CHALMERS

Drawbar Eye Identification and Guiding

TME180 Automotive Engineering Project 2024

MAHIN GARG
ANDREAS WINBO
BIYING LIU
SHIYI QIU
FELIX RENBERG

Department of Mechanics and Maritime Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden, 2024

Drawbar Eye Identification and Guiding
TME180 Automotive Engineering Project 2024

Supervisor: Fredrik Von Corswant, Revere Lab
Supervisor: Tobias Johansson, VBG Group Truck Equipment AB
Examiner: Alexey Vdovin, VEAS

Studentarbeten – Mekanik och maritima vetenskaper (M2) – Projektarbete
Department of Mechanics and Maritime Sciences
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone +46 (0)31 772 1000

Acknowledgements

For the TME180 course, we the project members of **Group H** (focusing on Drawbar eye identification and guiding project) would like to express our deepest gratitude to our supervisors: Fredrik Bruzelius and Fredrik von Corswant from Chalmers University. They both have helped us by constantly supervising, assisting and helping us to analyze this project from various angles. We would also like to extend our appreciation to our examiner Alexey Vdovin for helping us with the project management aspects of the course.

Additionally, we would also like to express our gratitude to Tobias Johansson (Vice President at R&D, VBG Group Truck Equipment AB) for providing us with the opportunity to work on the project and supporting us technically in different aspects of the same.

It is through consistent support, guidance, and dedication of all these people which has played a crucial role in the successful development and advancement of the drawbar eye identification and guiding project.

Abstract

The purpose of this research project is to study if a machine learning algorithm can utilize a video feed from a camera mounted on the back of a truck to detect a drawbar eye under varying environmental, instrumental and lightning conditions.

In order to achieve this, lab based and real world testing data was collected from a setup which simulated connecting a truck and a trailer. With these setup cases, different video feeds were collected and later, images were extracted from this feed, which were labeled for the machine learning algorithm.

In order to train the algorithm, YOLOv8n (You Only Look Once Version 8) was used, which is a real time object detection algorithm to identify objects in an image. This was used to classify the drawbar eye and its position in real time. In order to better guide the driver, distance between camera and drawbar eye was calculated using the pinhole camera principle, and here the angle of the camera was introduced. From the calculation, we had the horizontal distance (X), vertical distance (Y) and distance in depth (Z).

Two additional tests were done in order to verify the accuracy of distance and height calculation, and take the distortions of a fisheye camera into account while calculating the said distances.

The final detection accuracy (ability to detect position of drawbar eye with 50-95 percent accuracy of the labeled position) of the model came at around 75.4 percent.

Contents

List of Figures

1
Introduction

Connecting a drawbar trailer to a truck is a challenge for both novice and experienced drivers. The coupling point is often not visible from the driver's position. The coupling requires high precision and can become a challenge in various environmental conditions with issues like poor visibility, bad weather and tight areas.

This section highlights the background for the problem statement, mentions the specific aims and and objectives which are required to be completed, and also some of the potential limitations regarding the execution and implementation of this project.

1.1 Background

The current solution on the market from the VBG Group includes a radar systems to provide audible driver assistance. However this system is heavily impacted by different factors such as dirt and most importantly, requires modification of the trailer (addition of 6 radar reflectors on the drawbar) for the system to detect the drawbar eye.

This project is supposed to use machine learning together with image recognition to explore if it is a suitable solution for drawbar eye identification. Then later use this together with an HMI system to provide audible feedback to a driver to assist in the connection between truck and trailer. The system itself is supposed to learn and improve after more data is collected to be able detect the drawbar eye easier in the future. Since the system will be improved when adding a more versatile dataset it should be better at handling problems such as dirt, snow and different variations of tow-bars in different conditions.

1.2 Aim and Objectives

The aim of the project is to develop a proof on concept (POC) that provides audible assistance in the connection between truck and trailer. This system consists of a image recognition system that is supposed to recognize and detect the drawbar and an HMI that utilizes the location of the drawbar to provide guidance to the driver for a simpler connection process.

Objectives:

- Collect data for image recognition training and validation.
- Create a model using YOLO(You Only Look Once) for detection of the drawbar.
- Create a program that can calculate the real-time position of the coupling in relation to the drawbar.
- Create a HMI to give instruction for the coupling procedure.

1.3 Limitations

It is important to define limitations to the project to keep the work relevant and the goal achievable within the time-frame of the required deadlines. The key limitations for the project are listed below:

- The project timeline is limited to around four and a half months from September to January. This means that we need to have a working prototype by mid-December before the presentation, thus limiting any additional functionalities or features to try and test on the product due to limited time-frame.
- Since the concept is supposed to be leaned to fulfill functionality in all conditions so the aesthetics will not be prioritized.
- The project scope will be limited to the types of drawbar shapes and sizes mentioned by the VBG Group (focusing on three different shapes and sizes of 50 mm and 57 mm for the drawbar eyes).
- The amount of pictures for creating a dataset for the machine learning is limited by the videos we can capture during testing and other factors.
- The system only focuses on detecting the drawbar within 2-3 meters from the coupling point, so a certain amount of driver input is expected before the system is activated.

2
Theory

Before we started the project, we first needed to be well versed with the different aspects of the project, ranging from equipment type, softwares and algorithms, physical equipment to name a few. This chapter gives an overview of the theoretical background of these components.

2.1 Drawbar

A drawbar is a key part of towing setups, connecting a trailer to the vehicle pulling it. It usually has a sturdy loop or hole that fits over a hitch pin or ball, creating a secure link between the trailer and the towing vehicle. Drawbars can often be seen on farm equipment, trailers, and various types of heavy machinery. This design allows for quick and easy hookups and releases, making it ideal for safely moving cargo and equipment from place to place. A important part of the drawbar is the drawbar eye that which is the part where the connection between a truck and trailer happens. Figure 2.1 shows how a typical drawbar looks that are located on the trailer[1].

Figure 2.1: Picture of a typical drawbar. (Image by VBG Group)

2.2 Fisheye Camera

There are two types of lenses for photography - rectilinear and curvilinear. Rectilinear are a lenses that captures straight lines like how our human eyes sees. Curvilinear lenses has a strong distortion which gives a wider perspective over the world. A fisheye camera is a camera which uses ultra-wide angle curvilinear lenses, that are used for creating a panoramic view [2]. The images that the fisheye lens gives is extremely wide compared to cameras with a rectilinear lens. The wide view comes from that the lens creates a strong visual distortion, which can make it a bit tricky to easily visualize distances in an image. However, there are certain tools which can be used to un-distort such images [3]. Figure 2.2 shows a picture of a panoramic view taken by a fisheye camera.

Figure 2.2: Panoramic view of a fisheye lens.

2.3 Detection Model

For this project, machine learning was used for the rear-mounted fisheye camera to identify the drawbar eye in real-time. In machine learning, the detection model serves as a cornerstone, playing an important role in the overall performance of the system.

YOLOv8n was selected for this project because it combines high-speed, real-time

performance with excellent detection accuracy. Its lightweight architecture makes it efficient and suitable for resource-limited environments, and its robustness makes it adaptable to a wide range of detection tasks.

YOLOv8n is a real-time object detection algorithm for identifying objects in images. It uses a Convolutional Neural Network (CNN) to extract features from the input image. CNN is a type of deep learning model that is particularly effective for image processing tasks. It works by applying convolutional layers that use small filters to scan an image and detect patterns such as edges, textures and shapes. These patterns are combined and refined over multiple layers to capture more complex features [4, 5].

In YOLOv8n , the extracted features from the CNN are analyzed to predict bounding boxes and confidence scores for objects in the image. The advantage with this is that is eliminate the need of going through the image several times. Another important thing for YOLO is the use of regions of interest. This is predefined bounding boxes that works as references for the predictions. The performance and the accuracy of YOLO is what is standing out for it. YOLO is using a pipeline approach and this allows it to handle objects of different shapes which makes it very robust for different situations.

Varying object scales refer to the differences in the size of objects as they appear in images, depending on their actual dimensions or distance from the camera. This variability provides a challenge to detection Model because there is a need for detection Model to accurately detect large and small objects by processing features at multiple scales.

