Danmarks Tekniske Universitet

DTU

Radar Tracking and Object Detection Special Course Report AgriRobot

11.12.2023

Author

s230003 Yu Fan Fong

Supervisors

DTU Associate Professor Søren Hansen
AgriRobot Robotics Engineer Christian Metz

Contents

1 Introduction

The content of this report covers the results and findings from the special course *Radar Tracking and Object Detection* in collaboration with AgriRobot, under the supervision of Associate Professor Søren Hansen and Christian Metz from AgriRobot.

1.1 Learning Objectives

The learning objectives are as follows:

- Use the static versus dynamic classification pipeline from AgriRobot and adjust it to the requirements for object tracking.
- Implement an algorithm for tracking radar targets.
- Investigate possibility of continuous tracking, even when the object is not any longer classified as dynamic. (e.g., Kalman Filter with positional covariance).
- Integration of solution into the current software stack of AgriRobot (ROS2 node written in C++).

1.2 Problem Description

The overall goal is to build an autonomous tractor platform to traverse safely in an agricultural field without colliding with static and dynamic obstacles. Dynamic obstacles may be the human operator supervising the platform operator or any other workers on the field, while static obstacles may be the crop or other idle farming equipment.

Hardware Setup

The tractors are equipped with two front-facing doppler radars, one front and one rear camera, a LIDAR scanner, and an alarm system with red, yellow and green LEDs.

Obstacle Detection Pipeline

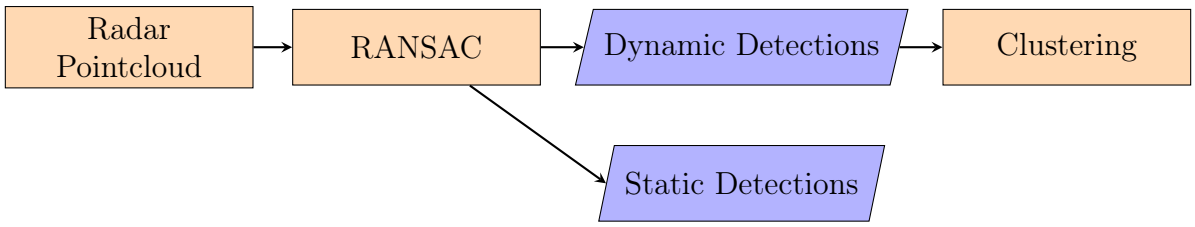

Figure 1: Classification Pipeline

AgriRobot has developed a pipeline for obstacle detection primarily using the Doppler radars. The Doppler radars produce a non-overlapping point cloud that covers the frontal 180°. The points in the point cloud are classified into static and dynamic points using a random sample consensus (RANSAC) algorithm. A group of dynamic points are then clustered together into one obstacle based on the euclidean distance.

Object tracking is needed to improve the functionality of the obstacle detection pipeline so that the obstacle positions are known as much as possible. However, the current pipeline fails in the following scenarios:

Radial Velocity Due to the nature of Doppler radars, if the dynamic obstacle's velocity is orthogonal to the radar, the dynamic obstacle will not be picked up by the radar.

Occlusion A radar detection requires a direct line-of-sight to the obstacle. If there is a static obstacle in between the radar and the dynamic obstacle, the dynamic obstacle will not be detected. This would be a common occurrence as the human may walk behinds trees on the field.

Drops in Detection The obstacle detection pipeline may fail to detect the obstacles, or have drops in detections in between successful detections.

Thus, object tracking should continue to estimate the position of the obstacles in the above scenarios so that the platform can continue to operate safely on the field.

2 Literature Review

2.1 Odometry

In the paper described by Kellner et al [1], it uses the measurements of the Doppler radars to estimate the egomotion of the ego-vehicle. This approach is advantageous as it can be done using measurements from a single cycle, and storage of history and data association is not required. The algorithm uses the radial velocities $v_{r,i}$ and azimuth positions θ_i from at least two stationary targets to construct the velocity profile of the vehicle. From here, the vehicle's ego-motion is estimated using a single=track model with Ackerman condition.

In the process, the algorithm is able to identify stationary targets as the majority of the targets with the same velocity profile. Outliers from are considered to be dynamic obstacles or noisy detection.

