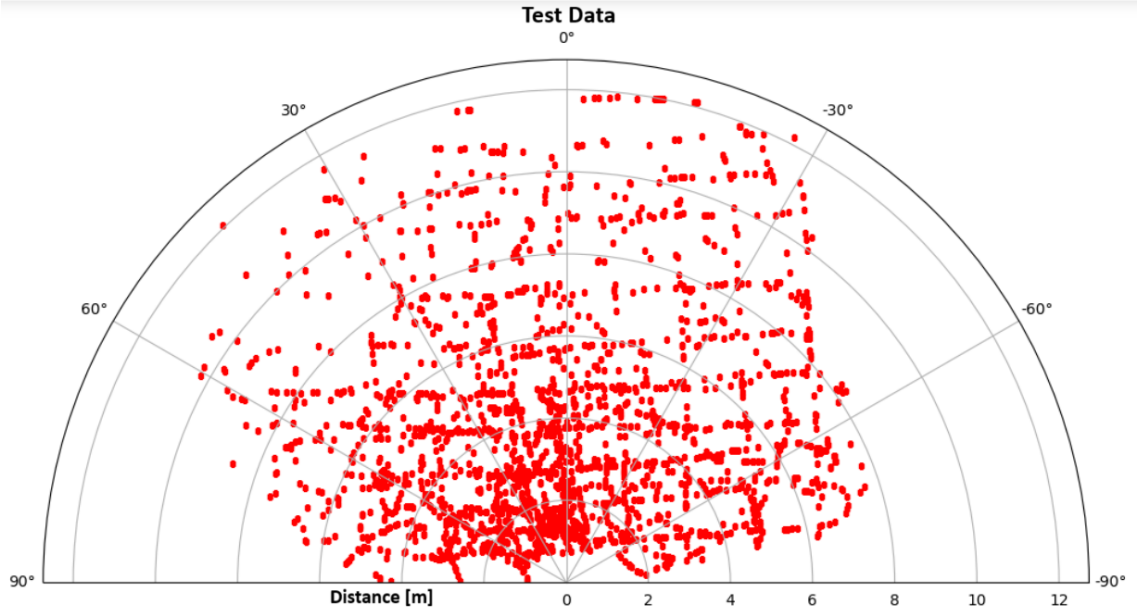


Figure 4.2: Testing dataset in polar coordinates

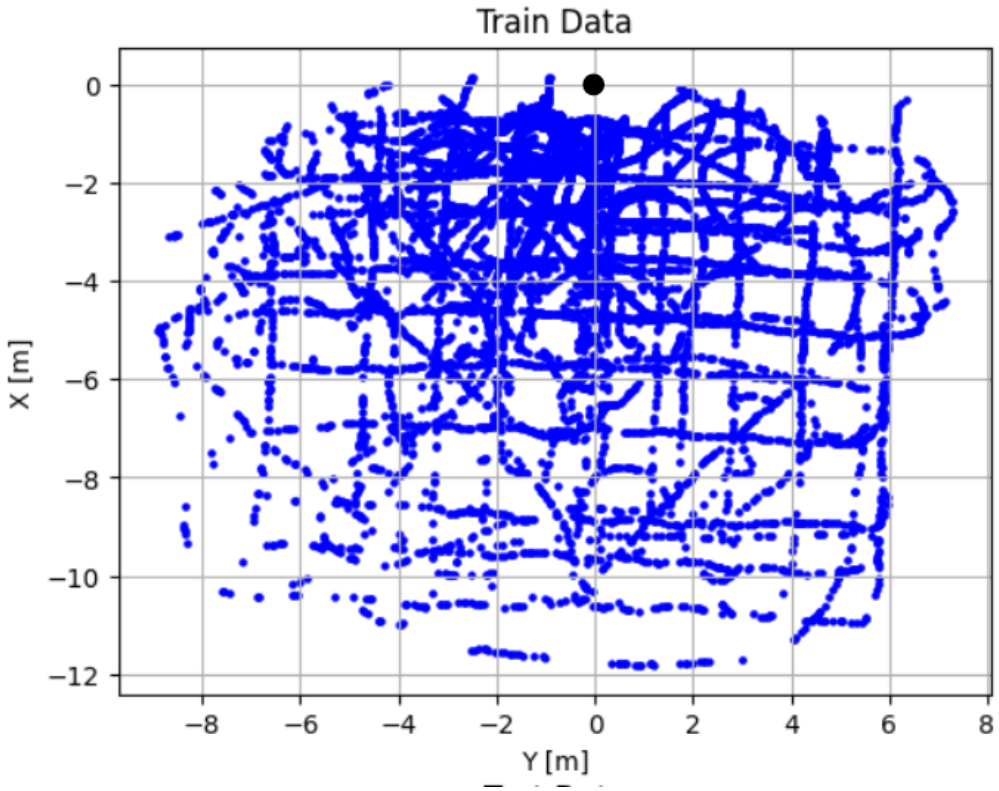


Figure 4.3: Training dataset in cartesian coordinates

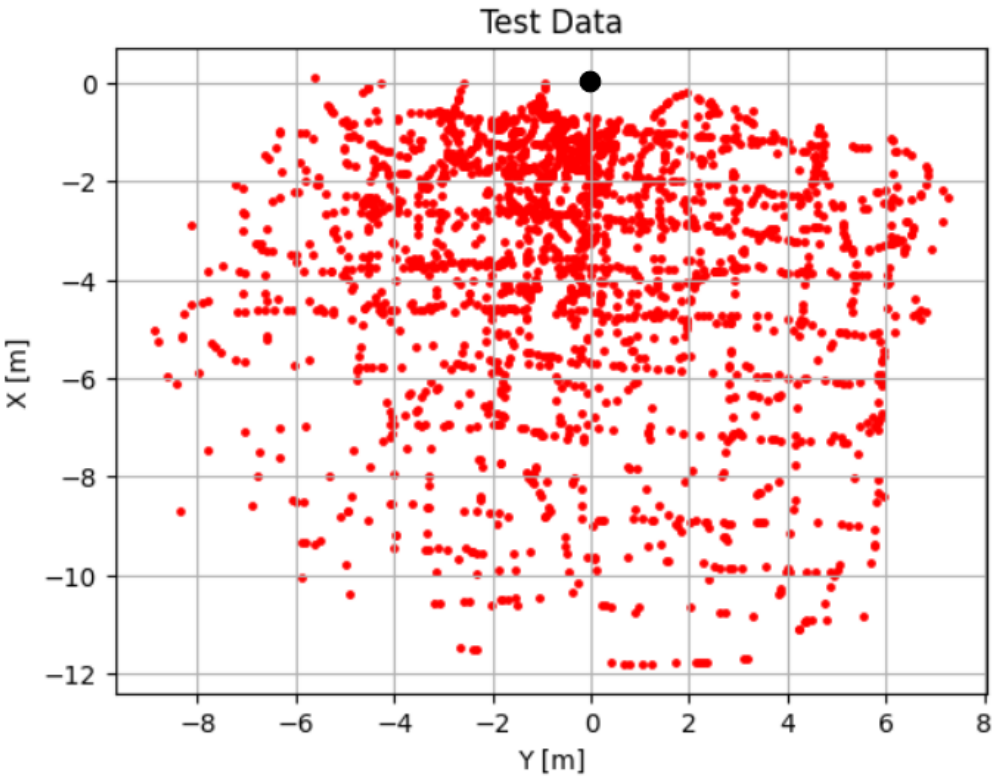


Figure 4.4: Training dataset in cartesian coordinates

4.2 Model parameters

All of the regression models, including decision tree, random forest, SVR, and polynomial regression, underwent tuning by experimenting with various parameters through trial and error. The parameters that yielded the most favorable outcomes were selected for each model, and the details of these parameter configurations are presented below:

1. For the decision tree model, the optimal setup involved using the "squared error" criterion, setting the maximum tree depth to 10, and utilizing the default complexity pruning parameter of 0.
2. In the case of the random forest model, performance improvements were achieved by employing the "squared error" criterion, increasing the number of trees to 300 (a slight improvement over the default value of 100), setting the maximum tree depth to 20, and using a complexity parameter of 0.
3. The SVR model demonstrated its best performance with an "rbf" kernel and a regularization parameter of 50.
4. As for the polynomial regression model, a polynomial degree of 3 was determined to be the optimal choice for achieving the desired performance.

The aforementioned model parameters were configured specifically for the polar model, which predicts distance. Notably, parameter tuning yielded limited enhancements for the angle prediction in the polar model, with the exception of SVR, where a regularisation factor of 50 was effective. As a result, default parameter values were retained for the angle prediction models. For the cartesian model, default values were retained.

4.3 Model evaluation

In the upcoming sections, the summary tables outline the performance metrics for all machine learning models. These metrics depict the models' performance on the test data points. The prediction error corresponds to the difference between the measured value and the predicted value. The Min and Max columns represent the minimum and maximum values of the prediction errors for each model. At the same time, Std indicates the standard deviation of the errors and Mean indicates the mean of the errors. The R^2 Score column reflects the overall model performance, with higher scores indicating better results as explained in 2.4.1.