Distinct target features are unique patterns, shapes, or textures that make an object recognizable. If these features are not distinctive, it becomes more difficult for the model to recognize the object.

Ordinary convolution is a standard operation in Convolutional Neural Networks that involves sliding filters across the entire input image to extract features like edges and textures. This method is effective but it is expensive to compute. Partial convolution applies filters to only selected regions of the input image, thus the computational requirements are reduced while the essential features are captured.

The SE attention mechanism provides additional enhancement to the detection capability, especially in complex environments, that is achieved by helping the model to focus on the important regions of the image and reducing the background noise interference.

A P2 layer is a detection layer which improves the detection of small objects by capturing richer semantic and spatial information. The neck is the part of the model that connects the backbone network for extracting features to the detector heads that predict the location and class of objects.

In YOLOv8n, there are several main parts, the backbone is responsible for extracting basic features from the input image such as edges and texture. The neck is to refine and combine these features to enhance the ability of the model to detect objects at multiple scales. The head generates the final prediction including the location of the object and its classification.

The CSPPC module divides the convolution process into stages which reduce computational cost while maintaining the quality of the extracted features [6].

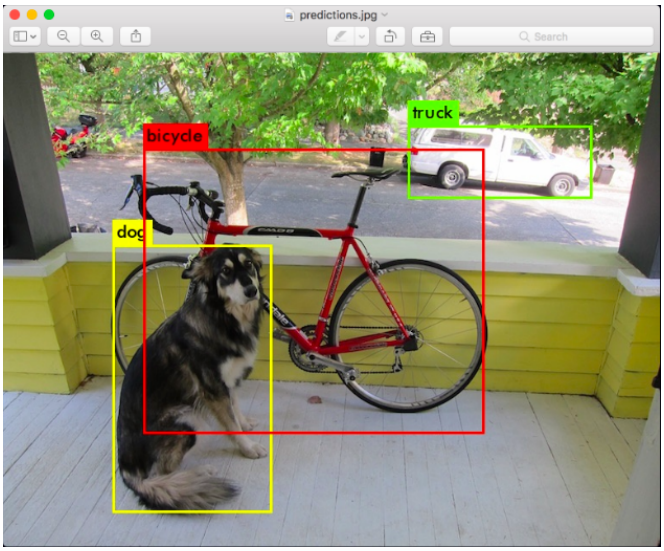

Figure 2.3: Image from YOLO predictions. (Image by Aditya Sharma)

2.4 LabelImg: An Annotation Tool for Object Detection

LabelImg is a tool used to label objects in images. It helps create datasets for tasks like object detection and recognition. Users can draw bounding boxes around objects in an image and give each box a label to classify it. This labeled data is important for training models like YOLO, which require labeled images for learning. LabelImg works well with YOLO, making it a popular choice for object detection projects. The tool is simple and easy to use, with a clear interface that makes annotation straightforward. However, it does have some limitations, such as no automatic bounding box suggestions and limited features to handle large datasets quickly [7].

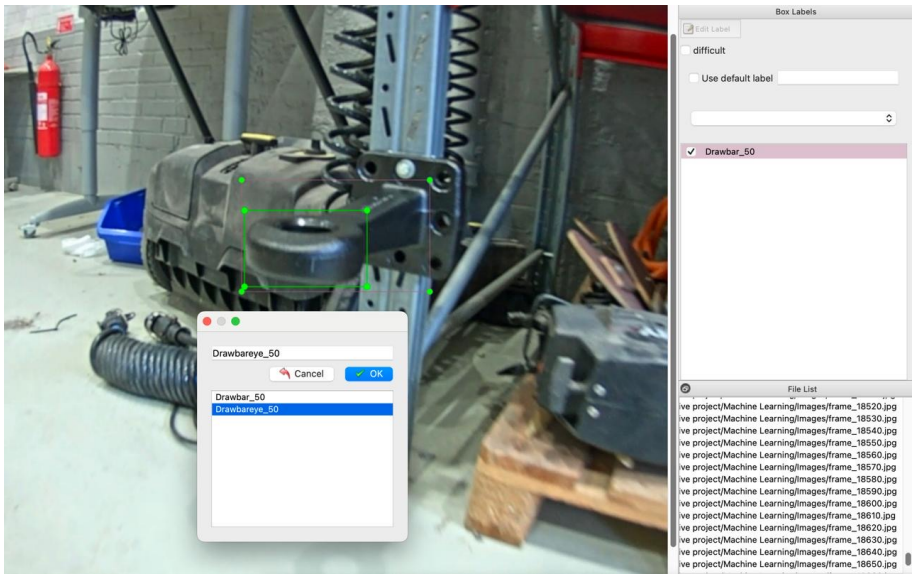

Figure 2.4: labelimg software interface.

2.5 Calibration board

A calibration board is a tool used for camera calibration. It usually has a checkerboard pattern. The pattern has clear points that the software can detect. These points help calculate the camera's intrinsic parameters and distortion coefficients. It also helps to correct image distortion. The checkerboard pattern provides clear, well-defined points that the calibration algorithm can detect and use to map the relationship between 3D points in the real world and their 2D projections in the camera image.

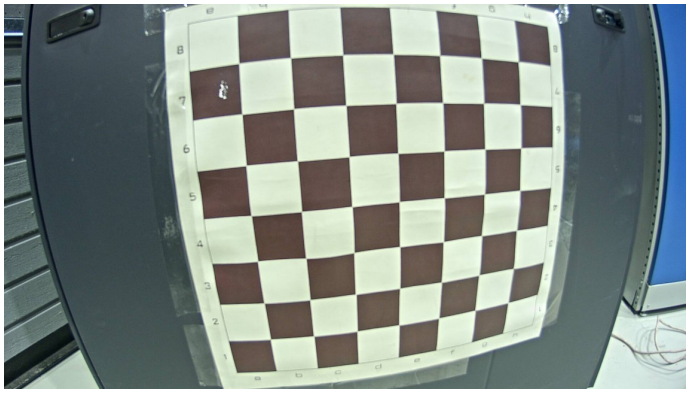

Figure 2.5: Calibration board.

2.5.1 Fisheye Camera Calibration

To use codes to calculate the intrinsic matrix and distortion coefficients, several steps are required.

First, the 3D points of the checkerboard pattern are defined. These points represent the actual coordinates of the corners on the calibration board. Then, the 2D points are obtained by detecting the corners of the checkerboard in the images. Corner detection identifies the exact pixel locations of the pattern in the image.

Second, the cv2.fisheye.calibrate function is used. This step calculates the intrinsic parameter matrix which includes the camera's focal length and optical center, the distortion coefficients which describe how the image is distorted, the rotation matrix, and the translation vector which define the camera's position and orientation relative to the calibration board.

Third, the cv2.fisheye.projectPoints function is used. This function takes the calculated intrinsic parameters, distortion coefficients, rotation matrix, and translation vector to project the 3D points onto the 2D image. It simulates how the 3D points in the real world appear in the camera image.

Finally, the backprojection error is calculated. This involves comparing the projected 2D points obtained from the calibration process with the actual detected 2D points in the image. The error helps determine how accurate the calibration process is.

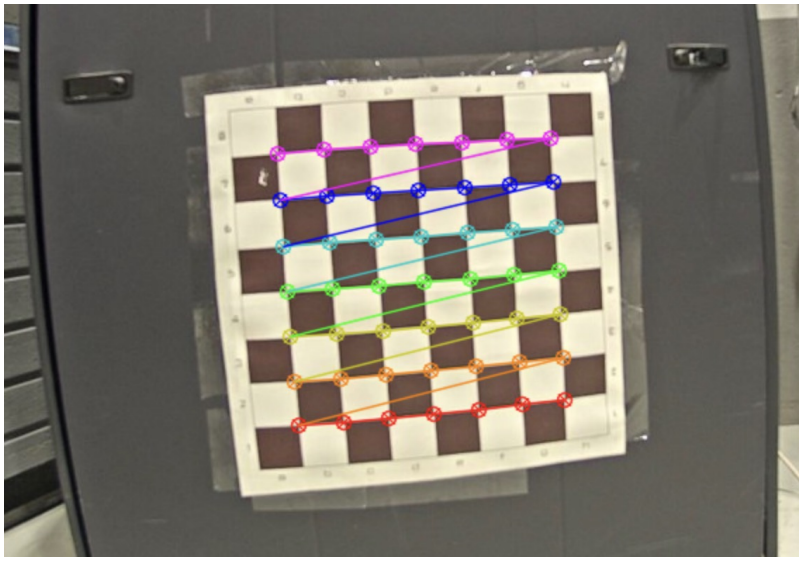

Figure 2.6: Fisheye camera calibration.