2.2 Object Tracking

Autonomous systems should be aware of their surroundings in order to interact with the environment and operate safely. Object detection and tracking is a widely studied problem

used in image analysis, autonomous driving, security, and many other robotic systems. The Simple Online and Realtime Tracking (SORT) [2] was developed as a minimalistic multiple object tracker to be used for tracking humans in a static video stream. It uses Kalman filters to estimate the object positions and the Hungarian algorithm for linear assignment between multiple detections and trackers. Kalman filter is a popular tool for tracking as it is a recursive and optimal linear estimator. This makes it suitable for online real-time processing where it does not require much data history and less time to compute. It assumes the noise from the system model and measurements are Gaussian. Whereas the Hungarian algorithm is responsible for the data association by computing the intersection-over-union (IOU) between each detection and trackers, and assigning them optimally. However, it does not work well with occlusion or objects that re-enter the frame because these instances are not common in their use case. In the followup model DeepSORT [3], appearance information is used to create a deep association metric to improve the performance of the data association to work better with long occlusions and reduce identify switches.

The state of the object in SORT's estimation model includes the object's pixel location, scale, aspect ratio, rate of change of pixel location and the rate of change of scale. The aspect ratio is assumed to be constant.

Another tracker developed by Alori et al. [4] offers greater flexibility as there are many preset distance functions that can be customised and it is possible to include a variable number of points per detection instead of using the bounding boxes and centroids only. [5] introduces a cumulative confidence metric that deteriorates with each missed detection of an object and accumululates with each detection to improve continuous tracking in the case of occlusions.

2.3 Object Trajectory

To improve the predictions of object positions in the future, it is possible to estimate the object trajectory. In A Social Spatio-Temporal Graph Convolutional Neural Network for Human Trajectory Prediction (Social-STGCNN) [6], it is applied in the context of autonomous driving on the road considering human agents at its core. Trajectories are modelled based on the learned interactions between the pedestrians and its surroundings.

3 Methodology

3.1 Selected Approach

In this project, the tracking was done largely based off SORT due to the available access to the code base and simplicity. It was adapted from tracking humans in a static video stream to tracking dynamic obstacles from a radar scan. In both scenarios, the problem and objects exist in a 2D plane with position measurements. SORT offers the advantage of multiple object tracking and assignment, however, it was not fully exploited in this project as only

one object was considered for development and testing.

3.2 Tracker Design

Kalman Filter

The tracking uses a discrete Kalman filter [7] assuming a constant velocity model with no input into the system. It assumes all measurement variables are independent. The aspect ratio of object, width and height of the object are assumed to be constant. The output of the classification pipeline is a 3D polygon, projected onto the 2D ground plane. Thus, the centroid, scale and aspect ratio of the object can be calculated from the 2D bounding box.

State Space Representation

$$x_{n+1} = F_n x_n + B_n u_n + Q_n$$
$$y_n = H_n x_n + R_n$$

$$State\ Vector, x_n = \begin{bmatrix} c_{x,n} & c_{y,n} & s_n & r_n & \dot{x}_n & \dot{y}_n \end{bmatrix}^T$$

$$Measurement\ Vector, y_n = \begin{bmatrix} c_{x,n} & c_{y,n} & s_n & r_n \end{bmatrix}^T$$

$$State\ Transition\ Matrix, F_n = \begin{bmatrix} 1 & 0 & 0 & 0 & dt & 0 \\ 0 & 1 & 0 & 0 & 0 & dt \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Observation\ Matrix, H_n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$Measurement\ Covariance, R_n = \begin{bmatrix} \sigma^2_{meas} & 0 & 0 & 0 \\ 0 & \sigma^2_{meas} & 0 & 0 \\ 0 & 0 & \sigma^2_{shape} & 0 \\ 0 & 0 & 0 & \sigma^2_{shape} \end{bmatrix}$$

$$Process\ Noise\ Covariance, Q_n = \begin{bmatrix} \sigma_{pos}^2 & 0 & 0 & 0 & \sigma_{pos}\sigma_{vel} & 0 \\ 0 & \sigma_{pos}^2 & 0 & 0 & 0 & \sigma_{pos}\sigma_{vel} \\ 0 & 0 & \sigma_{shape}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{shape}^2 & 0 & 0 \\ \sigma_{pos}\sigma_{vel} & 0 & 0 & 0 & \sigma_{vel}^2 & 0 \\ 0 & \sigma_{pos}\sigma_{vel} & 0 & 0 & 0 & \sigma_{vel}^2 \end{bmatrix}$$

$$Initial\ Covariance\ Matrix, P_0 = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix}$$

The initial covariance matrix P gives high uncertainty to the unobservable initial velocities. When there are no detections, the state extrapolated using the prediction step of the Kalman filter without measurements.