4.3.1 Polar models

The polar model corresponds to the regression models directly fitting the distance and angle.

4.3.1.1 Without normalising input features

Table 4.1 is the result summary for distance prediction without normalising the input feature, for various models

Model	Min[m]	Max[m]	Std[m]	Mean[m]	R^2 Score
Decision Tree	-4.238	2.610	0.476	-0.006	0.968
Random Forest	-3.524	2.170	0.414	-0.004	0.976
SVR	-3.784	4.545	0.992	0.151	0.858
Polynomial	-3.799	3.791	1.022	0.003	0.853

Table 4.1: Distance prediction performance metrics for non-normalised polar models

The data presented in Table 4.1 highlights that the mean error is centered around zero for all models except SVR, which exhibits a positive offset of approximately 15cm. Additionally, the error dispersion for SVR is relatively higher, with values approaching 1m. Despite the fact that the initial two models demonstrate adeptness in capturing the variance between output and input, as indicated by their higher R^2 scores, the error range remains on the higher side compared to the standard deviation. Overall, the random forest model appears to outperform the others with the lowest range, standard deviation, and a mean close to zero.
Similarly, table 4.2 is the summary for angle prediction

Model	Min[deg]	Max[deg]	Std[deg]	Mean[deg]	R^2 Score
Decision Tree	-39.100	43.699	3.937	0.017	0.991
Random Forest	-22.304	38.497	3.140	0.072	0.994
SVR	-19.980	29.737	3.944	0.512	0.991
Polynomial	-20.348	28.378	3.937	0.068	0.991

Table 4.2: Angle prediction performance metrics for non-normalised polar models

As shown in Table 4.2, a comparable pattern can be observed in the mean error values, consistently centered around zero but with the highest offset originating from SVR. Likewise, the error range appears to be relatively larger than the variance. Nevertheless, all models exhibit an equal capacity to capture the variance between the output and input, as evident from their R^2 scores. No clear victor emerges in this scenario. The random forest model possesses the lowest standard deviation, but its error range tends to be on the higher side, which is not desirable. Conversely, SVR exhibits the best range performance, albeit with a mean error that leans toward overestimation rather than underestimation.