2.6 Pinhole camera principle

In computer vision, the pinhole camera model is used to mathematically describe how a camera captures an image. It uses projection equations to map 3D points in the real world to 2D points on an image. The model is defined by intrinsic parameters such as focal length and optical centre and extrinsic parameters such as camera position and orientation.

2.7 Human Machine Interface (HMI)

A Human Machine Interface is a way that a person can be connected to a machine or a system. A example of this is when using air condition to see and change the temperature in a room. There are three main types of HMI systems - visual, audio, and haptic systems. The systems are mainly used for a human to get output information from the machine or system. HMI systems are used today in roughly all industrial organizations. HMIs are important tools when technology is an important part of society [8].

3

Problem statement

Connecting a trailer to a truck is a task that has to be done a few times each shift for a truck driver. It can be a challenging task in various different scenarios even if the driver has a lot of experience. The connection process requires relatively high precision and due to the limited amount of visibility that normally only comes from the side view mirrors. The lack of view of the actual connection process, forces the driver to sometimes leave the truck and check multiple times before a successful connection can be made. Weather conditions such as dirt, snow, rain, and poor lighting make this process harder and more tedious for the driver. Having the driver make multiple attempts and leaving the truck to check the connection process not only tears down the drivers but also takes a lot of time as well as risking vehicle and property damage when the drivers become too lazy and do a few steps of the connection by their feel.

Figure 3.1: Truck hitch and drawbar.

The truck coupling and drawbar has to be fairly aligned in height and width. There are some error margins that the coupling can force the drawbar into the correct position and after this alignment, the truck can reverse to make the connection hap-

pen. Compared to a car connecting to a car trailer, where you just can go out and change the trailer manually. When connecting with a truck you cannot make these last second adjustment neither is it easy to pull of big changes when you are close to the trailer. So you will have to redo the whole connection again and repeat until it is good enough which makes a system that helps with process extremely useful for drivers.

One solution currently on the market offered by VBG utilizes a radar with reflectors. This solution requires permanent modifications of the trailer in advance of the connection, which makes the system harder to adapt when changing trailers constantly, since you would also have to apply these reflectors at the correct distances. This system works, are reliable and fairly simple but the permanent modifications as mentioned earlier makes the system less desirable and requiring them to be in good condition and put in advance of the connection process. These systems are also highly sensitive to weather conditions and dirt that can interfere with radar and reflectors which is something that might be easier to solve with a camera setup since you could easily have some kind of cover on the truck or make the system work around the obstructions. But when you also need to have permanent modifications on the trailers it will be hard to make sure they are clean as well.

So with our solution we want to create a system that does not require permanent modifications of the trailer and therefore can be adaptable to every trailer as long as it has a drawbar eye.

4
Methods

In order to guide drivers to connect the truck and trailer successfully with the machine learning-based camera system, it is crucial to figure out how to set up the camera, how to collect data, how to use the data, how to build the machine learning model, and how to integrate with Human Machine Interface (HMI).

4.1 Setup

This section explains the steps taken to set up the system, including decisions on camera positioning, camera type, and testing environments. It also covers how the lab and experimental setups were arranged to gather initial data.

4.1.1 Camera positioning

VBG gave us restriction on were the camera could be placed. Figure 4.1 shows in what region it could be placed on the coupling on the truck. The camera could be place between the red arrows and were the red circles are. This is areas that are known to be the same on all the trucks. The camera could not be placed where the x is (over the coupling) because parts could vary between different trucks and there is not necessary room for a camera in this position on all different trucks.

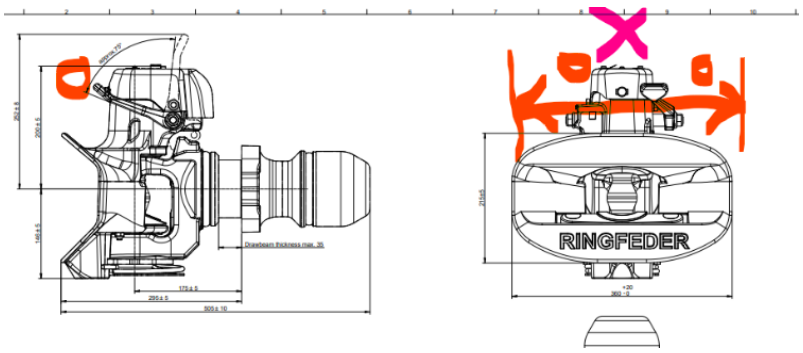

Figure 4.1: Camera possible positions.

4.1.2 Camera type testing

With the available cameras there was a choice to use either a camera with a rectilinear lens or a fisheye camera that has a curvilinear lens. To evaluate which one to use we made a test to measure the field of view for the cameras. The camera should be able to identify the drawbar eye from a range of between 2.5 meters away to within a few centimeters. This to be able to guide the driver to connect to the drawbar eye in a good way.

The rectilinear camera was first tested. Figure 4.2 shows how the camera was angled during the test. The angle was 25 degrees downwards since we wanted to be able to see the drawbar eye decently from 2.5 meters away but also up close and we found out while testing that around 25 degrees worked good. The angle of the camera is very important regarding from which distances the camera is able to see the drawbar eye from.

Figure 4.2: Camera angle.

For the first test, the coupling height on the truck was the same as the height of the drawbar eye. Here, the minimum distance that the camera could see the drawbar

eye was from 16 cm before being occluded by the coupling. For the second test, the coupling was moved 15 cm above the drawbar eye. With this change the closest that the camera could see the drawbar eye was 90 cm. This was not nearly good enough for the guiding of the truck. Therefore the choice became to use a fisheye camera instead even if that requires dealing with image distortion.

4.1.3 Lab Setup

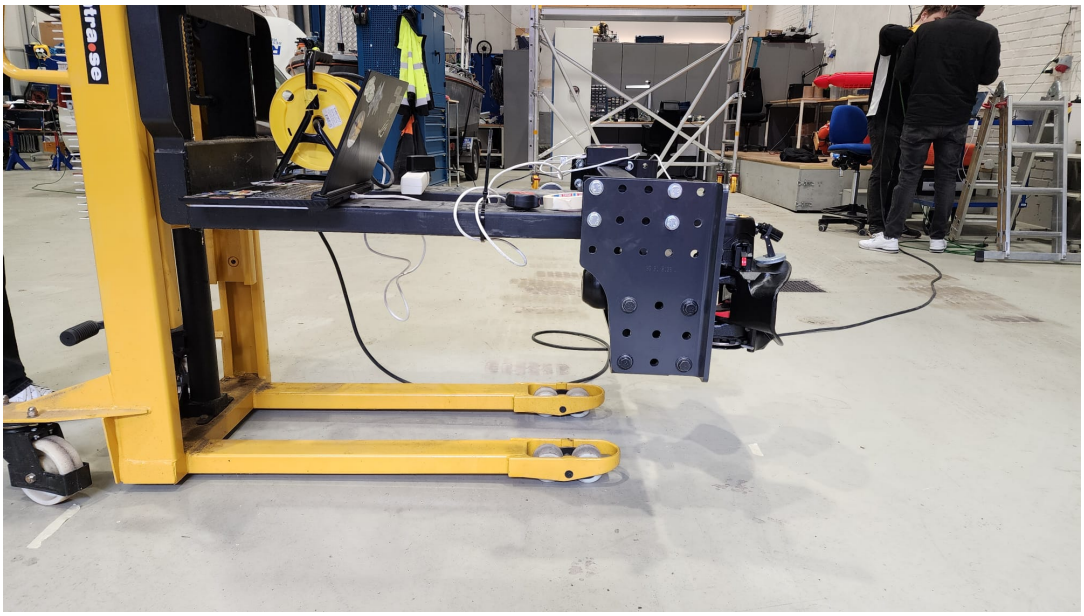

Figure 4.3: Lab Setup.