Kalman Filter Parameters

$$\begin{aligned} dt &= 0.2 \\ \sigma_{meas} &= 1.0 \\ \sigma_{shape} &= 0.1 \\ \sigma_{pos} = \sigma_{vel} &= 0.05 \end{aligned}$$

dt was derived from the period of the SORT tracker which matches the output frequency of the classification pipeline. It was noted that the detection deviations from AgriRobot's tests were up to 0.5m, but after trial and error, it was that $\sigma_{meas} = 1.0$ worked well with the rosbags provided. σ_{shape} was given a smaller value of 0.1 as the height and width of the obstacles are not expected to change over time, so the shape measurements are expected to be less noisy.

SORT Parameters

$$\begin{aligned} frequency &= 5\ Hz \\ min\_hits &= 3 \\ max\_age &= 10\ seconds \\ max\_speed &= 2.0/s \end{aligned}$$

The *frequency* of 5 Hz was selected to match the output frequency of the classification pipeline. *hits* is the number of times a tracker has a match with a detection, while *age* is the duration since a tracker last matched with a detection. The maximum speed is the

maximum allowable speed of trackers. At each iteration, only valid trackers are returned. Valid trackers satisfy the following conditions:

$$\begin{aligned} hit\_streak &> min\_hits \\ time\_since\_update &< max\_age \\ |\dot{x}|, |\dot{y}| &< max\_speed \end{aligned}$$

Imposing the minimum hit condition prevents noisy trackers from being initialised as detections that only appear for a few frames are not returned as valid trackers. Trackers that do not have a matched detection beyond the *max_ age* duration are considered lost and deleted from memory. A flowchart describing the workflow of the tracker can be seen in figure 2 below.

A constant acceleration model was tested in an attempt to capture the direction change of the object, but it introduced more noise into the tracker. $\dot{s}$ (scale derivative) was removed as this state is relevant for video feed from a static camera, but not for a radar scan. The minimum IOU values was reduced from 0.3 to 0.01 as the tests did not include multiple objects and sparse objects are expected.

3.3 Implementation

A virtual software development machine was provided by AgriRobot to run their code base. The code is written in Python to be used in ROS2. ROS Visualisation (RViz) was used to have a realtime visualisation of the detections and trackers (see figure 3), while PlotJuggler [8], an open-sourced time series visualisation tool, was used to generate plots.

AgriRobot has recorded a few rosbags that store ROS message data from their outfield tests. This allows for offline testing and development without physically operating the hardware at the real location. The rosbags contain data on the cameras, Doppler radars, odometry, alarm status, and the output of the classification pipeline.

The detections from the classifcation pipeline are given a blue 3D polygon, projected into purple 2D bounding boxes on the ground plane. The output of the tracker are given orange 2D bounding boxes. Solid orange boxes correspond to the detected trackers, while translucent orange boxes correspond to the undetected trackers. It should be noted that trackers are removed after being undetected for the *max_ age* duration. Due to the limitations of RViz, these dead trackers were not immediately removeable from RViz. Thus, orange boxes that are not moving on RViz are probably dead trackers.