4.3.1.2 Normalising input features

As mentioned, Z score normalisation has been adopted to keep the mean 0 and a standard deviation of 1.
Table 4.3 is the summary of the results for distance prediction using polar models on normalised input

Model	Min[m]	Max[m]	Std[m]	Mean[m]	R^2 Score
Decision Tree	-4.238	2.610	0.476	-0.007	0.968
Random Forest	-3.355	2.184	0.417	-0.004	0.975
SVR	-3.723	2.680	0.496	0.003	0.965
Polynomial	-3.561	2.381	0.588	0.001	0.951

Table 4.3: Distance prediction performance metrics for normalised polar models

Table 4.3 indicates a uniform ability among all models to effectively capture the variance between the output and input as reflected from the R^2 score, with the mean error consistently centered around zero across all models. The variation of error is also notably consistent, remaining below 50 cm for all models except the polynomial fit model. In terms of the error range, the random forest model stands out as the frontrunner, ultimately demonstrating the most favorable overall performance. Similarly, table 4.4 shows summary for angle prediction

Model	Min[deg]	Max[deg]	Std[deg]	Mean[deg]	R^2 Score
Decision Tree	-39.699	43.699	3.937	0.039	0.991
Random Forest	-20.584	38.828	3.140	0.079	0.994
SVR	-22.109	31.044	4.192	0.237	0.990
Polynomial	-20.281	25.060	3.695	0.079	0.992

Table 4.4: Angle prediction performance metrics for normalised polar models

Table 4.4 effectively displays consistent R^2 scores for each case, all of which exhibit a positively centered offset, with the SVR model displaying the highest offset. The average standard deviation across the models is approximately 3.5 degrees. In terms of error range, the decision tree model displays the poorest performance, while the polynomial fit model showcases the best performance. Taking into account all the metrics, the random forest and the polynomial fit model emerges as the two most favorable performer among the models. The random forest model has a higher range (difference between minimum and maximum values) by 11 degrees than the polynomial fit but a smaller deviation by 0.5 degrees and an equal mean error.

4.3.2 Cartesian models

The cartesian model corresponds to the regression models predicting the absolute position in the XY plane.To enable comparison with the models in the polar coordinate system, we estimate the distance and angle using the absolute positions using the formulas mentioned below.

$$\text{distance} = \sqrt{x^2 + y^2}$$
$$\text{angle} = \arctan\left(\frac{y}{x}\right)$$

where x and y are predicted cartesian coordinates.

4.3.2.1 Without normalising input features

Table 4.5 shows the result summary for distance prediction based on cartesian models

Model	Min[m]	Max[m]	Std[m]	Mean[m]
Decision Tree	-4.450	2.270	0.486	-0.002
Random Forest	-3.620	2.270	0.414	-0.001
SVR	-3.620	4.780	1.075	-0.219
Polynomial	-3.669	4.499	1.138	-0.081

Table 4.5: Distance prediction performance metrics for non-normalised cartesian models

Table 4.5 presents the occurrence of overestimation in distance estimation across all models, evident from the negative mean values. Notably, the SVR model exhibits the highest overprediction, indicated by its negative mean. Additionally, the standard deviation for models like SVR and polynomial fit is approximately twice that of the first two models, which have a standard deviation below half a meter. The range of values also exhibits a similarity between the last two models and the first two models. Among these, the random forest model demonstrates superior performance, showcasing the lowest standard deviation, mean, and range. Similarly, Table 4.6 summarises performance for angle prediction

Model	Min[deg]	Max[deg]	Std[deg]	Mean[deg]
Decision Tree	-340.600	223.830	12.082	-0.238
Random Forest	-347.120	36.600	10.363	-0.325
SVR	-317.88	40.090	11.254	-0.264
Polynomial	-340.590	223.829	12.082	-0.238

Table 4.6: Angle prediction performance metrics for non-normalised cartesian models

Table 4.6 shows overprediction in angle estimation from every model with a mean of around -0.27 degrees. The standard deviation also seems consistent for every model like the mean with a value of around 11 degrees. The range of error seems large for every model due to wrong sign estimation of coordinates.

4.3.2.2 Normalising input features

Table 4.7 shows the result summary for distance prediction based on cartesian models with normalised inputs

Model	Min[m]	Max[m]	Std[m]	Mean[m]
Decision Tree	-4.450	2.930	0.489	-0.007
Random Forest	-3.560	2.270	0.416	-0.001
SVR	-3.730	2.640	0.499	0.012
Polynomial	-3.780	2.609	0.639	-0.052

Table 4.7: Distance prediction performance metrics for normalised cartesian models

Table 4.7 suggests the range of distance predictions varies across the models, with the decision tree model exhibiting the widest range. While all models show mean predictions close to zero, indicating a tendency to align with actual values on average, the random forest model boasts the lowest standard deviation of 0.416 meters, reflecting its consistent and precise predictions. In contrast, the Polynomial model presents the highest standard deviation of 0.639 meters, suggesting greater variability in its distance predictions. Overall, the random forest model emerges as a promising choice, with its narrower range, minimal deviation from zero mean, and low standard deviation, signifying reliable and accurate distance predictions. Similarly, Table 4.8 shows summary for angle prediction