The Lab setup was made to be able to test the system. A fisheye camera was placed on top of the coupling and the whole coupling setup was mounted on a forklift with height adjustment to be able to simulate a truck with air suspension. The mounting of the camera was limited a lot by VBG and during testing we saw that a higher placement gave good results and since the mounting spot on highest point of coupling was restricted, this spot seen in the picture was the highest we could get. In the figure above can be seen an adjustable forklift with the whole coupling setup mounted on the forks. Camera mounted on coupling as well as computer, camera box and wiring on the forks of the forklift.

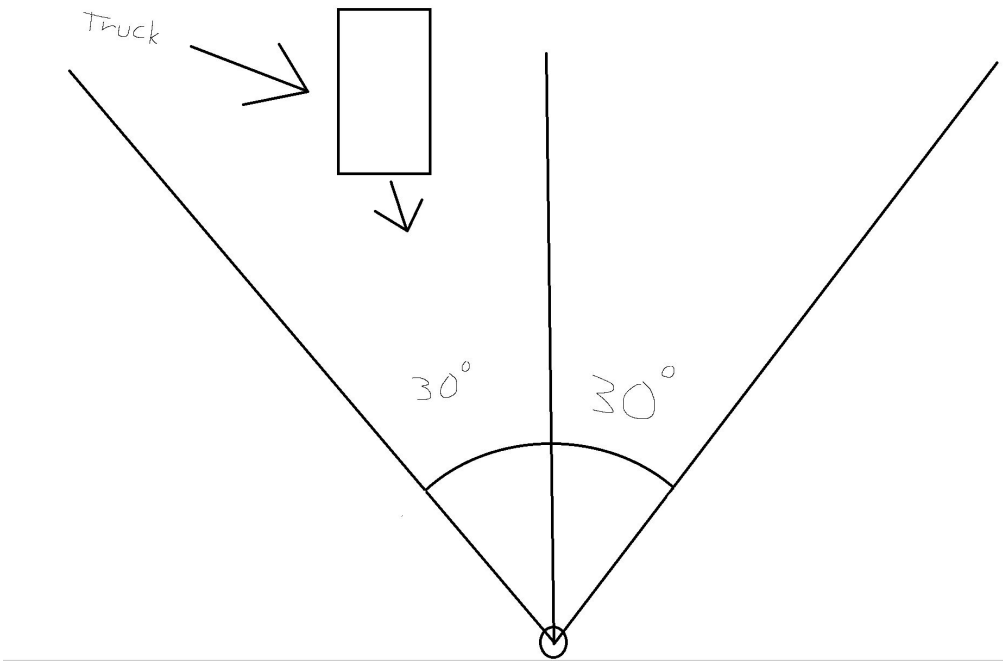

Figure 4.4: 60 degree cone

When recording our test videos that we took the frames out of for the machine learning algorithm we moved the whole setup in a 60 degree cone with the drawbar eye in the center as can be seen in the figure above. This was done to simulate different connection scenarios and give a diverse and broad view of different angles of the drawbar eye. A 60 degree cone was chosen since it would show some of the most extreme connection angles while most connections will either be straight on or just a few degrees off center.

This approach made sure that we got different frames of the drawbar eye so not all of them looked the same but it also made sure that we covered more angles of the drawbar eye to make the machine learning algorithm able to identify the drawbar eye from many possible approach angles.

4.1.4 Experimental setup

Figure 4.5: Outside Setup.

The results from the lab setup show that the frames had almost exactly the same background and with little to no variation. To make the machine learning algorithm "background independent", the drawbar eye needed to have different backgrounds, different scenarios in form of lightning and weather. To make this happen we mounted the setup outside on a real truck and collect the data we needed outside to get different weather conditions, dirt and various lightning conditions.

To further give variation to the picture frames we decided to utilize different types of trailers as well as trying to make the connection scenarios more realistic. To achieve this we hade a wider range of different motions towards the towbar such as different curved paths as well as having the air suspension starting out high/low and adjusting to correct height when moving towards the tow bar. We also created scenarios were we both hit the tow bar and missed to create life like scenarios of a driver successfully connecting and failing the connection.

4.2 Calibration tests

As stated in the previous section, a fisheye camera is used to capture video footage of the drawbar eye in this project. Therefore, it is necessary to calibrate the fisheye camera before its further use. In the machine learning-based camera system, the distances are calculated by algorithms once the drawbar eye is detected. A distance accuracy test is conducted to check whether the distances calculated by the algorithms match the actual distances.

4.2.1 Fisheye calibration test

To account for fisheye distortion, fisheye calibration should be performed. To ensure accurate and robust camera calibration, several considerations were made during the image capture process. The calibration board was clearly visible in all frames, to capture it from various angles and distances. This included top-down, bottom-up, and horizontal views to achieve a comprehensive distribution of calibration points. The calibration board should be fully visible within the frame, especially at all corners to maintain accuracy. Diverse poses of the calibration board were captured by varying its translation (moving it closer to and farther from the camera) and tilt (simulating different angles). The camera moved to the calibration board along three different directions at angles of -30°, 0°, and +30° relative to the horizontal centerline. Along each trajectory, the camera stops at specific distances from the checkerboard: 2.5 m, 2.0 m, 1.5 m, 1.0 m, 0.5 m, and 0.4 m (the closest possible distance). These steps were taken to enhance the robustness of the calibration process. The figure below shows the setup for capturing calibration images with a checkerboard.

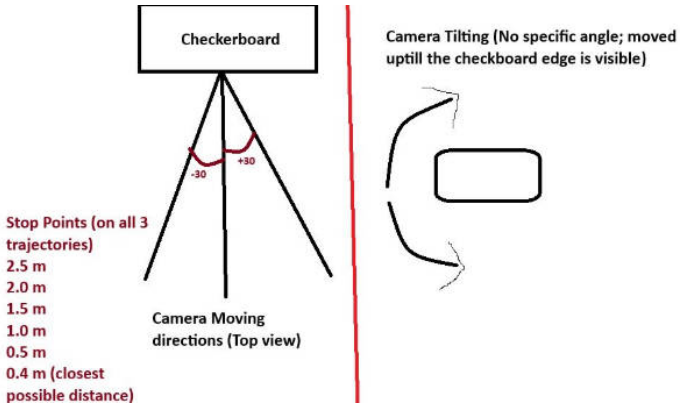

Figure 4.6: Calibration test setup.

4.2.2 Distance accuracy test

To evaluate the accuracy of the distances calculated by the algorithms, a distance accuracy test was conducted. The test was designed to find the difference between the predicted distance and the actual measured distance.

In the setup, the fisheye camera was mounted above the drawbar coupling which was fixed on a rig. This configuration ensures a stable and clear view of the drawbar eye. The camera was moved in three different directions, simulating real-world perspectives, at angles of -30°, 0°, and +30° relative to the horizontal centerline. These angles were chosen to simulate the situation where the drawbar eye may not always be directly aligned with the camera's view fields.

To collect data, the camera was gradually moved from approximately 3 m to 0.2 m from the drawbar eye. Images were captured at about every 0.5 m. Each image was annotated with the actual horizontal distance, vertical distance, and depth distance. This test ensures a diverse dataset that covers a range of distances and angles, which is important for evaluating the performance of the algorithm under different conditions.

Figure 4.7: Picture showing the actual distances.

5

Emperical Study

5.1 Data Collection and Labeling

To build an effective system, a large and comprehensive dataset of the 57 mm drawbar eye under several different conditions is needed. The video feed data was collected from the camera installed on truck. This data consists of video feed of the drawbar eye in various locations and lighting conditions, and from different distances and angles. After this process, images were extracted from the videos.

5.1.1 Lab based data collection

The first set of data was collected at the Revere lab as shown in figure 5.1. The data collection process in this environment is mentioned in figure 5.2. In this data collection process, 3 main routes (black lines) simulated the cases when the truck goes straight towards the drawbar eye. The angles of these potential routes were kept as +- 30 degrees, to simulate the most extreme offset angles from which a driver may attempt to connect to the trailer. In these cases the coupling and the drawbar eye were perfectly aligned to get a smooth connection.

Figure 5.1: Lab environment.

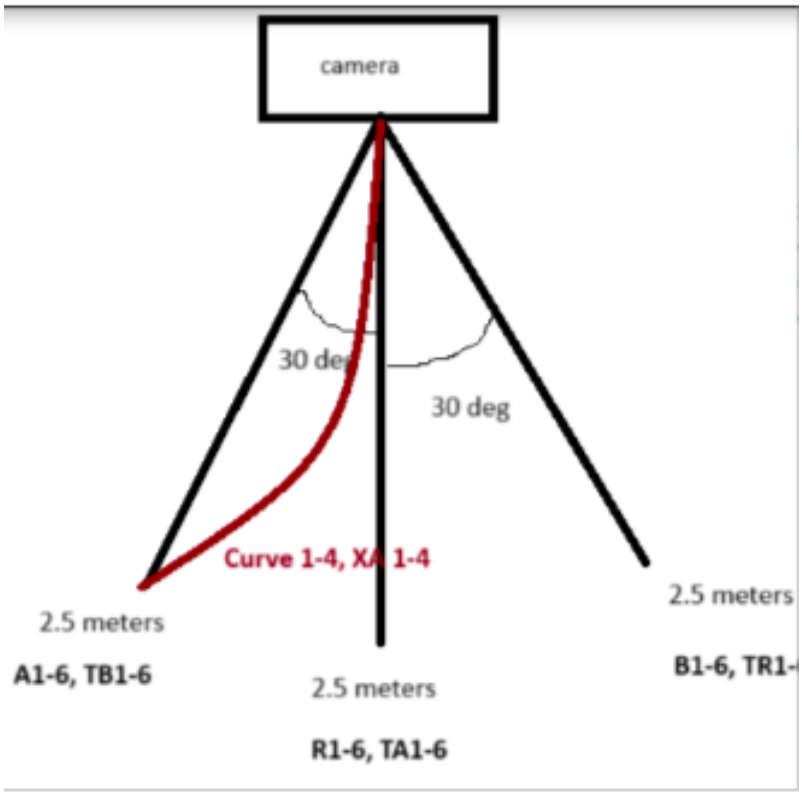

Figure 5.2: Image annotation procedure.

In order to cover other possible connection scenarios, additional videos were recorded along these 3 routes, but the height of the coupling point was moved higher up while retaining the existing height of the drawbar eye. This was done to simulate the cases when the coupling and the drawbar are not exactly aligned. The red curve in figure 5.2 describes the scenario when the truck goes for a curved path for connection. During this path, we also collected data that simulated the driver missing the drawbar eye connection. This again was done with the two same height as the previous cases.

In total 48 videos were recorded from this event. Every 10th frame from the videos was extracted to get images for the database. This gave a total of 2000 images. These images were then split into training, validation and testing data in the ratio of 8:1:1.

Once these images are extracted, each image needed to be annotated. We used Labelimg software for this process, which is a python based graphical image annotation tool. These images were labeled manually, with bounding boxes indicating the position of the drawbar eye in each image and tags the drawbar eye. This process is crucial to ensure that the model can accurately detect and classify the drawbar eye. Once the images are annotated, the data is extracted in the YOLO format for training the image recognition model.

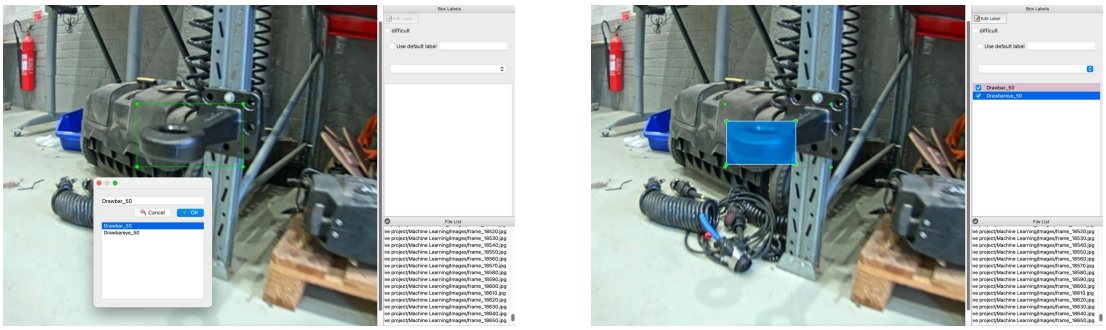

Figure 5.3: Labeling process of the drawbar and the drawbar eye.

The reason for only labeling the drawbar eye for locating of trailer in these images is due to the fact that it is a standardized part, and its surrounding attachments to the trailer can differ between different trucks and trailers. This should make it possible to switch between different trailers and still be able to connect the coupling.

5.1.2 Real world data collection

After the video data was collected in the lab environment, the next logical step was to collect data from real world scenario so that the model could detect and connect with trailers in real world conditions instead of the simulated conditions in an indoor setting. For this reason, data collection was done on two separate occasions and in two different locations.

Figure 5.4: Outdoor data collection.

The first data collection was activity was done at the Volvo Lundby area. In this activity, we used 3 different trailers, each with a different type of 57 mm drawbar

eye (differences being in the paint and supporting attachments of the same). In this instance, the data was collected in daylight conditions, albeit with slightly cloudy skies. The final data collection activity was done in Hisings Backa. In this activity we used 2 different trailers, each with a different type of 57 mm drawbar eye. In this instance, we collected data in night time conditions with pitch black darkness, and only the truck rear lighting to assist the trailer connection.

In order to collect data in these activities, the truck was reversed into the trailer from different angles and distances, along different paths, and sometimes with differences in connection height of the coupling and the drawbar eye in order to cover all possible scenarios that may be encountered. To label the images extracted from this data, Labelimg was again used with YOLO as the preferred format.

In these instances of outdoor collection, special care was taken since the images were usually not as clear, and many times the drawbar eye was not clearly visible, so the bounding boxes were only drawn when either the drawbar eye was fully or partially visible and differentiable from the surroundings. This was done to prevent occurrence of false positives in the image recognition model.

In total 36 videos were recorded from these 2 events. Every 50th frame from the videos was extracted to get images for the database giving a total of around 2000 images.

5.2 Model Implementation

The model implementation includes model selection, training, real-time position calculation, and evaluation. Additionally, we will integrate an audio-based Human Machine Interface (HMI) to guide drivers during the coupling process.

5.2.1 Model Selection

YOLO (You only Look Once) is a real-time object detection algorithm that predicts bounding boxes and class probabilities directly from images. As stated in the theory section, YOLOv8n was used in machine learning in this project. And we fine-tuned the model to be suitable for the detection of drawbar eyes in a variety of conditions.

5.2.2 Fisheye Camera Calibration

To use codes to calculate the intrinsic matrix and distortion coefficients, several steps are required. First, set the 3D points of the checkerboard, and get the 2D points by corner detection. Second, use cv2.fisheye.calibrate to calculate intrinsic parameter matrix, distortion coefficients, rotation matrix, and translation vector. Third, use the calculated intrinsic parameter matrix, distortion coefficients, rotation matrix, and translation vector, we utilized the cv2.fisheye.projectPoints function to compute the projection of 3D points onto a 2D image. Then the error is calculated between the point obtained by backprojection and the point detected on the image.

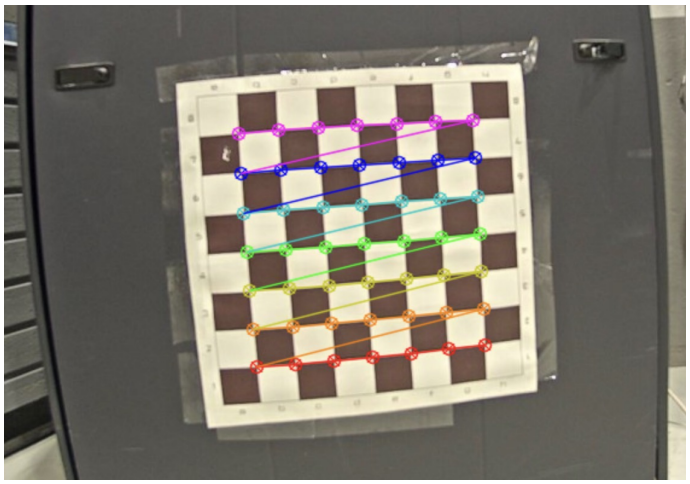

Figure 5.5: Fisheye camera calibration.

5.2.3 Data Processing

Images were extracted from videos, selecting one frame every 50 frames. In total, more than 4,000 images were extracted from 84 videos during three data collection sessions. For the final model, 1,519 images were used. These images were chosen based on real-life test scenarios and ensure the model's performance.

To transform the images and correct distortion, first, a tool in OpenCV called cv2.fisheye.initUndistortRectifyMap was used to calculate the adjustments needed for correction. Then, these adjustments were applied to the images using another tool, cv2.remap. This process ensures that the image is properly adjusted to closely match the real-world views.

The original image from the fisheye camera and the transformed image are shown in the following figures. Note the curved line in the original image, which is actually straight. The images were labeled in the LabelImg software in .txt form (label class and coordinates of the bounding box). When the labeling was done, images were split into training, validation and testing data in a general ratio of 8:1:1.

Figure 5.6: Original Image.

Figure 5.7: Transformed Image.

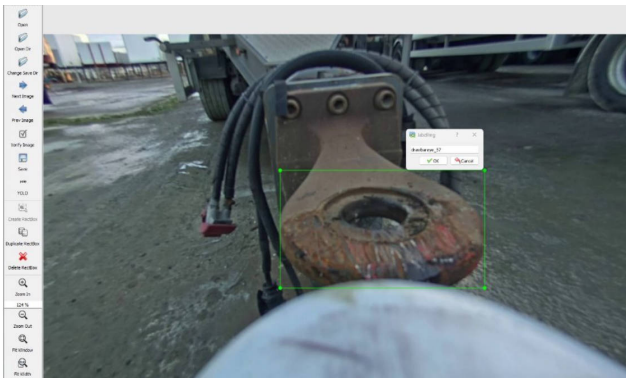

Figure 5.8: Image labelling 1.

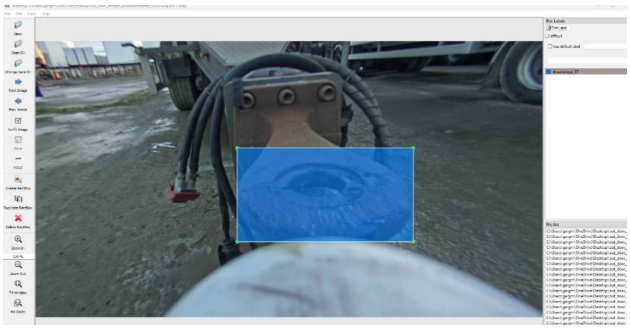

Figure 5.9: Image labelling 2.

5.2.4 Model Improvement

In this project, the varying object scales and the lack of distinct target features are significant challenges for detection. To address these issues and improve the performance of YOLOv8n, three modifications were implemented based on a review of relevant literature.

First, the ordinary convolution was replaced with partial convolution to make the model more lightweight and efficient. Second, SE attention mechanisms were added before each detection head to enhance the model's focus on important areas of the image, reducing background noise interference in complex environments. Third, a P2 layer was add to the Neck section for small object detection, enabling the feature map to capture richer semantic and spatial information. With the four detection heads, the model can achieve more effective multi-scale small object detection.

To evaluate the improvements,the model was trained and tested in four sessions. The first session used the original YOLOv8n model with no modifications. This served as a baseline for comparison. In the second session, the CSPPC module was introduced to replace the standard convolution. The third session added SE attention mechanisms to enhance detection accuracy. Finally, in the fourth session, a P2 layer was added to improve small object detection.

```
YOLOv8n summary: 225 layers, 3157200 parameters, 3157184 gradients, 8.9 GFLOPs
```

Figure 5.10: The layers of baseline model.

```
YOLOv8_csppc summary: 217 layers, 2577891 parameters, 2577875 gradients, 7.3 GFLOPs
```

Figure 5.11: The layers of improved mode.

After applying the lightweight modifications, the model was simplified by reducing 8 layers, resulting in a decrease of 579,309 parameters and 579,309 gradients. This reduction makes the model more efficient while maintaining performance.

The detection accuracy was evaluated using the metric mAP@50, which measures how well the model detects objects with at least 50% overlap between predicted and actual bounding boxes. The baseline model achieved an mAP@50 of 0.924.

After adding the SE attention mechanism, the mAP@50 increased slightly to 0.929, indicating a slight improvement in accuracy by reducing background noise interference.

After adding the P2 layer, the mAP@50 increased to 0.955, indicating that the P2 layer effectively enhanced the ability of the model to detect small objects, which improved the overall detection accuracy.

5.2.5 Training Process

For training the annotated dataset, the YOLO loss function combines localization loss, confidence loss, and classification loss. It is optimized using the stochastic gradient descent optimizer with a learning rate scheduler. We need to tune the batch size, learning rate and number of epochs to optimize model performance.

```
model = YOLO('yolov8n.pt')

# load the configration file
data_yaml = '/Users/liubiying/drawbareye_57.yaml'

# model training
model.train(
    data=data_yaml,
    epochs=50,
    batch=8,
    imgsz=512,
    name='drawbareye_57',
    device='cpu'
)

# model evaluation
metrics = model.val()
print(metrics)
```

Figure 5.12: YOLO training

The detection accuracy in YOLO is evaluated based on the overlap between the predicted labels, which represent the model's detection results (bounding boxes and classification), with the real labels, which are the data manually annotated in the images. By measuring the overlap between these labels, it is possible to assess how well the model's predictions match the real objects in the image.

5.2.6 Distance calculation

To guide the truck driver to adjust the drawbar coupling and connect it to the detected drawbar eye, it is necessary to calculate the distances in left-right, up-down and back-forth. First, the distance between the camera and the drawbar eye was calculated using the pinhole camera principle considering the camera mounting angle. The equations used for calculation are as shown in the figure below.

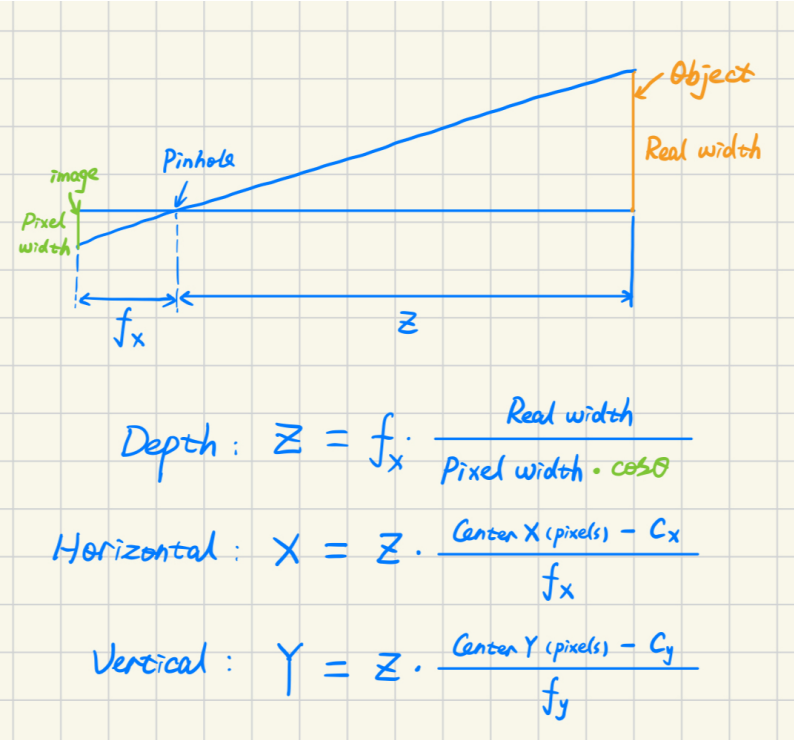

Figure 5.13: Distance calculation between camera and drawbar eye.

From the calculation, we had the horizontal distance (X), vertical distance (Y) and distance in depth (Z) between the camera and the drawbar eye. Then the distance between drawbar coupling and drawbar eye was calculated. Since the camera was mounted above the drawbar coupling, a height offset should be taken into account. To calculate the vertical distance (Y) between the drawbar coupling and the drawbar eye, the height offset between the camera and the drawbar coupling should be added to the vertical distance (Y) between the camera and the drawbar eye. The horizontal distance (X) and depth distance (Z) should be the same as the distances between the camera and the drawbar eye. This adjustment ensured that all distances were correctly referenced to the drawbar coupling.

5.2.7 Distance accuracy

From last section, the distances were calculated using the equations. In order to check if these distances were well matched to the actual distances, we output the calculated distances in the pictures taken in the distance accuracy test.

5.2.7.1 Comparison of the calculated distance with the actual distance

All the pictures taken in the distance accuracy test were run through the algorithm, and the algorithm would first transform the distorted picture to a normal one, then detect the drawbar eye,and calculate the horizontal distance(X), vertical distance(Y) and the distance in depth(Z). Finally, the calculated distances would be displayed on the input images.

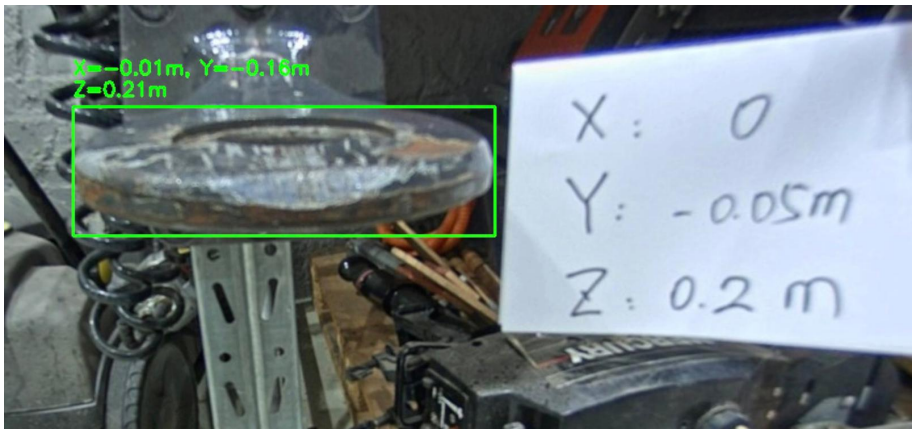

Figure 5.14: Picture showing both the actual and calculated distances.

By doing this, we can compare the calculated distance with the actual distance easily. It was found that the horizontal distance (X) and distance in depth (Z) were close to the actual distances. While for vertical distance (Y), the accuracy was not so good as the other two, especially when the camera was far away from the drawbar eye. It can be seen in Figure 5.15 way we get this error.

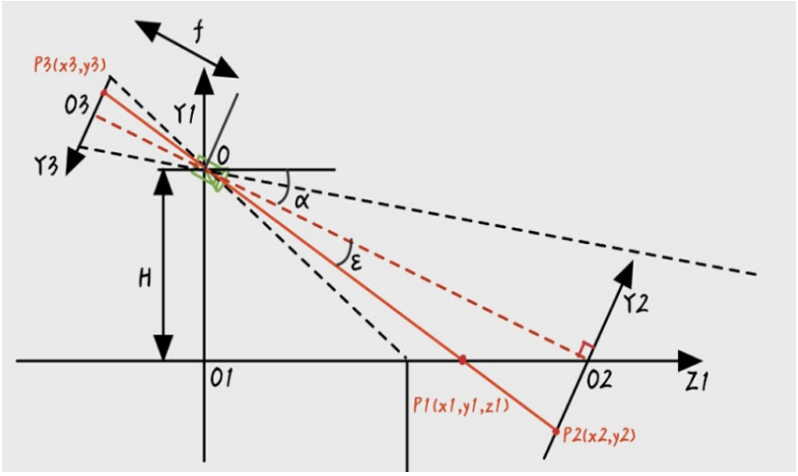

Figure 5.15: Vertical distance calculation model.

Here α is the angle of the camera, which was the camera mounting angle in this project, and ϵ is the relative angle between the object and the camera. When the camera is far away from the object, ϵ becomes larger, which will affect the calculation results. But when the camera is close to the object, ϵ will be relatively small, so the distance calculation in these positions had better accuracy.

5.3 Logic of guidance

This section describes the logic design of the guidance system, which is used to issue reminders to the driver during reversing. The system operates with a reminder interval of 3 seconds and provides specific instructions based on calculated distances to assist the driver

- If the depth distance Z > 0.3 m, the guidance suggests moving backward.
- If the horizontal distance |X| > 0.1 m, the guidance suggests moving left or right.
- When the depth distance |Z| < 0.3 m and the vertical distance |Y| > 0.1 m, the guidance suggests moving up or down.

This logic ensures that the driver receives clear and timely reminders during the reversing process.

5.4 Real-Time Position Calculation

When the YOLO model training is done, we can use the YOLO model to detect the position of the drawbar eye in the video stream in real-time. The trained model outputs the bounding boxes in the form of normalized coordinates relative to the input image. Using transformations to convert these to pixel values in the image. We need to create a position calculation algorithm to compute the center of the detected drawbar eye and the relative positional relationship between the drawbar eye and the drawbar coupling in distance and direction.

5.5 Model Evaluation

We evaluated the model using metrics such as accuracy, precision, recall, test the system under various real-world conditions including in different lighting and weathers, and validate the feasibility and stability of the model through multiple tests and optimizations, ensuring high accuracy under varying conditions.

5.6 Human Machine Interface (HMI)

The setup for the HMI can be seen below in figure 5.16. Here the microcomputer is a BeagleBone Black connected to a speaker through a USB-sound board. The microcomputer was able to give guiding instructions through a programming script out through the speaker. There was some issue with the HMI and the reason for that the HMI did not work was a memory issue regarding the microcomputer. The BeagleBone Black had a memory of 2 GB which was not enough. YOLO model required special python libraries which needed more memory that was available. This was realized late in the project when there was no time to change to a new microcomputer with enough memory.

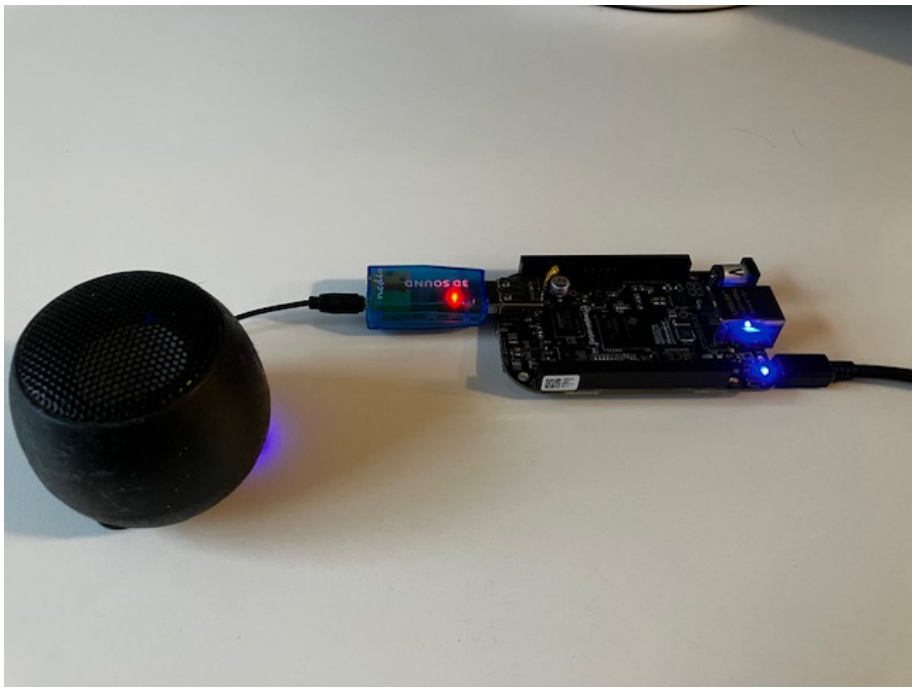

Figure 5.16: HMI Setup.

6
Results and Discussion

6.1 Project output (beginning goals vs tasks achieved)

In the beginning of the project, the following objectives were expected as project outputs:

- Presentation of a "lab solution" working for a number (2-3) of drawbar eye variants, preferably mounted on a vehicle, including a drawbar coupling - camera/computer for image processing and a Human Machine Interface (HMI).
- Presentation of possible technologies, transport layers, HMI etc
- Documented set-up/guide for how machine learning works to add additional drawbar eyes/environments to the database
- Documented analysis of technical requirements like equipment, choice of environment for further development into a commercial product.

In this project, we managed to present a working lab solution. The solution is working for a 57 mm drawbar eye. This variant is the most commonly used in Sweden, and due to experimental and time constraints, other drawbar eye variants were not covered. The solution is a program that can take any video and then use the train model to try to detect drawbars in the video. If a drawbar is found in the video then it calculates the distances from the camera. By using the calculated distances, the program then gives audio instructions for guiding the driver to connect to the drawbar.

This setup was mounted on a vehicle with a drawbar coupling and a camera for image processing. However since the HMI system was not integrated in this solution, so driver feedback for the subsequent system remains to be seen.

Through this report, we have documented the various technologies used in the project, explained the set-up for the working of machine learning and also provided a detailed analysis about the technical requirements and highlighted about the scope of expansion in this product.

So overall, a vast majority of the project objectives assigned were fulfilled. While as a short term project, this is sufficient as a working proof of concept, the solution will need to be worked upon before it can be made into a commercial product.

The camera used was a fisheye camera, although it comes with a distortion of the

pictures. For the limitation on were the camera could be placed this was necessary to get a full view during the guiding. With this comes calibrations to undistort the images to be able to get more accurate distance calculations.

6.2 Image recognition

For our trained model, when the overlap exceeded 50 %, the accuracy reached 98.7 %, showing the model works very well for general detection. For overlap within the range of 50 % to 95 %, the accuracy was 75.4 % meaning the model is less precise in some cases but still performs reasonably well. This shows that there is good potential in this model. To achieve a even better result, more images needs to be collected from different scenarios and added to the dataset.

6.3 Distance accuracy analysis

It was found that the horizontal distance (X) and distance in depth (Z) were close to the actual distances. While for vertical distance (Y), the accuracy was not so good as the other two, especially when the camera was far away from the drawbar eye. The reason for this is probably the pinhole camera model for vertical distance (Y) calculation. However, this error reduces when the distance gets closer. Thus, we will only use the vertical distance (Y) when the camera is close to the drawbar eye in further work.

6.4 Challenges

This project also had its fair share of challenges faced during its process, some expected while others not expected, some of these challenges were:

- The impact of image resolution on the model efficiency is not known, which may result in the model not performing to the given standard in case of change in camera setup.
- The placement of the camera was restricted and was also influenced by the size of the used camera.
- How to angle the camera to get a good result for distance calculations and image detection.

6.5 Team and personal learnings

Throughout the duration of this project, we were constantly learning about different new aspects of the project, ranging from technical work, to experimental setup, to even team work and management. It is through them that the team saw both individual and common growth.

Some of these learnings are as follows:

- **Technical enhancement:**
  Through this project, there was an influx of exposure into different technical domains and technologies like image recognition systems, machine learning algorithms, calibration and verification testing, camera types and HMI types to name a few. This helped in knowledge sharing between people from different technical backgrounds, leading to positive enhancement of every member.

- **Experiment formulation:**
  Planning different experiments by considering the aims and objectives of the project was crucial here. Also real-life factors and different biases related to experiment conduct were taken into account, which helped in grasping the essence of physical testing.

- **Real-life practicalities:**
  During the project, the team faced expected and unexpected challenges, ranging from setup constraints to experiment limitations, which led to constant changes and modifications in the project. Executing such changes while managing to fulfill the core objectives is an important learning of this project.

- **Team work:**
  The project was done with a group of people from different nationalities, thus understanding and adapting to different cultural differences and work styles is an important skill that all team members learned during this project.

6.6 Future Work

Next step would be to implement a Human Machine Interface (HMI) for the system. This to make the system applicable and also to evaluate the audio based guidance by a real driver to improve it further.

Another future work would be to add different types of drawbars to the dataset with different shapes, sizes and colors. Expanding the dataset to include more types of drawbars would make the model more versatile and applicable to a wider range of trailers. This is necessary if the goal is to launch a product like this on the market.

Although different environment and weather conditions have been explored in this project, further studies could be beneficial. For example, include extreme weather conditions such as heavy rain, snow, and fog. This could improve the robustness of the system and also evaluate if the guiding is any good under such conditions.

Another thing is to investigate different camera setups and evaluate whether moving the camera could improve the detection and distance calculations. VBG told us that we were restricted to the placement of the camera, but it could be worth to look into if there were a more optimal placement for it. Also, evaluating whether

a stereo camera could improve distance calculation, especially height measurement, compared to a fisheye monotype camera.

7
Conclusions

The purpose of this project was to investigate whether a machine learning algorithm could utilize a image feed from a camera to detect a drawbar eye and guide a truck for coupling a trailer. The results show that the YOLO model can successfully detect a drawbar eye with an accuracy of 98.7 % when the overlap exceeds 50 % and a accuracy of 75.4 % for a overlap within the range of 50 % to 95 %.

For the guiding and connection aspect, the distance calculations were found to be reasonable accurate in the x- and z-direction, but in the y-direction (height over ground) it showed less precision, particularly from greater distances. The y-direction however becomes more accurate the closer the camera comes to the drawbar eye. Height guidance are therefore done when the coupling and the drawbar eye are close to each other when the height guidance is more reliable.

The aim was to develop a proof of concept(POC) to demonstrate that a camera-based system can be used to assist truck drivers during the connection process. The result shows that it should be achievable if adding a larger and more diverse dataset. Testing in a broader range of environments is necessary to ensure robust detection under varying conditions. For example snow, dust and other environmental factors. This needs to be explored before implementing the solution on a real truck.

The Human Machine Interface (HMI) part was not fully implemented in the project. Therefore real-time guidance testing was limited. However, the guiding was valid through running videos on python scripts. This indicates that integrating an HMI for real-time use should not be a problem if using a suitable hardware.

In conclusion, the project demonstrates that it is possible to detect a drawbar using a single camera. The distance calculations shows a promising result but requires a bigger dataset and broader range of environments to get better accuracy for practical application. Additional exploring of more environments that can disturb the camera such as snow and dust need to be done before implementation. If these refinements, there is potential for implementation on a real truck.

Bibliography

[1] Drawbar (haulage). [Internet]. Wikipedia. Available from: https://en.wikipedia.org/wiki/Drawbar_(haulage)

[2] Fisheye Lens. [Internet]. Wikipedia. Available from: https://en.wikipedia.org/wiki/Fisheye_lens

[3] Fisheye lenses. [Internet]. Jessops. Available from: https://www.jessops.com/c/advice/bg/fisheye-lenses

[4] Lightweight small object detection algorithm based on improved YOLOv8n aerial photography: PECS-YOLO. WANG Shu-Meng, XU Hui-Ying, ZHU Xin-Zhong, HUANG Xiao, SONG Jie, Li Yi. DOI: 10.19678/j.issn.1000-3428.0069353

[5] Documents of YOLOv8. [Internet]. GitHub. Available from: https://docs.ultralytics.com/

[6] Code of YOLOv8n. [Internet]. GitHub. Available from: https://github.com/shuai-cao/YOLOv8_cs

[7] HumanSignal. GitHub - HumanSignal/labelImg. [Internet]. GitHub. Available from: https://github.com/HumanSignal/labelImg

[8] HMI: Human-Machine Interface. [Internet]. Inductive Automation. Available from: https://inductiveautomation.com/resources/article/what-is-hmi

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

CHALMERS
UNIVERSITY OF TECHNOLOGY