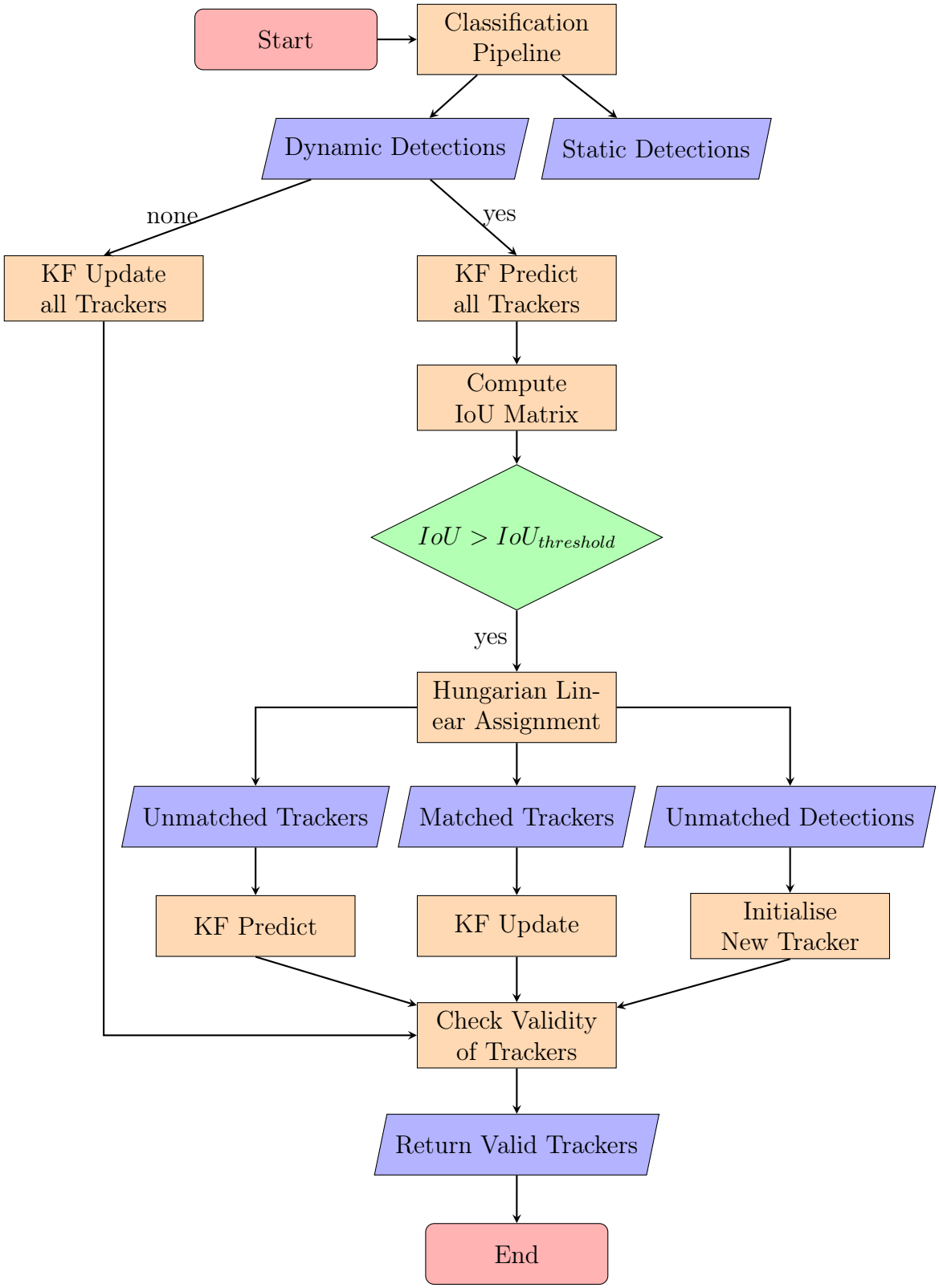

Figure 2: Workflow of tracker

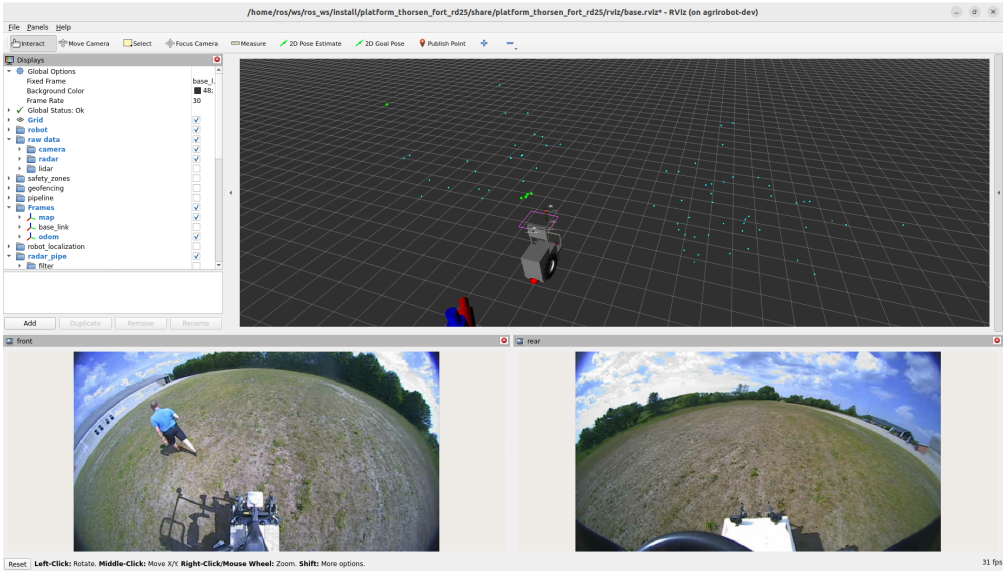

Figure 3: RViz Configuration from AgriRobot

4 Results

Overall, there were no issues with the computational speed as it was able to match the output frequency of the classification pipeline of about 5 Hz.

4.1 Simulation

The tracker was developed and tested using simulated dummy data. The dummy data introduces Gaussian noise to the center positions and bounding box shape parameters. As expected, the tracker's performance drops with larger standard deviations as the trackers have bad velocity initializations due to the noise. It was noted that the detection deviations from AgriRobot's tests were up to 0.5m. Thus, the standard deviation used to generate the dummy data was tested from 0.2m to 0.5m, and it was finally settled on 0.3m for further testing.

In figure 4, the detections are always published with and the object travels in a straight line.

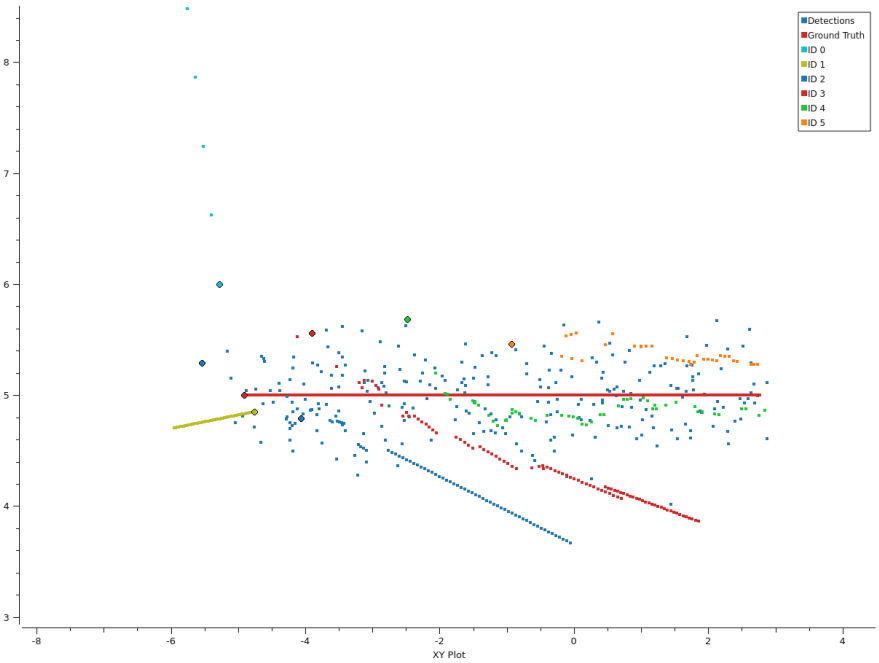

Figure 4: XY plot of trackers with an object travelling in a straight line.

Tracker 4 had the best performance and its x-position is plotted in figure 5. However, this tracker was only created after 8 seconds as bad trackers were created before that.

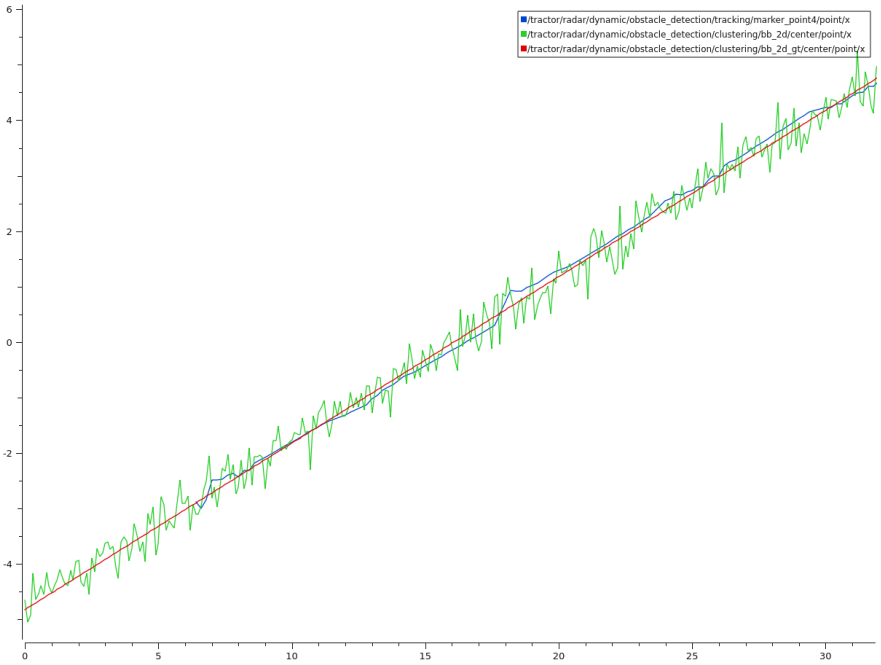

Figure 5: X plot of tracker ID 4 (blue)

To simulate occlusion, the detections are published cyclically for 3 seconds and stopped for 3 seconds in figure 6.

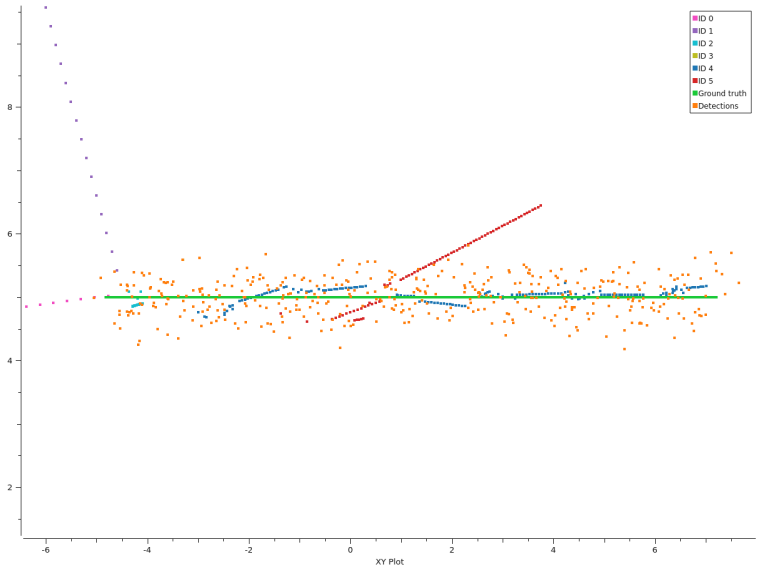

Figure 6: XY plot of trackers along straight path with occlusion

In figure 7 below, the object travels in a circular path with radius of 5.0m. The detections are published cyclically for 3 seconds, and stopped for 3 seconds. Due to the constant velocity model, the trackers continue along the tangent of the circle once the detections stop.

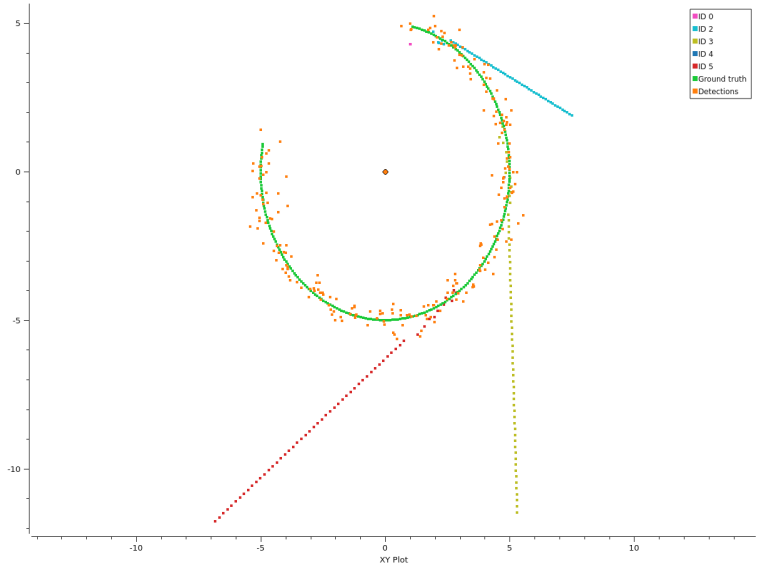

Figure 7: XY plot of trackers in a circle path

4.2 Recorded Data

In the rosbags available, there was always only one dynamic human obstacle in an open field. There were no sensors mounted on the human to collect ground truth positions of the dynamic obstacle. Thus, it could only be inferred from the images of the front and back camera. Screen recordings can be found in this OneDrive folder.

General Observations

As the tests were carried out in an open field, instances of occlusion were not present in the data available so the performance of the tracker on real occlusion cases was not tested. With the constant velocity model in the Kalman filter, the tracker is unable to track the position of the dynamic obstacle accurately when the human changes direction or stops when there are no detections. However, when the human walks in a straight line alongside the tractor, it was able to track the human in some instances. When the human walks along side the tractor and stops, the tracker is able to continue to track when the detections drop off.

Samples from rosbags

Two scenarios were chosen from the rosbags available to illustrate the capabilities of the tracker in figure 8 and 9. The detections and trackers start from the dot. In figure 8, the object travelled in a flipped-C shape tracker. However, at the bottom of the C, the measurements were noisy and the detections did not match with the tracker, so the tracker went off in a straight line using predictions only.

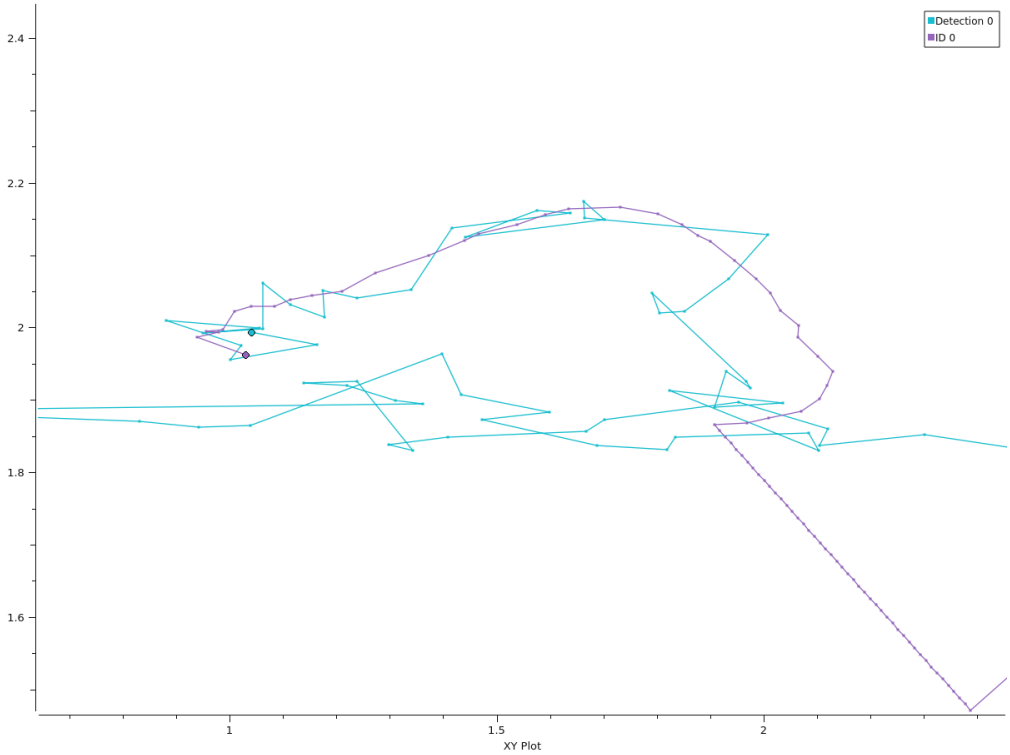

Figure 8: XY plot of tracker ID 0 (purple) from rosbag

Tracker ID 2 was created about 8 seconds into the recording, when the object started travelling in a straight line. The tracker was able to capture this motion adequately.

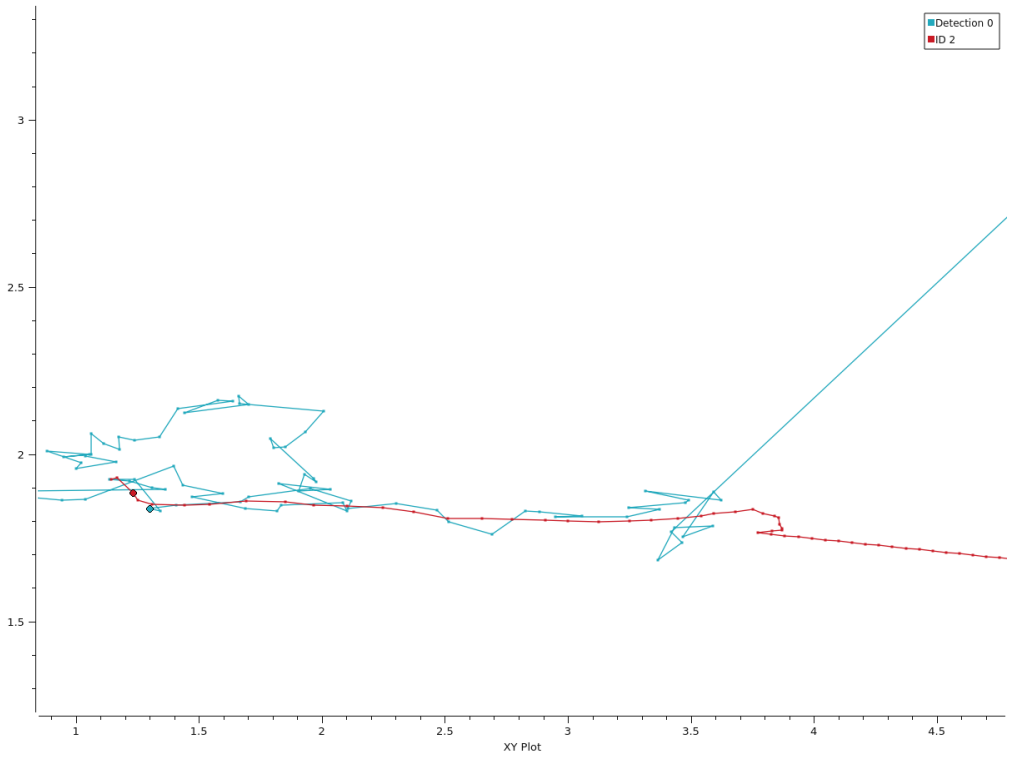

Figure 9: XY plot of tracker ID 2 (red) from rosbag

5 Conclusion

A tracker was implemented using a modified version of the Simple, Online, and Realtime Tracking (SORT) developed by Bewley et al. It was tested in simulations and on pre-recorded data. The tracker performed well with noisy detections, but it does not predict the object's position well if there are no detections for more than 5 seconds due to the limitations of the constant velocity model.

The performance of the tracker is still largely reliant on the performance of the detections from the classifcation pipeline.

Future Works

The initial velocity estimates of the model has a large impact on the quality of the trackers generated. Due to the properties of Doppler radars, it is possible to estimate the lateral velocities of the objects [9]. The estimates can be passed as into the Kalman filter to improve the overall tracker performance.

Due to the nature of the platform's operating environment, the objects will probably be sparsely separated on the field. Thus, it is possible to improve the assignment problem of matching detections to trackers by matching based on the nearest detection instead of IOU.

References

[1] Dominik Kellner, Michael Barjenbruch, Jens Klappstein, Jürgen Dickmann, and Klaus Dietmayer. *Instantaneous ego-motion estimation using Doppler radar*. 2013.

[2] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, September 2016.

[3] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[4] Joaquín Alori, Alan Descoins, javier, Facundo Lezama, KotaYuhara, Diego Fernández, Agustín Castro, fatih, David, Rocío Cruz Linares, Francisco Kurucz, Braulio Ríos, shafu.eth, Kadir Nar, David Huh, and Moises. tryolabs/norfair: v2.2.0, January 2023.

[5] Axel Jernbäcker. Kalman filters as an enhancement to object tracking using yolov7. Master's thesis, KTH, Mathematics (Div.), 2022.

[6] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction, 2020.

[7] Alex Becker. *Kalman Filter from the Ground Up*. 2023. Accessed: 28-11-2023.

[8] Plotjuggler: Time series visualisation tool. `https://github.com/facontidavide/PlotJuggler`. Accessed: 28-11-2023.

[9] Dominik Kellner, Michael Barjenbruch, Klaus Dietmayer, Jens Klappstein, and Jürgen Dickmann. Instantaneous lateral velocity estimation of a vehicle using doppler radar. In *Proceedings of the 16th International Conference on Information Fusion*, pages 877–884, 2013.