Model	Min[deg]	Max[deg]	Std[deg]	Mean[deg]
Decision Tree	-340.600	223.830	11.805	-0.071
Random Forest	-343.310	37.550	10.325	-0.314
SVR	-341.03	26.889	10.516	-0.296
Polynomial	-340.590	223.829	11.805	-0.071

Table 4.8: Angle prediction performance metrics for normalised cartesian models

Table 4.8 shows overprediction in angle estimation from every model with a mean of around -0.18 degrees across models. The standard deviation also seems consistent for every model like the mean with a value of around 11 degrees. The range of error seems large for every model due to wrong sign estimation of coordinates.

4.3.3 Best performing model

It is important to highlight that angle prediction achieves a higher level of accuracy when employing the polar model, whereas the performance for distance prediction remains consistent in both scenarios. The corresponding heatmaps (Figure 4.5 for distance and Figure 4.7 for angle) depict the distribution of errors across test data points for the random forest polar model without input normalisation. Further analysis has focused on the non-normalised random forest polar model, as it stands out as the clear winner for distance prediction and shares the top spot with the polynomial fit model for angle prediction. However, it's worth noting that the polynomial fit model exhibits strong performance with normalised input, which may be seen as a limitation due to its dependency on dataset size and comprehensiveness during the normalisation process. On the other hand, the random forest model

demonstrates resilience to input normalisation, yielding consistent results in both scenarios. The parameters used for this model have been listed under section 4.2.

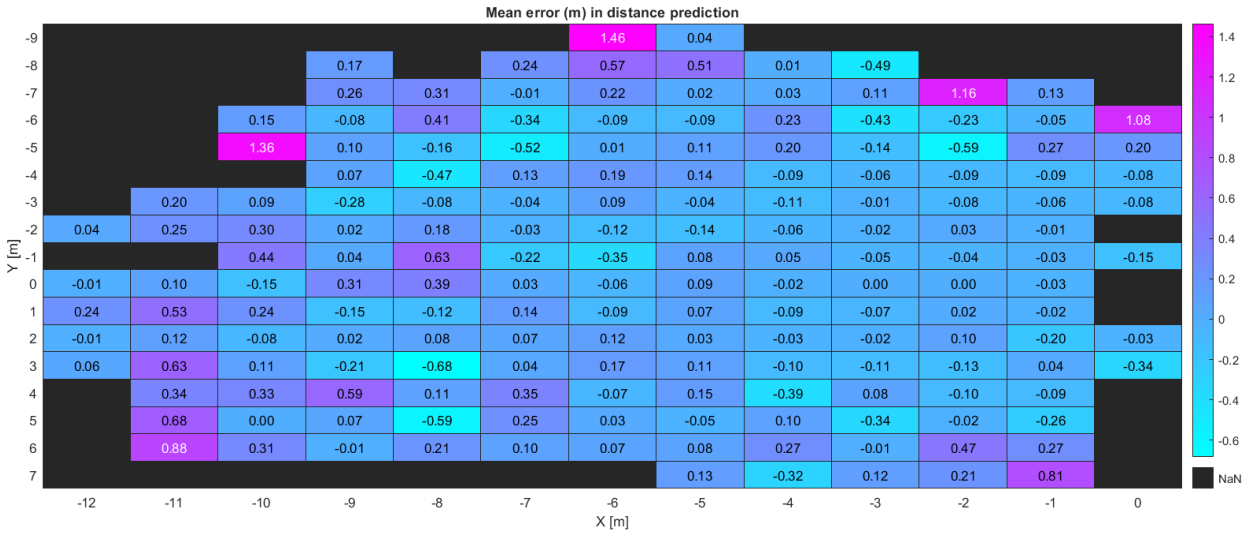


Figure 4.5: Heat map showing error distribution in distance for non-normalised polar model

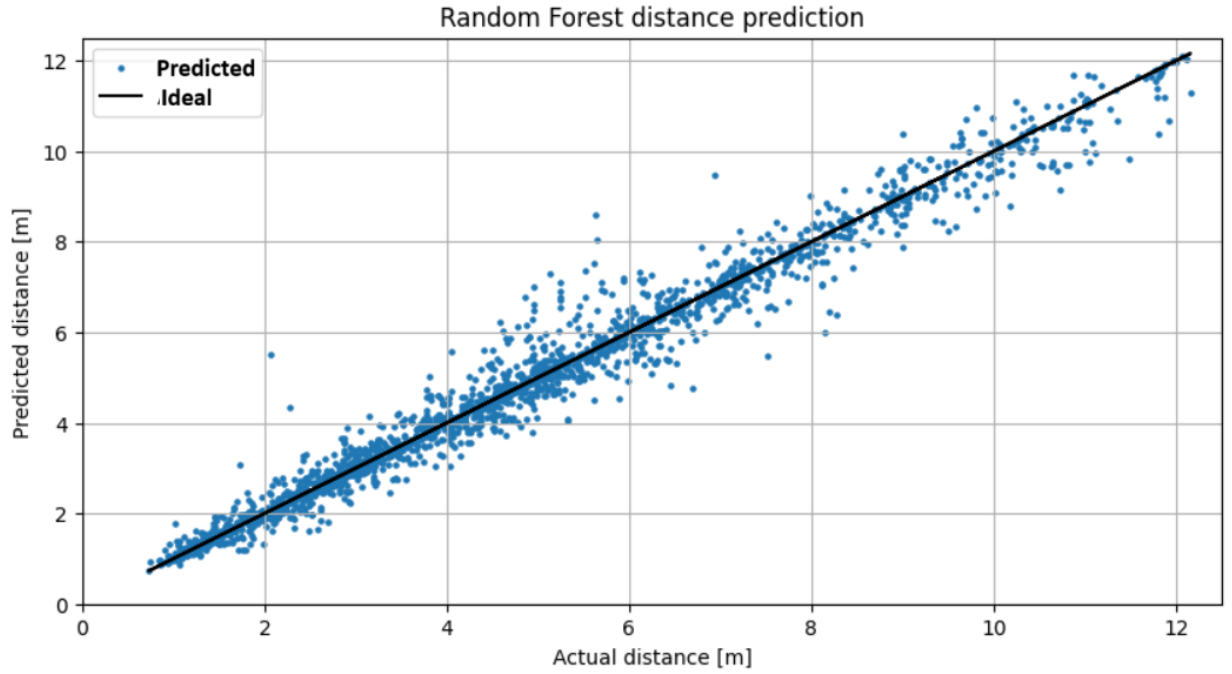


Figure 4.6: Random forest non-normalised polar model distance prediction vs ground truth

Examining Figure 4.5, it becomes apparent that the mean error tends to be near zero within the region encompassing a distance of 6 meters and non-extreme angles. Notably, outliers are predominantly situated at greater distances or at angles considered extreme. This observation can be attributed to the limited availability of

training data within these specific domains. Upon analyzing Figure 4.6, it becomes apparent that the highest absolute error is concentrated at shorter distances, particularly around 2 meters. However, due to the substantial volume of data points within this range, the mean value is moderated, as evidenced by the heatmap displayed in Figure 4.5.

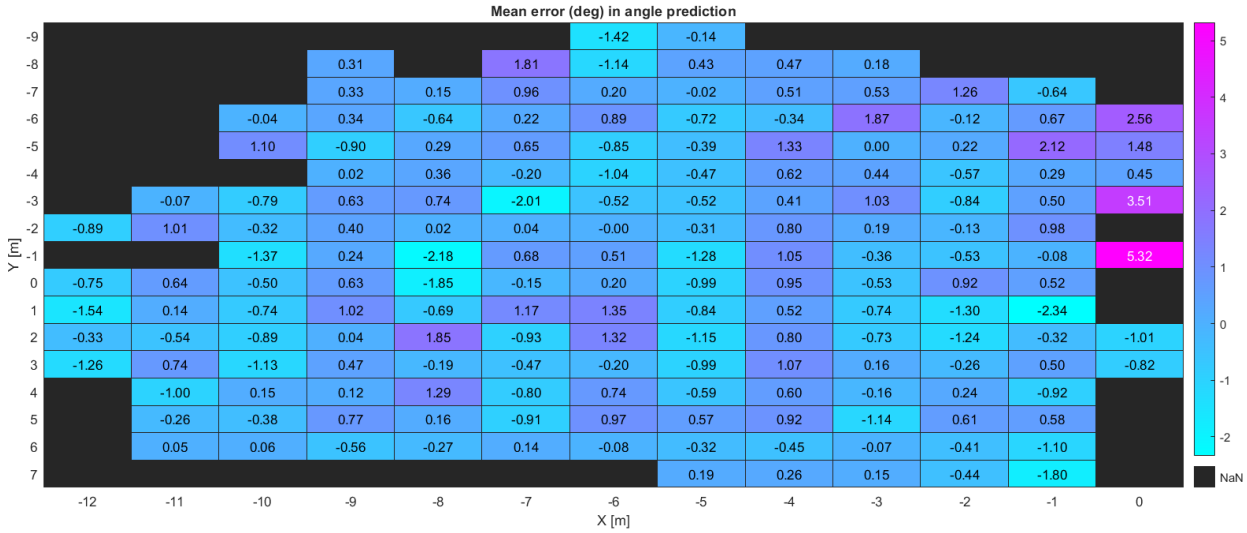


Figure 4.7: Heat map showing angle distribution in angle for non-normalised polar model

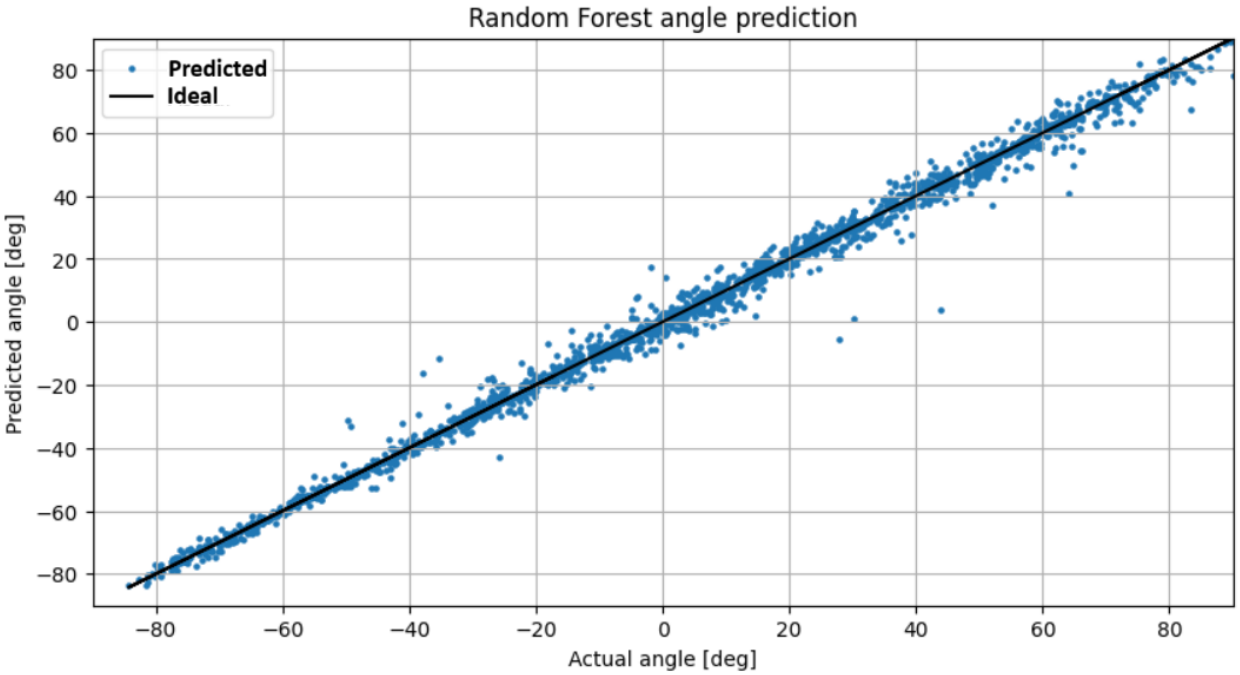


Figure 4.8: Random forest non-normalised polar model angle prediction vs ground truth

Analyzing Figure 4.7, it becomes evident that a consistent pattern of prediction er-

rors is not discernible across the entire domain. Nevertheless, the majority of points with higher errors are concentrated within a 2-meter distance range at extreme angles. This phenomenon is reasonable, particularly for points closer to the camera, as the angle becomes considerably more sensitive to pixel variations at shorter distances. Consequently, even a slight deviation in input can result in a substantial change in output. Strikingly, some outliers are apparent at moderate distances, around 8 meters, and approximately zero angles. These outliers pose a challenge to explain. Notably, an observable concentration of high-error points is observed on the right side of the camera (negative Y-axis direction), where the training dataset is comparatively less extensive due to data collection protocols. Upon examining Figure 4.8, it becomes evident that outliers are not predominantly situated at extreme angles, but rather at moderate angles. This phenomenon becomes apparent in the heatmap presented in Figure 4.7, where the values are attenuated due to the averaging of a substantial volume of data points at moderate angles. This contrasts with the scenario involving extreme angles, where the mean is not as greatly influenced by absolute error. Thus, although the absolute error appears lower, the overall mean remains proximate to the absolute error due to this effect.

5
Discussion

This thesis endeavors to explore diverse machine learning models for extracting the position of identified pedestrians, including distance and angle from a fisheye lens camera affixed to an escooter.

In pursuit of these objectives, the research engaged in multiple data collection endeavors. Video recordings captured visual data, while lidar sensor gathered spatial coordinates of surrounding objects, establishing a ground truth reference for model training. Various regression models, such as decision trees, random forest, SVR, and a polynomial-fitted linear model, were employed. Two primary approaches emerged for predicting the desired position output. The first involved segmenting and tracking a pedestrian cluster's centroid from lidar-derived point cloud data, subsequently utilizing these coordinates for prediction. Alternatively, the lidar data was pre-processed to derive ground truth values in terms of distance and angle, directly training the model on the modified ground truth.

To enhance model performance, data preprocessing techniques, including normalisation, were implemented. Normalisation aids in mitigating input feature biases arising from differing value scales. Both approaches, utilizing raw and pre-processed lidar data, were adopted and will be discussed in this section.

5.1 Region of interest

The idea involved establishing the region of interest based on the capabilities of the employed sensors and tools within the subsequent pipeline. It was discovered that the utilization of a fisheye lens in the camera led to substantial image distortion around the edges (-90 to -70 degrees and beyond 70 degrees). This distortion severely impacted the efficacy of object detection within this range, resulting in a sparse dataset in this specific domain. This observation consequently imposes a limitation on the range of angles considered.

Furthermore, the pedestrian detection algorithm exhibited reduced accuracy for distances exceeding 6 meters, especially at extreme angles as can be seen in the polar plot in Fig.4.1 showing the training dataset, and this detection accuracy further deteriorated beyond 11 meters. This limitation could potentially arise from the camera sensor's lower resolution or the detection algorithm itself. As a result, data

points beyond the 11-meter mark are notably limited in quantity. Given these constraints and observations, we can now formally define the bounds of our region of interest.

5.2 Machine learning models

An extensive exploration was undertaken involving four distinct machine learning regression models to effectively uncover and predict inherent patterns within the dataset. The first model, a **decision tree**, operates through the application of a selection criterion that evaluates the quality of data splits based on the "criteria" parameter. This criterion, in combination with the complexity parameter for Minimal Cost-Complexity Pruning denoted as "ccpalpha" and the maximum depth of the tree, forms the foundation of the model. Notably, this model demonstrated a marked susceptibility to overfitting, particularly with default parameters. Although the training score was perfect, its performance diminished when applied to unseen test data points, as evidenced in the results section. Through parameter tuning, restricting the tree's depth to 10 showed improvements in generalization, albeit at the expense of reduced fitting quality. Altering the criteria yielded minimal impact, while the default ccpalpha value of 0 appeared to provide the best combination in terms of performance metrics. It is important to note that this model did not perform optimally for any case.

Proceeding onward, the **random forest** model, a potent meta estimator, harnesses an ensemble of regression decision trees trained on distinct subsets of data. The model leverages an averaging technique to enhance predictive accuracy while simultaneously mitigating the risk of overfitting. Parameter tuning for this model encompasses variables such as the number of trees within the forest (defaulting to 100), the criteria for splitting nodes (defaulting to squared error), the maximum depth of individual trees (defaulting to expanding nodes until all leaves are pure), and the complexity tuning parameter ccpalpha (defaulting to 0). A refined combination of these parameters emerged through iterative tuning and trial and error. After thorough evaluation of performance metrics, the optimal parameter values were established: 300 trees, the default split criteria, a maximum tree depth of 20, and the complexity parameter defaulting to 0. Notably, varying the split criteria yielded minimal impact, increasing the number of trees marginally improved performance, limiting tree depth counteracted overfitting tendencies, and any value other than 0 for the complexity pruning parameter resulted in a decline in overall model performance. Impressively, this model exhibited superior performance across all evaluation metrics, encompassing mean error, standard deviation in error, and the range of errors for both distance and angle predictions in both cartesian and polar domains.

The **SVR** model, characterized by its incorporation of a range of modeling parameters including kernel algorithms, polynomial degrees, and regularization factors, demonstrated its effectiveness in Y-coordinate predictions when fitted with cartesian coordinates. It is noteworthy that the model consistently excelled in angle predic-

tion, if not in distance prediction. The regularization parameter, a critical factor inversely linked to the intensity of regularization, played a pivotal role in enhancing generalization, showcasing its impressive ability to surpass the default unity when set at 50. The "rbf" kernel emerged as the best-performing option, aligning with the default choice. Intriguingly, subsequent incremental increases in kernel-related parameters did not yield substantial improvements. Moreover, the model displayed a higher degree of compatibility with normalized input, leading to superior performance compared to raw data. In sum, this model consistently demonstrated a tendency to underestimate in the polar domain, evident from its positive mean error.

On the other hand, the **linear model with polynomial transformation**, characterized by its utilization of a polynomial degree of 3, demonstrated relatively subpar performance across various scenarios, except for angle prediction in the polar domain with normalized inputs. Notably, this model showcased a minimal decline in scores when transitioning from training to test data, distinguishing it from other models, but it proved to be less resilient in practical applications. Specifically, its performance closely mirrored that of SVR when predicting distance and angle directly. However, this performance experienced a slight decrease when considering the alternative approach. Similar to the SVR model, normalizing the input data played a pivotal role in significantly enhancing the results. Notably, this model displayed an underestimating tendency in the polar domain.

After a comprehensive assessment, the **random forest** model emerges as the optimal selection. Particularly, in the polar domain without normalizing input, it stands out as the prime candidate, showcasing the ability to predict distance with a relatively moderate error margin of around 0.4 meters and an angle error of 3 degrees. This model, adept at capturing the intricate nuances of the dataset, exhibits a well-balanced performance that could align with the goals of the study.

5.3 Polar vs cartesian approach

When it comes to distance prediction, the choice between direct distance prediction, referred to as the polar approach, and calculating the distance based on predicted coordinates, known as the cartesian approach, yielded negligible differences in performance. However, for angle prediction, the impact on performance was considerably more pronounced across all models. Directly predicting the angle proves to be superior to angle estimation derived from predicted coordinates.

This distinction is reflected in the standard deviation of errors: approximately 3 degrees for the direct angle prediction (polar approach) versus around 10 degrees for the coordinate-based estimation (cartesian approach) in addition to extreme errors (min-max, as shown in 4.6 and 4.2) in angle prediction across all models irrespective of the normalisation. As a result, the overarching recommendation leans toward the adoption of the polar approach, where the model directly fits the desired output. This method demonstrates superior performance and accuracy in predicting angles.

5.4 Feature engineering

In the course of data preprocessing, strategies to enhance model performance have been implemented, including the removal of duplicate values to mitigate potential overfitting. Additionally, the application of input feature normalisation has been a central facet.

The impact of normalisation is noteworthy, particularly for models like SVR and the linear model with polynomial fitting. These models experience discernible enhancements in prediction performance following input normalization. Conversely, the effect on models such as random forest and decision trees is less pronounced.

5.5 Conclusion

The thesis results demonstrate the feasibility of calibrating a monocular camera equipped with a fisheye lens to extract positional information using machine learning regression models. This can be achieved with an accuracy of approximately 0.5 meters for distance prediction and roughly 3 degrees for angle prediction, even without prior knowledge of camera parameters. It is worth noting that this level of accuracy is attainable even when training the models on a relatively limited dataset, consisting of approximately 8,500 data points. To put this into perspective, this dataset size is roughly half the size of the labeled data used to train convolutional neural network (CNN) models on KITTI datasets [16], which typically contain around 16,000 images for both training and testing.
Across various model approaches, encompassing both polar and cartesian models, with and without input normalisation, the random forest polar model without input normalisation emerges as the top performer. Interestingly, for random forest models, the impact of input normalisation on performance is not substantial, but the decision to avoid normalisation is deliberate because it relies on the dataset's size, which can introduce dependencies. Furthermore, expanding the training dataset has been observed to enhance model performance. However, achieving uniform data distribution across the entire grid can be challenging, particularly at greater distances (beyond 10 meters) due to reduced object detection because of low camera resolution and at extreme angles (beyond 70 degrees in either direction) due to pronounced distortion caused by the fisheye lens in the camera.

6
Future Work

This section presents recommendations for future research endeavors that can build upon the foundational work presented in this thesis, aimed at advancing the overarching objective of enhancing active safety through a comprehensive understanding of traffic interactions and the modeling of drivers' behaviors.

Central to the success of these future endeavors is the extension of the proposed methodology beyond the scope of escooter-pedestrian interactions. It is crucial to encompass a diverse range of road users, including not only pedestrians and escooters, but also other critical entities such as motor vehicles, trucks, and bicycles. The algorithm developed should be versatile and adaptable to different types of road users, allowing for comprehensive kinematics extraction and analysis.

Once a robust and adaptable algorithm has been established, it opens the door for in-depth analysis on a larger scale. Subsequently, this model will be deployed on naturalistic video data to detect critical incidents like collisions or near misses. Such occurrences will then undergo further scrutiny, seeking to unveil underlying patterns that illuminate driver behavior and the triggers for such critical events. These scenarios warrant thorough investigation to uncover underlying contributing factors and patterns, shedding light on root causes that can inform safety policies.

Furthermore, a compelling avenue for future exploration lies in the realm of modeling drivers' behavior. Expanding on the current framework, the scope could include not only the actions and responses of drivers to various road users' behaviors but also the assessment of driving style variability. By reducing the entire video content and focusing on drivers' interactions, one can gain insights into how drivers react, the prevalence of safe responses, and common patterns in driving behavior.

In summary, the suggested avenues for future exploration aim to expand upon the overarching goal of enhancing active safety through a comprehensive comprehension of the fundamental factors behind critical incidents during everyday road user interactions and variations in driver behavior. This entails delving into the extensive dataset provided by naturalistic video recordings to make strides in road safety and influence forthcoming policy interventions.

Bibliography

[1] *Active Safety.* URL: https://www.safeopedia.com/definition/8251/active-safety.

[2] Jace Allen et al. "Testing Methods and Recommended Validation Strategies for Active Safety to Optimize Time and Cost Efficiency". In: *SAE Technical Paper* (2020). DOI: 10.4271/2020-01-1348.

[3] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision.* Addison-Wesley, 1992.

[4] IBM. *What is Computer Vision?* URL: https://www.ibm.com/topics/computer-vision.

[5] Nina Nuyttens (Vias institute). *European Commission (2020) Facts and Figures Pedestrians. European Road Safety Observatory. Brussels, European Commission, Directorate General for Transport.* URL: https://road-safety.transport.ec.europa.eu/system/files/2021-07/facts_figures_pedestrians_final_20210323.pdf.

[6] Andrew James. *What is a fisheye lens and when would you use one?* URL: https://www.digitalcameraworld.com/features/what-is-a-fisheye-lens-and-when-would-you-use-one.

[7] MathWorks. *Segment Point Cloud into Clusters Based on Euclidean Distance.* 2023. URL: https://www.mathworks.com/help/vision/ref/pcsegdist.html (visited on 08/25/2023).

[8] Mathworks. *Segment ground points from organized lidar data.* URL: https://se.mathworks.com/help/vision/ref/segmentgroundfromlidardata.html.

[9] Anton Morgunov. *Object Detection with YOLO: Hands-on Tutorial.* 2023. URL: https://neptune.ai/blog/object-detection-with-yolo-hands-on-tutorial.

[10] Euro NCAP. *Vulnerable Road User (VRU) Protection.* URL: https://www.euroncap.com/en/vehicle-safety/the-ratings-explained/vulnerable-road-user-vru-protection/.

[11] M. A. O'Connell. *Introduction to Mobile Robots.* 3rd. MIT Press, 2017.

[12] Rahul Rajendra Pai. *Logging Data From E-Scooters To Improve Traffic Safety.* 2022. URL: https://hdl.handle.net/20.500.12380/305446.

[13] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[14] R. Smith. "An Overview of the Tesseract OCR Engine". In: *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007).* Vol. 2. IEEE. 2007, pp. 629–633.

[15] Department for Transport UK Govt. *National statistics Reported road casualties Great Britain: e-Scooter factsheet 2021.* URL: https://www.gov.uk/

government / statistics / reported - road - casualties - great - britain - e - scooter - factsheet - 2021 / reported - road - casualties - great - britain - e - scooter - factsheet - 2021.

[16] Marek Vajgl, Petr Hurtik, and Tomáš Nejezchleba. "Dist-YOLO: Fast Object Detection with Distance Estimation". In: *Applied Sciences* 12.3 (2022). ISSN: 2076-3417. DOI: 10.3390/app12031354.

[17] Velodyne Lidar. *What Is Lidar?* URL: https://velodynelidar.com/what-is-lidar/.

[18] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. 2022. arXiv: 2207.02696 [cs.CV].

[19] WHO. *Road traffic injuries*. URL: https://www.who.int/health-topics/road-safety#tab=tab_1.

DEPARTMENT OF MECHANICS AND MARITIME SCIENCES (M2)
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY