Autonomous Surface Vessel detection and tracking of static and dynamic obstacles

Submitted by
Hashir Zahir
Department of Electrical and Computer Engineering

In partial fulfilment of the
requirements for the Degree of
Bachelor of Engineering
National University of Singapore

ABSTRACT

Obstacle detection and tracking is crucial for Autonomous Surface Vessels (ASVs) to navigate safely in a sea environment with other moving vessels and static obstacles. The Bumblebee ASV has been the culmination of 5 years of continuous effort by Team Bumblebee at the National University of Singapore. Over the years, many key capabilities have been added but obstacle avoidance was one of its key deficiencies of the software stack despite a stable platform with strong controls system. One of the key reasons behind this was the poor obstacle detection and tracking performance. The previously used grid-based approach also caused smeared grids when moving obstacles passed by the ASV, making the grid unusable. Moreover, over the years, the radar was largely left unused for the perception system due to difficulty in integration, lowering the ASV's capabilities to detect obstacles. In this project, an algorithm was developed to help the ASV detect and track static and dynamic obstacles. In particular, the focus was on near shore and near port scenarios with higher density of vessel movement as this was the primary deployment of the Bumblebee ASV. To accomplish this, an effective framework to represent static and dynamic obstacles is required. For this project, a combination of a probabilistic grid map for static obstacles and a centroid based tracker approach with state estimation was taken for dynamic obstacles. Camera images, lidar point clouds and radar data were fused to obtain the detections and tracks. The algorithm was evaluated on the Bumblebee ASV at the Republic of Singapore Yacht Club (RSYC) via 2 separate scenarios, where it showed good results in segmenting the static and dynamic obstacles, with a small number of false positives.

ACKNOWLEDGEMENTS

I would like to thank Prof Mandar Chitre for his continuous support during this year-long effort. I would also like to acknowledge the efforts of the Bumblebee Autonomous Systems (BBAS) Team in assisting with the Surface Vessel hardware setup and deployment, which was crucial for this project. In particular, I would like to thank Ng Yong Jie, Ng Zhia Yang and Niu Xinyuan. I would also like to thank Goh Eng Wei for sharing his expertise and experience in marine autonomy systems as well as shedding light on some of the industry. I would also like to thank the Republic of Singapore Yacht Club (RSYC) for allowing us to use the marina for ASV testing. Finally, I would like to thank NUS Engineering, NUS IDP, NUS Computing and all the other industrial sponsors of Bumblebee for their continued support of the project and making all this possible.

Table of contents

List of figures

1. Introduction

Unmanned or Autonomous Surface Vessel interest and research have been increasing over the years [1] with usage in marine research and transportation. These include usage in operations such as coastal surveys, patrols and even deployment of Autonomous Underwater Vehicles (AUVs) from the ASV platform for underwater inspections in regions further away from land. Given these applications, the typical ASV will have to

1. Navigate out of a docked location near a port with the surrounding bustling port traffic
2. Navigate in open sea where larger ships pass and perform mission
3. Return to the dock after the end of operation

From this, it is observed that collision avoidance is the most crucial and basic requirement for an ASV to safely operate in tandem with other manned boats and vessels. The development of a reliable and robust perception system to aid navigation of these autonomous agents has been of increased interest to researchers over the past few years.

Counterparts from the Autonomous Vehicle (AV) side have similarly been making huge strides in development, with Tesla a prime example. However, true autonomous vehicles based on the Technology Readiness Level or the Autonomy Level scale still have a long way to go with many experts predicting a 5-10 year development cycle before autonomous deployment.

Various perception strategies have been previously implemented to detect 3D static and dynamic obstacles, especially in the autonomous vehicle scene, to enhance the autonomous capabilities of these systems. Many of these strategies have been ported over to autonomous marine vessels and typically involve individual detection of obstacles at the

sensor level. New research work is appearing in multi sensor fusion approaches to increase overall accuracy, reliability, and coverage. However, reliability in detection and tracking of objects have suffered with false positives.

At the National University of Singapore, an undergraduate student team called Bumblebee Autonomous Systems (BBAS) is involved with maritime autonomous systems research and testing. Every year, BBAS prepares for international competitions such as the International Robosub and the International Maritime RobotX, which utilize Autonomous Underwater Vehicles (AUVs) and Autonomous Surface Vessels (ASVs) to perform tasks such as docking, obstacle avoidance and dynamic positioning in open waters. The Bumblebee ASV is currently located at the Republic of Singapore Yacht Club (RSYC) for testing and development of autonomy solutions.

Figure 1: Bumblebee ASV navigating around the RSYC marina

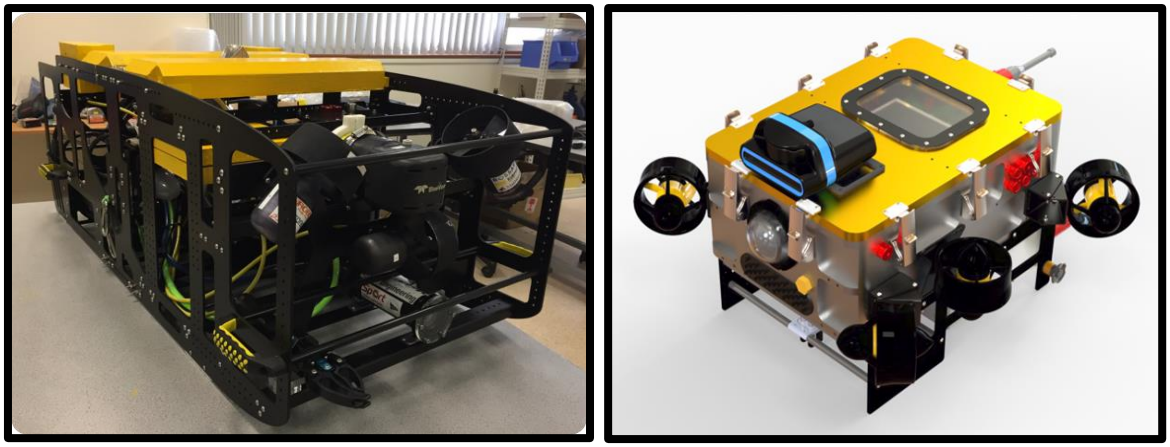

Figure 2: Bumblebee AUV3.99 and AUV4.0

The Bumblebee ASV has a stable electrical system with a wide array of powerful sensors such as RGB cameras, 3D lidars and a marine radar. The Bumblebee ASV has a control stack that allows for robust stationkeeping in rough weather as well as waypointing to coordinates. However, the Bumblebee ASV has relatively weak obstacle avoidance capabilities that has hindered its goal in usage for fully autonomous missions. This is mainly due to its weak obstacle detection system and the previously used grid map approach which smeared in the presence of dynamic obstacles, making it unusable for navigation. This is further complicated by the fact that the ASV needs to navigate in a variety of complex scenarios, such as docking, high density port side traffic and sparse open sea traffic. Additionally, previous integration difficulties with the old marine radar reduced its perception capabilities. This necessitates the development of a perception system that can detect obstacles reliably.

The aim of this project is to establish a strong perception suite for the Bumblebee ASV to detect and track static and dynamic obstacles. This will allow for the development of obstacle avoidance capabilities once obstacles can be clearly identified by the ASV.

2. Project objectives and contributions

The Bumblebee ASV is equipped with 3 RGB cameras, 2 3D lidars and 1 marine radar. The aim of this research project will be to improve detection and tracking of marine targets via multi sensor fusion approach and to test out the developed work in real life via the Bumblebee ASV to detect static and dynamic marine targets such as buoys and small marine vessels.

These are the requirements needed in order to hit the project objective:

- Setup of system and collection of high-quality datasets via the Bumblebee ASV
- The algorithm must work minimally with a radar and a camera, as these are the 2 most common sensors available on ASVs
- Identification of static obstacles such as landmass and parked vessels in crowded ports
- Detection and tracking of moving vessels / boats with state estimation of position and velocity

These are my contributions and achievements for the project:

1. Development of a software driver for the new marine radar of Bumblebee ASV
2. Collection of a new high-quality dataset via Bumblebee ASV for testing of perception system
3. Development and testing of new perception system to detect and track static and dynamic obstacles for the Bumblebee ASV

3. Literature review

In this section, various techniques that are commonly used in perception systems of autonomous agents to detect and track obstacles are described with descriptions on the advantages and disadvantages.

3.1. Overview of sensor fusion techniques for obstacle detection and tracking

As with most autonomous robotic agents, a multi-sensor fusion approach is taken for tracking obstacles in the environment. This is done so as to prevent over-reliance on any one sensor as well as to have the various different sensors overcome each other's pitfalls and combine their advantages into a robust and accurate detection and track. Fayyad [2] provides a detailed explanation that spans most of the commonly used perception strategies for autonomous systems.

3.1.1. Fusion architectures

Firstly, there are mainly 3 types of fusion strategies when dealing with multiple sensor data:

- Decision level fusion
- Feature level fusion
- Raw data level fusion

For decision level sensor fusion, each sensor modality computes its own individual information about the obstacle and the individual decisions from the obstacles are then fused. This is the most typical sensor fusion architecture used in research as it allows for modularity, ease of debugging and allows for more generic fusion strategies to be applied. However, this

usually comes at the cost of ignoring sensor specific traits at the decision level fusion, which might be able to provide additional details useful for fusion.

At the other end of the spectrum, raw data fusion strategies involve end to end data processing whereby raw data is fed into the sensor fusion model and a decision outcome of the obstacle is output. This allows for high levels of custom behaviour to be specified for each sensor, potentially increasing accuracy with increased complexity as the fusion strategy is directly linked with the sensor modality, losing out the modularity and ease of debugging features. Moreover, it will be unable to handle situations in which a sensor modality has a failure since all the sensor data is needed in this model to predict a decision.

As a middle ground of the above 2 strategies, feature level fusion involves extracting meaningful features from each sensor modality and fusing them to output a decision.

3.1.2. Sensor fusion algorithms

Classical sensor fusion algorithms for data association and tracking across sensor modalities include probabilistic and statistical methods such as K-nearest neighbours, Kalman filters and probabilistic data association filters. These are common strategies employed to tackle common perception-based sensor fusion setups such as camera-lidar fusion and lidar-radar fusion. More recent work explores deep learning-based approaches to fuse the data, classify and track the resultant obstacle. However, deep learning approaches to sensor fusion are in their early stages and are not used in research and industry as extensively as probabilistic and statistical based fusion methods.

3.2. Overview of marine object detection and sensor fusion techniques

Most of the content discussed in the literature review thus far has been applicable to the majority of the perception multi-sensor fusion systems present in many autonomous agents (Autonomous Vehicles, Aerial Drones, etc). However, this paper focuses on the application of such strategies to the marine environment to aid the detection of vessels and objects. A crucial sensor utilized by almost all marine vessels (manned or unmanned) is the marine radar. These marine radars typically provide a 360-degree horizontal field of view and have maximum sensor detection ranges from 1km to 20km depending on the radio frequency band of the marine radar. This is crucial for the long-range navigation capabilities in the open sea where rain, fog and lack of landmark features can make detection and tracking of obstacles challenging in open seas.

3.2.1. Lidar-Radar marine sensor fusion approach

Han [3] utilizes a lidar and radar for the detection and tracking of marine targets in the open sea. The main idea behind the paper was to take advantage of the lidar's high resolution near range (0-100m range) detection capabilities and to combine it with the radar's lower resolution long range detection capabilities as some marine radars have significant circular dead-zones up to 100m. Han [3] implements an Extended Kalman Filter (EKF), with constant velocity model, to track marine targets. A decision level fusion model was implemented with dynamic thresholding to detect marine targets in the radar image, and Point-Distance Based Segmentation (PDBS) algorithm applied to 3D lidar pointclouds to detect marine targets. In their setup, the lidar and the radar sensor readings had almost non-existent overlapping sensor range and hence they opted to implement an automatic tracker that switches between the lidar and radar detections depending on the distance of the marine target from the lidar and radar.

While this method would work in the open sea, high density areas near ports and shore make this approach difficult to utilize as the lidar alone is insufficient for static and dynamic tracking. On the Bumblebee ASV, the lidar and radar have a very high overlapping range with the lidar detecting from 1m to 120m and the radar detecting from 1m to 1500m. As such, a more powerful sensor fusion approach would suit the Bumblebee ASV for near shore perception.

3.2.2. Camera-IR-Lidar-Radar sensor fusion approach

Hagbhayan [4] approaches the marine object detection and tracking problem with a high multi sensor approach in which 4 unique sensor modalities (RGB camera, IR thermal camera, lidar and radar) are utilized for a robust and accurate detection and tracking system. This is yet another paper utilizing the decision level fusion approach. As compared to the previous paper, 2D RGB camera and IR thermal camera images are harder to fuse into 3D perception systems as 1 degree of freedom has been lost in the data. 2D bounding box detections from the RGB and IR thermal camera images are detected separately via dynamic thresholding approaches. The 2D RGB camera and IR thermal camera image detections are then projected onto a flat 2D top-down bird-eye view metric map (similar to a radar based map), with some crude assumptions of the environment (such as surface of water) applied via inverse perspective mapping to obtain metric coordinates of the marine target not usually possible from monocular camera singular images.

Simple filtering and thresholding methods are applied to the 3D lidar and radar data and are then projected onto the same 2D bird eye view map. However, by combining the advantages and disadvantages of the various sensor modalities (such as IR thermal camera working well to detect land mass very distinctly, lidar giving high resolution near field detection, radar with weather tolerant long distance detections and RGB cameras providing rich features

that can identify the type of marine target) and fusing the 4 individual sensor detections via a probabilistic data association, providing fused object detection region proposals without labels. A convolutional neural network (CNN) was then applied to obtain the labels of these regions. The final obtained accuracy and robustness of this approach was much higher than what each sensor modality could provide.

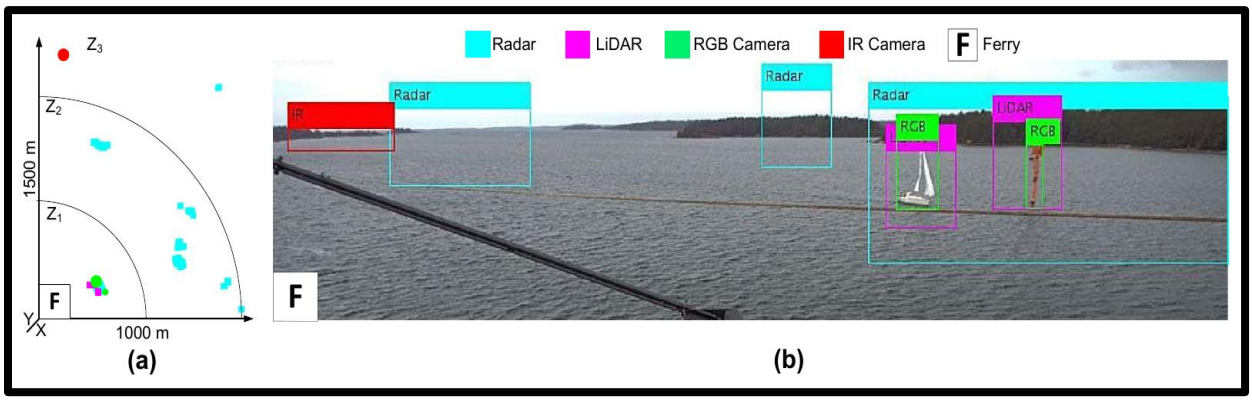

Figure 3: Projection of individual sensor detections on to 2D metric grid

However, this approach was only tested on post processed data collected from a ferry instead of an online real-time approach and it is unclear from the paper if the approach can run online as that was a future step. Moreover, the paper focused mostly on open sea detection in which targets are sparsely populated. However, this approach would suffer in higher density port and shore side navigation where there are higher chances of occlusion leading to a higher false positive rate for data association and tracking.

3.2.3. Deep learning approaches to radar image detection

Mou [5] states that traditional methods such as clustering, thresholding and filtering are usually utilized in radar image detection strategies but have limited results due to noisy radar images present from underlying radar technology, as well as huge clusters forming between obstacles resulting in large clutters which reduce detection accuracy and reliability.

Mou [5] implements a custom deep learning network based on the faster R-CNN architecture to better detect targets from the marine radar image.

The radar image was first pre-processed and cleaned to reduce the effects of noise disturbances. Next, the radar image itself is cropped and segmented into different sectors that are then fed into their custom RCNN model to classify if the region was a landmass or marine target. They managed to obtain high accuracy in classification but needed more work to speed up the algorithm as well as to improve upon the binary detection network.

3.3 Gaps in literature

As seen in the previous sections, most of the research for ASV perception systems are designed primarily for long range navigation in sparse open sea waters. This was evident in the testing methodology for most of these papers as well. However, as mentioned in section 1, the ASV is typically parked in a port environment. In order to support fully autonomous navigation, perception systems that work in these crowded port or near shore environments are crucial to the application of ASVs. As such, the remainder of this paper will investigate the problem of near shore navigation with self-collected data.

4. Design of algorithm

In this section, the main idea behind the proposed algorithm will be described. A sample scenario will also be described to provide context and motivation for the algorithm and some key definitions that are necessary to follow the rest of the paper will also be defined.

4.1. Necessary definitions

Before discussing any technical work, some key definitions need to be laid out to simplify and standardize terms across this report. Firstly, Robot Operating System (ROS) is an open-source robotic middleware framework commonly utilized on many robotic platforms. Rosbag is a particular data recording and storage tool used to record live data to be played back for analysis at a later time. The VRX-Gazebo Simulator [6] is an open-source simulator utilized for robotic simulations of ASVs. It will be utilized to perform high level testing of the algorithms. The ROS middleware framework and Rosbags will also be utilized heavily for this project.

Next, the perception sensor suite of the Bumblebee ASV is as shown in the figure below. Moreover, the Bumblebee ASV can move at a top speed of 5 knots and has vectored thrust to allow for fully holonomic motions. This makes it more agile and responsive and is better suited for near shore navigation operations with high stationkeeping requirements.

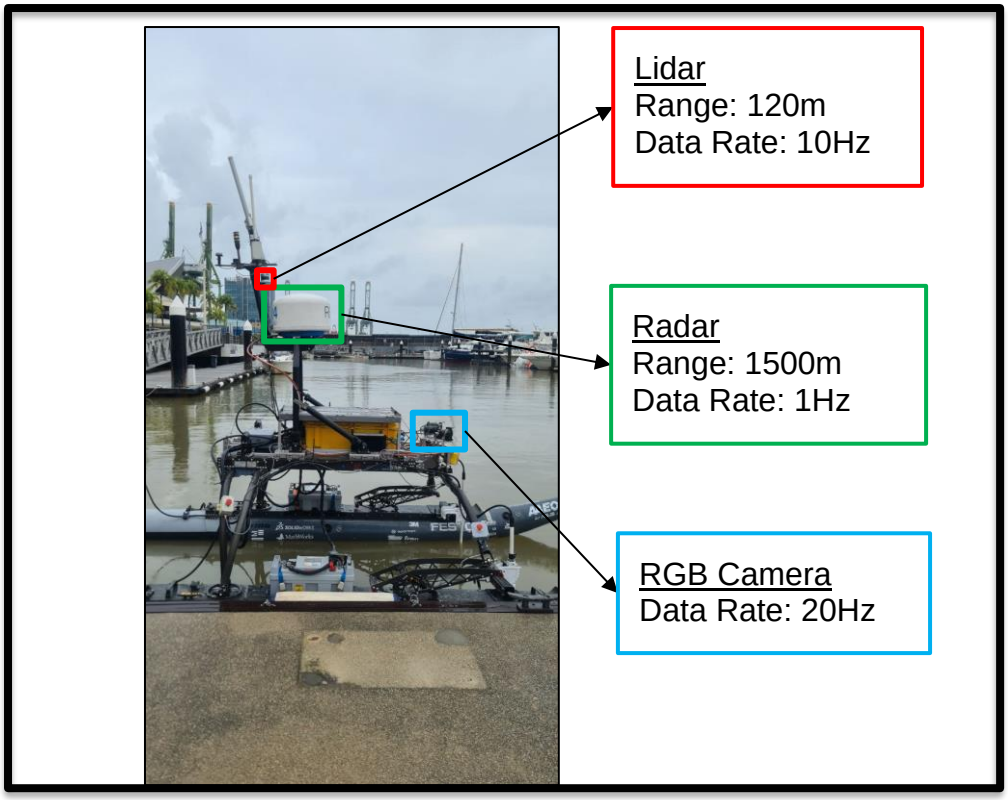

Figure 4: Bumblebee ASV sensor suite for perception

As in any robotics-based paper, a definition of the coordinate system used must be clearly defined. The North East Down (NED) coordinate system will be utilized for both body and world frame for the rest of the paper.

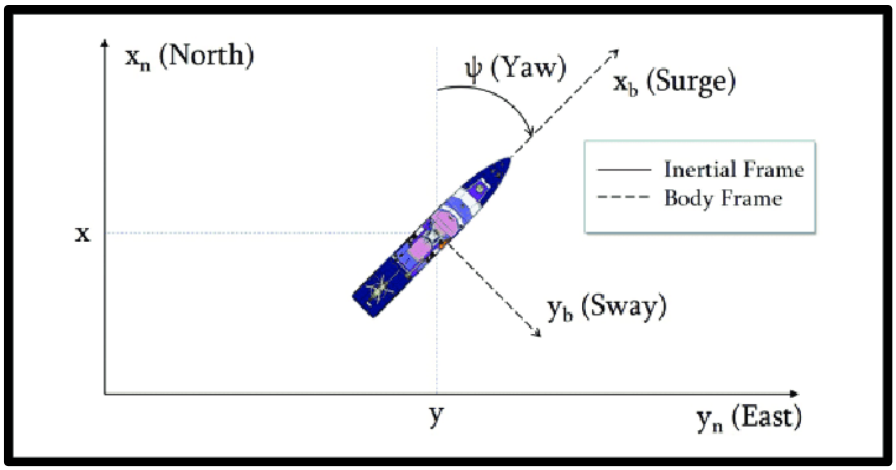

Figure 5: NED coordinate system used on marine vessels, Ejaz [7]

4.2. Example scenario

The aim of this project is the development of a perception system that is capable of detecting and tracking obstacles throughout the full journey of the ASV. Most of the current literature heavily focuses on the tracking of vessels in open sea but neglect the near shore and port scenarios where the environment is vastly different.

As mentioned in section 1, the typical ASV deployment route involves:

1. Navigate out of a docked location near a port with the surrounding bustling port traffic
2. Navigate in open sea where larger ships pass and perform mission
3. Return to the dock after the end of operation

The Bumblebee ASV will follow this particular route for this example scenario with a greater focus on part 1 of the route. In part 1 of the route, the ASV will initially be parked alongside many other stationary vessels in a crowded location and will try to navigate through the crowded port safely.

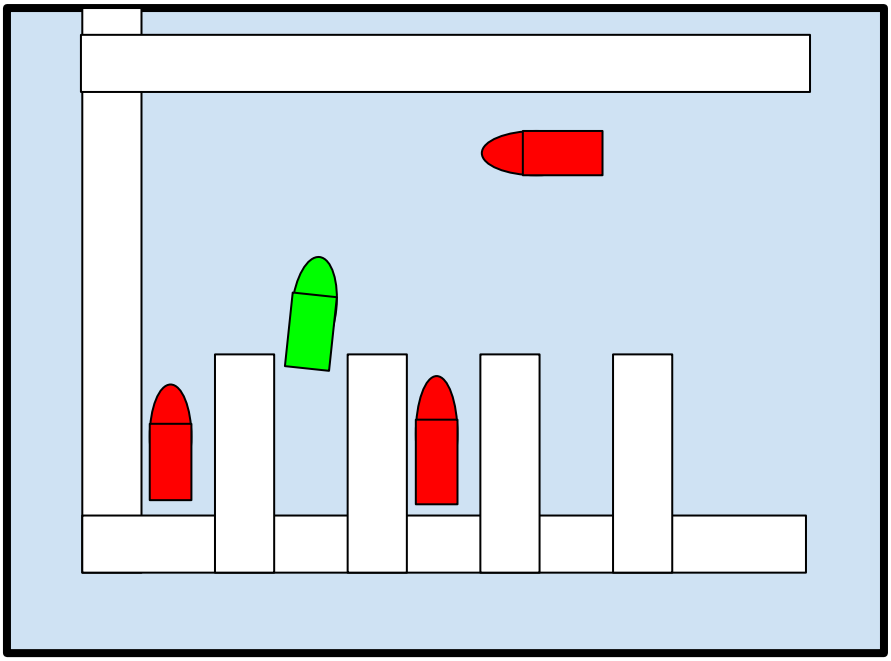

Figure 6: Example Scenario 1: Our vessel of interest (in green) trying to navigate out of parked dock with vessels (in red) and stationary port (in white) and sea water (in blue)

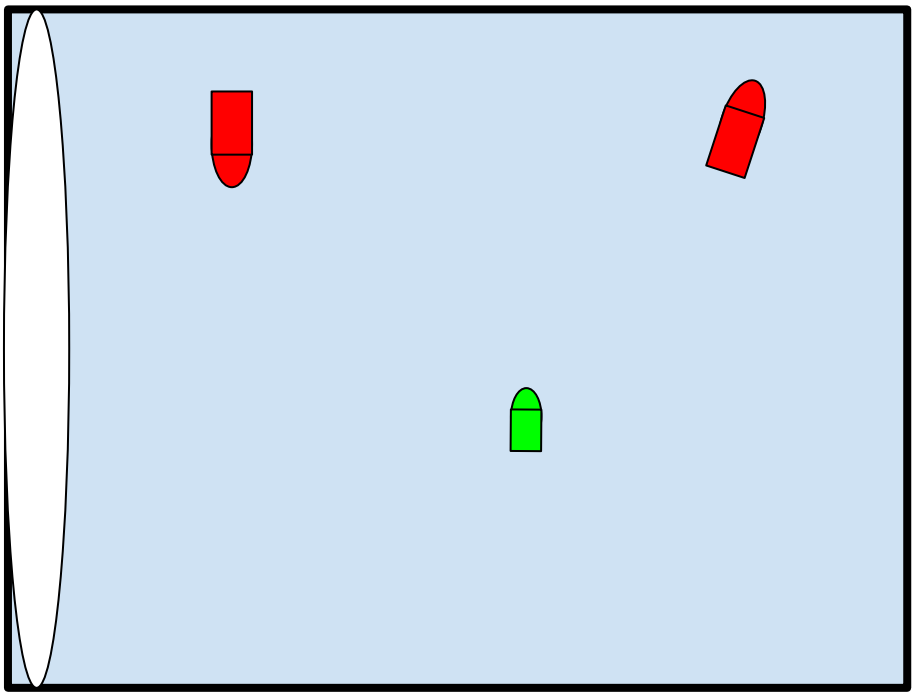

Figure 7: Example Scenario 2: Our vessel of interest (in green) trying to navigate out in open sea with vessels (in red) and land mass (in white) and sea water (in blue)

From the literature review, scenario 2 is the most commonly researched problem with the eventual solution relying mostly on radar detection with some fusion with camera information as well. However, scenario 1 is equally important for ASV navigation. For the rest of the paper, the focus will be on tackling scenario 1.

4.3. Distinguishing between static and dynamic targets in a crowded port scenario

Let us first define the concept of static and dynamic, and explore why there is a need to distinguish them. In scenario 1 from Figure 6 above, parked vessels and the dock platform are considered static obstacles while any moving vessels (classified as having speed greater than 0 or some threshold) are considered dynamic. Distinguishing between static and dynamic obstacles simplifies the navigation algorithm as static obstacles can be planned around while trajectory prediction is performed for dynamic obstacles based on its heading and velocity prediction.

Additionally, from the given sensor data (such as radar, lidar and camera), it is often difficult to distinguish the dock platform from the parked vessel and detections can falter between obstacles. Moreover, with occlusions, this can make obstacle tracking challenging. Additionally, certain obstacles cannot be captured in a single sensor detection reference and requires multiple readings over a long period (extended object tracking, [8]) in order to observe the full object. This extended object tracking problem is extremely challenging using standard “detect then track” architectures for obstacle tracking.

4.4 Representation of static and dynamic obstacles

Most of the literature ([3], [4], [5]) approaches this problem by tracking each obstacle as a centroid with a corresponding shape and state estimate. However, as mentioned in section 4.3, the entirety of obstacles may not be observed in a single frame and occlusions make it difficult to use this method. Instead, I propose the usage of probabilistic grid maps to store static obstacles and centroid with state estimates for dynamic obstacles. By combining these methods, we are able to remove some of the disadvantages of each method. Using probabilistic grid maps alone to mark locations where detections have occurred will not suffice as it will result in large “streaks” across the grid map caused by dynamic obstacles and sensor noise. Moreover, it is not able to easily provide information of state estimates for dynamic obstacles.

4.5 Theory of proposed algorithm

Before running into the main algorithm, each of the proposed sensor data will be individually pre-processed to remove noise and reduce data density to ease computational operations on the data, without losing crucial features.

We define a probabilistic grid map of size $N\,x\,N$ with resolution r. This grid map represents the probability of any particular grid cell on the grid map being static or stationary, as defined in section 4.3. The actual chosen size of the grid will depend on the movement capabilities of the surface vessel, and is further discussed in section 5. Range based sensors (such as lidars and radars) will be utilized to update the probabilities of each grid cell.

We also define an obstacle tracking framework for moving obstacles. Note that in the context of marine environments, boats, vessels, and ships are defined as moving obstacles. In this framework, lidar, radar and camera sensor detections can be used to determine the type of obstacle and the velocity of the obstacle. The obstacles will be tracked via their centroids and state estimates.

The static probabilistic grid map will provide the dynamic obstacle detection framework with information that can help to prune false positive tracks on known static obstacles.

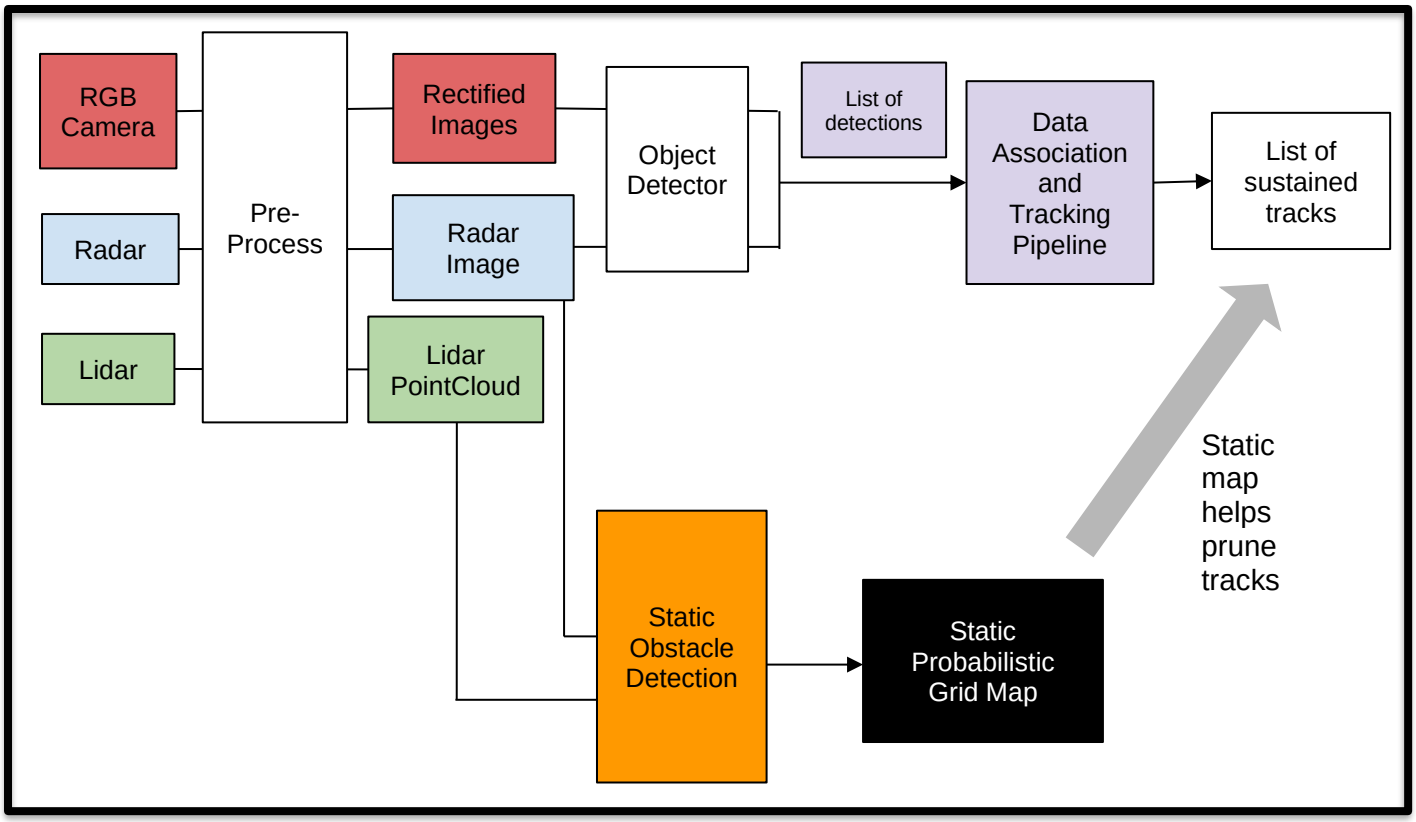

Figure 8: High level overview of proposed perception system

4.5.1 Static probabilistic grid map update

For the static probabilistic grid map, a set of parameters are defined for each range sensor. These parameters are:

Variables	Description
R	Max Sensor Range
S	Current Sensor Reading (containing 2D/3D Euclidean points in array format)
c	The corresponding 2D grid cell coordinate of a particular data point S[i] from S
conf	Confidence of a particular data data point S[i] from S based on the specific sensor model of the sensor. In range [0,1]
W	Size of the rolling window used to store the number of sensor hits on a particular grid cell for the past W sensor readings
p_s	Fraction of successful hits at a particular grid cell for the past W sensor readings within sensor range R
p_f	Fraction of failed hits at a particular grid cell for the past W sensor readings within sensor range R

Functions	Description
`intersects(S,c)`	Checks if any reading in `S` projects into grid cell coordinate `c`. Returns True/False
`cost(c,t)`	Grid cost at grid cell coordinate `c` at time `t`
`static_reward(c, p_s, conf)`	Static reward function for the grid map cell given historical sensor readings at grid cell `c`
`temp_decay(c, p_f, conf)`	Temporal Decay function for the grid map cell given historical sensor readings at grid cell `c`

A cost update algorithm is defined below based on these parameters and functions to update the cost of each cell based on the current and historical range sensor readings.

Algorithm 1 Static probabilistic grid map cost update algorithm

$cost(c,t) \leftarrow 0$
while $i < S.size()$ **do**
  $c, p_s, p_f, conf \leftarrow$ Update from $S[i]$
  $increment \leftarrow static\_reward(c, p_s, conf) \times intersects(S, c)$
  $decay \leftarrow temp\_decay(c, p_f, conf) \times (1 - intersects(S, c))$
  $cost(c,t) \leftarrow cost(c, t-1) + increment - decay$
  $i \leftarrow i + 1$
end while

The general idea behind this cost update equation is that grid cells with a historically high rate of detection should have increased reward growth. Similarly, grid cells with a history of low detection rates have a faster rate of decrease of cost. Simple linear models of reward and decay may not work well, and hence more complex polynomials or exponentials can be explored. Moreover, the sensor model of the range sensor should also be considered via the sensor reading confidence function as lidars and radars have different sensor models. In the ideal world, 2D and 3D ray tracing can be done to help update cost values much more

accurately. However, these are extremely computationally intensive. For the algorithm to run in real time, an approximation is made to increase the cost of the cell at which the reading was observed.

Another assumption is that false positives are rare and most likely removed by the preprocessor stage. However, even if noise crept in and increased costs of grid cells in a wrong manner, the temporal decay would eventually take care of it. As long as the noise is random, the temporal decay will eventually decrease the cost of the grid cell back to 0. The grid cell cost is proportional to the probability that an obstacle is static at that particular grid cell.

4.5.2 Dynamic obstacle tracker framework

For the dynamic obstacle tracking portion of the framework, it will be heavily reliant on the radar and camera. The pre-processing implementation is described in section 5.2. The radar obstacle detection will involve clustering techniques to identify separate regions of the data that are of interest. Similarly, a state-of-the-art image detection method is used for the camera to detect objects of interest. Further details of the object detection implementation are described in section 5.2. The output of the radar object detector will provide centroid coordinates of each cluster, and the dimensions of the cluster in metric units. The output of the camera object detector will provide object labels as well as the object yaw with respect to the ASV.

The high-level description for the radar object tracking algorithm is provided below. In summary, it is a simple distance-based cluster association technique, after which the Kalman filter is used to track its state estimate. There are other hyperparameters involved such as track initialization time, which will be further discussed in section 5.3.

Algorithm 2 Radar object tracking algorithm

```
tracks ← {}
while True do
    detections ← radar.detect()
    i ← 0
    while i < tracks.size() do
        track ← tracks[i]
        detection ← closest_detection(track, detections)
        if not valid_association(track, detection) then
            track.update(null)
            continue
        end if
        track.update(detection)
        i ← i + 1
    end while
    detections ← unassociated detections
    tracks.init(detections)
end while
```

Once the radar is able to track obstacles, it will be fused with the camera information to associate the object type (such as ship, dock, land mass, etc) with the radar track to better estimate its tracking profile. The radar camera association algorithm is as described below. The ability to obtain decent quality yaw object estimations from the 2D image is a crucial factor exploited to help associate the radar and camera detections. Further details are explained in section 5.3.

Algorithm 3 Radar-Camera association algorithm

```
while True do
    tracks ← radar.get_tracks()
    tracks ← transform(tracks, camera_frame)
    detections ← camera.detect()
    i ← 0
    while i < detections.size() do
        detection ← detections[i]
        track ← find_nearest_track_by_yaw(detection, tracks)
        if not valid_association(track, detection) then
            continue
        end if
        track.update(detection)
        i ← i + 1
    end while
end while
```

Finally, with the static probabilistic grid map and the list of radar tracks available, one last fusion is performed to prune away tracks at locations where the centroid and and shape of the tracks intersect with a known static location based on the static probabilistic grid map. For simplicity, a simple probabilistic threshold can be utilized to determine the pruning level for the generated tracks. The end result is a probabilistic static grid map that displays the highly likely static obstacles and the obstacle tracker that provides the state estimates for trajectory prediction of the dynamic obstacles. This should allow the Bumblebee ASV to reliably detect and track static and dynamic obstacles to facilitate safe navigation in the crowded port scenarios.

5. Implementation

With the proposed static and dynamic obstacle tracking framework defined, this section will focus on the implementation details to enable the algorithms to work on real life Bumblebee ASV sensor data.

5.1 Pre-processing sensor data

5.1.1 3D Lidar data

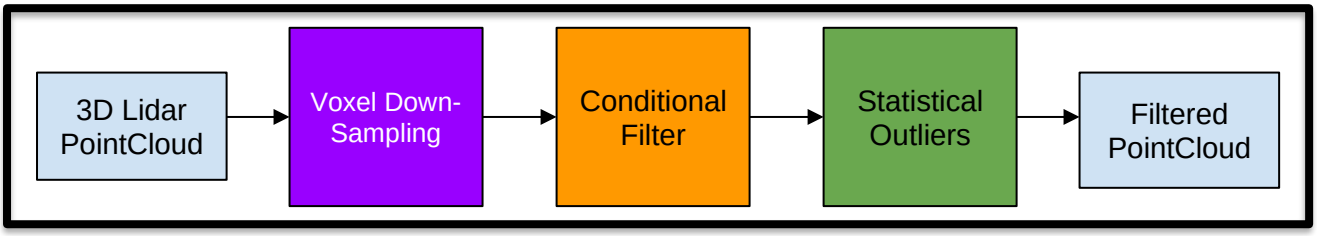

Figure 9: Lidar pre-processing pipeline

For 3D lidar data, a voxel filter was used to down-sample the pointcloud produced to reduce the computational load of processing. A conditional filter was then applied to remove points that intersected with the vehicle model. For simplicity, a box model was used to represent the Bumblebee ASV and all points within the box were removed. Lastly, a statistical outlier was applied to remove random noise that might appear by computing the mean of the k nearest neighbours of each point and removing points that were not within m standard deviations of the mean. For the Ouster OS1-64 lidar utilized on the Bumblebee ASV, $k = 10$ and $m = 1.0$ provided the best results, even in light rain conditions where lidar noise increased significantly. The filtering functions were implemented with the help of the PointCloudLibrary (PCL) in C++ [9].

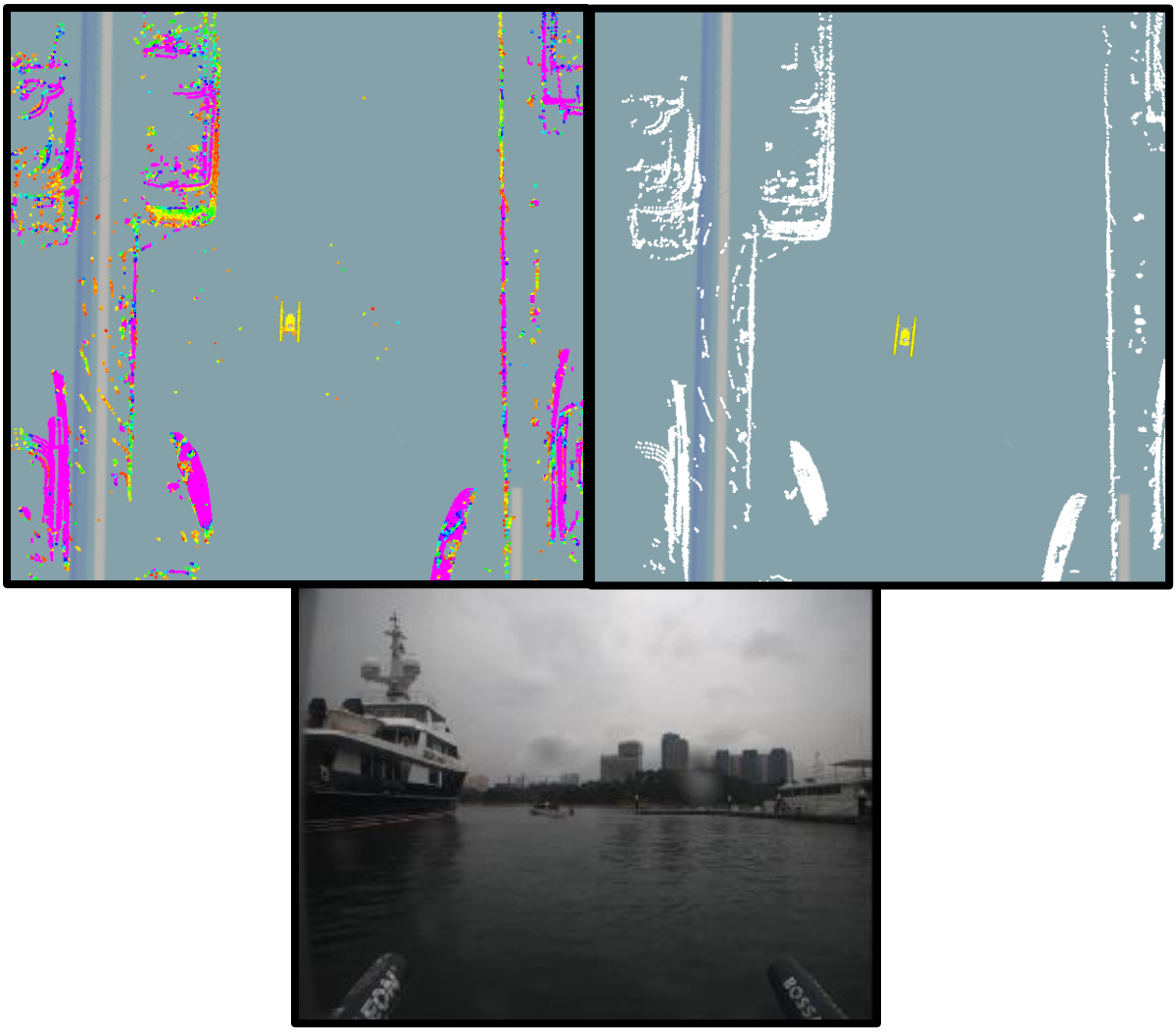

Figure 10: (Left) Raw lidar pointcloud, (Right) Filtered lidar pointcloud downsampled and with noise removed, (Below) Image feed for reference in slight rain condition

5.1.2 Radar data

With the radar being a new sensor on the Bumblebee ASV, this was the first time the radar was utilized onboard the vessel. As such, a custom Linux and ROS driver was developed for it, providing polar images of the raw readings.

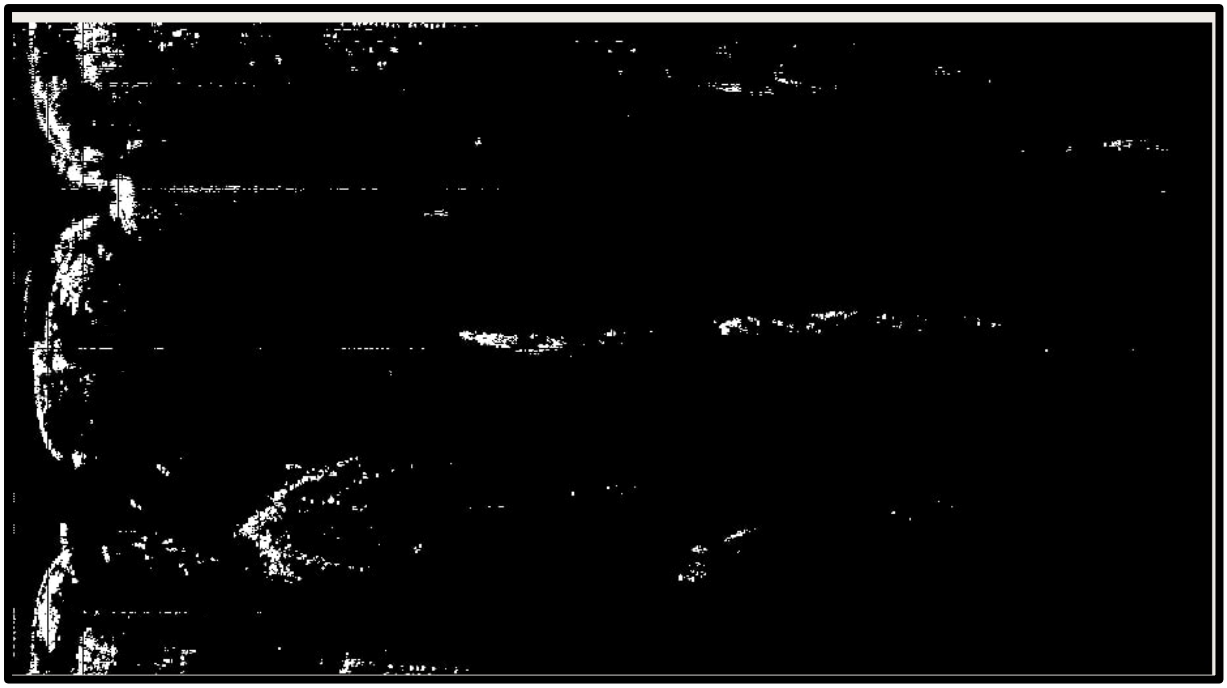

Figure 11: Raw radar polar image with noise, vertical axis represents yaw from 0 to 2π, horizontal axis represents range from 0m to 1500m

A median filter and line filter was applied to the image to remove random noise. This was implemented with the help of the OpenCV framework [10] in C++. This polar image was then converted to a 3D pointcloud, after which the same conditional filter (used for the lidar) was applied to remove points that intersected with the Bumblebee vehicle.

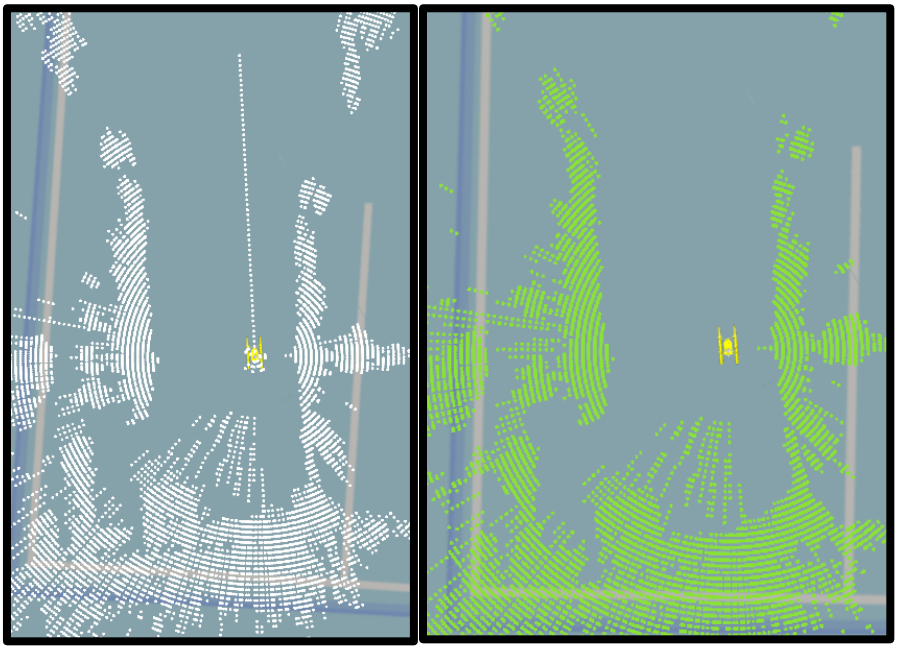

Figure 12: (Left) Raw radar pointcloud overlay, (RIght) Filtered radar pointcloud

The filtered 3D pointclouds from the lidar and radar were overlaid on a satellite image to check for accuracy and reliability.

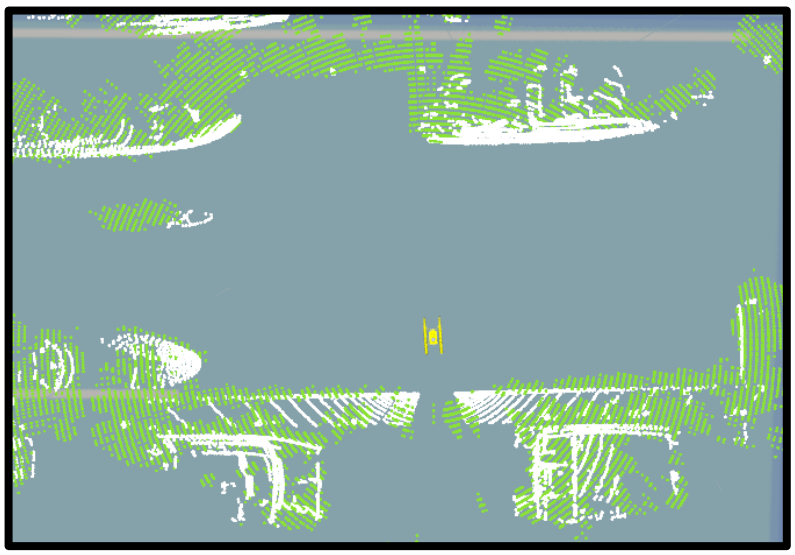

Figure 13: Zoomed in view of radar pointcloud (green) and lidar pointcloud (white) for near range 50m Perception

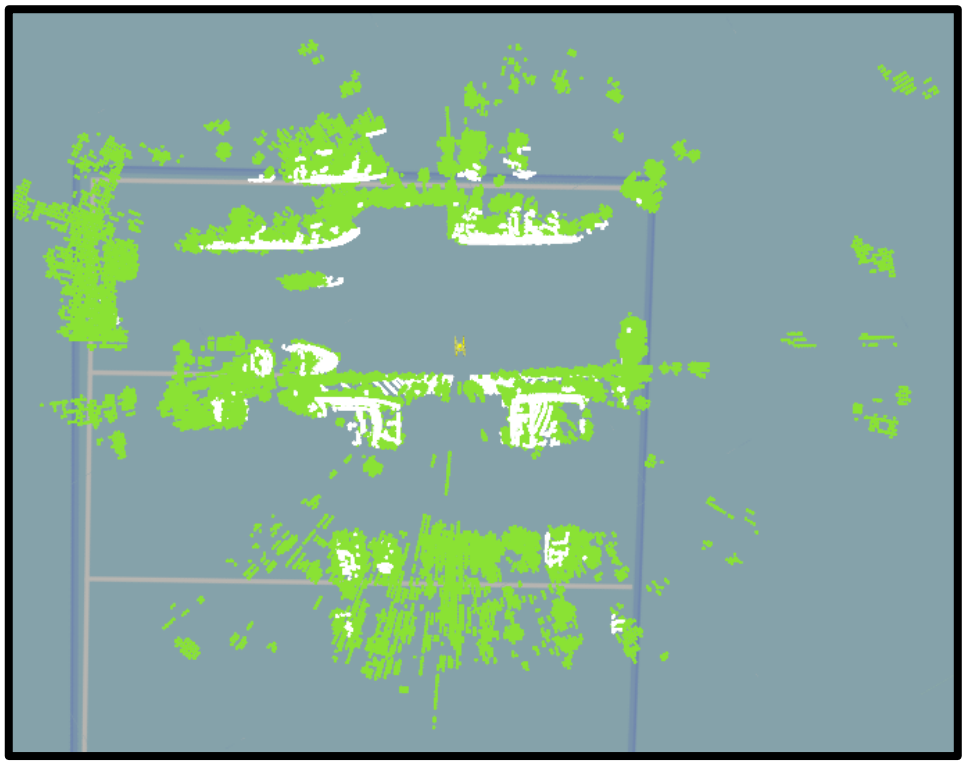

Figure 14: Further range 250m overlay of radar and lidar pointcloud data in crowded port with moving ships

5.1.3 Camera data

Lastly, the RGB cameras were fully calibrated using checkerboards to obtain the camera intrinsic matrix, which would be useful for the following section. This was done using OpenCV [10].

5.2. Object detectors

For the radar, an image-based object detector was utilized via OpenCV thresholding and contour finding with a convex hull.

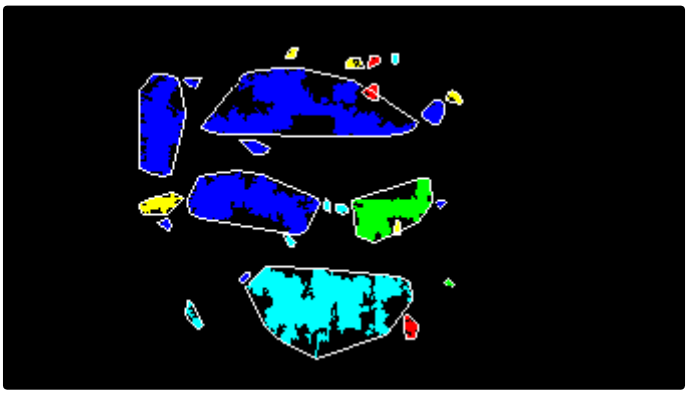

Figure 15: Radar clusters surrounded by white outline

For the camera, a transfer learning approach on a state-of-the-art image detector was utilized. The YOLOv5 image detector was trained on the Singapore Maritime Dataset (SMD). This transfer learning approach on existing state of the art deep learning image detectors is a standard approach used in marine obstacle detection and marine perception algorithms.

Given the 2D planar nature of motion of objects in the sea, the relative horizontal displacement and the relative angular displacement between the surface vessel and the targets of interest are the key parameters. These can be utilized to assist in obstacle avoidance and navigation of the surface vessel.

An estimate of the relative x,y and yaw states between the surface vessel and the targets of interest can be inferred via the projection of 2D object detection bounding boxes into 3D space via intrinsic camera properties. A crucial assumption is made:

1. The detected object has negligible roll or pitch relative to the flat-water surface
2. The bottom edge of the detected bounding box is close to the water surface

With these assumptions, a decent estimate for the relative x,y and yaw states can be obtained. Figure 16 shows implementation on the detection of a boat. The projected frustum (shown in green on the right) is overlaid with ground truth 3D lidar point cloud data.

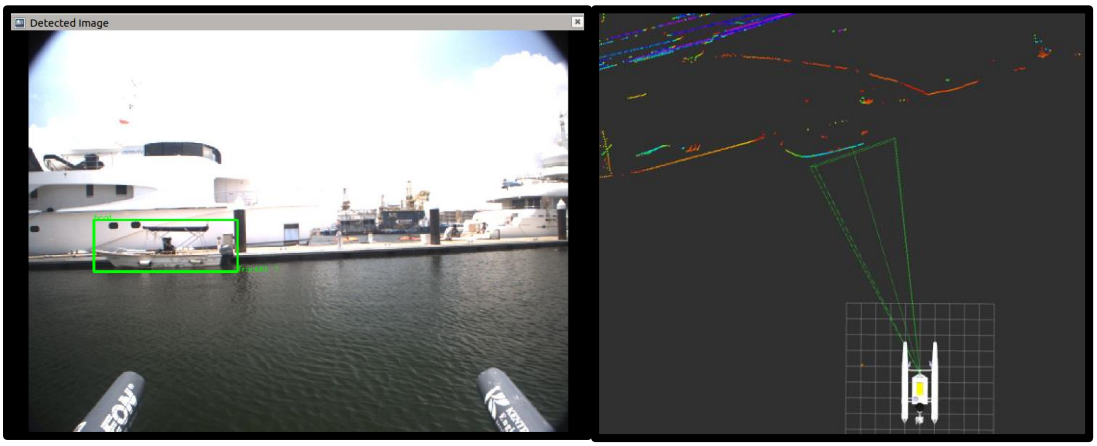

Figure 16: Detection of boat via monocular camera and 3D projection of frustum

The relative yaw between the ASV and the target is only reliant on the extrinsics of the ASV (namely roll and pitch) as well as the horizontal pixel displacement of the 2D bounding box center with the camera center from the camera intrinsic matrix. Hence, it is relatively accurate in most of the scenarios within some tolerance and can provide very useful information to the ASV when other sensors are unable to detect them due to range or size issues (e.g. lidar and radar).

5.3 Radar obstacle tracking

Once the radar clusters are detected, it is passed to a data association and tracking module. For ease of implementation, an existing ROS tracker package [11] was modified for this use case and the constant velocity model Kalman filter was utilized to predict the state estimate of the obstacle. This ROS package also assists with the visualization of tracks.

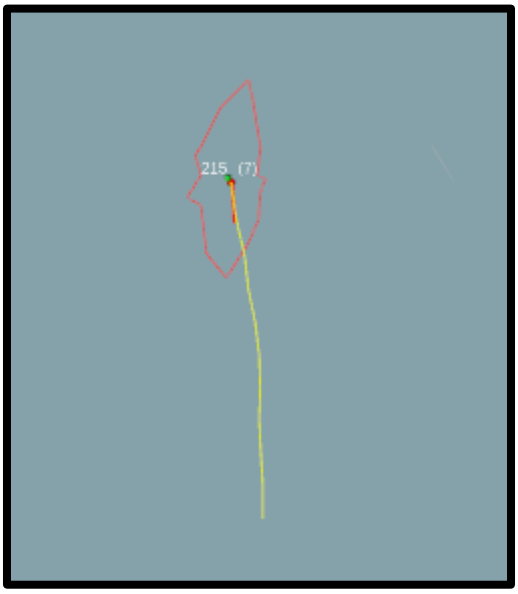

Figure 17: Sample radar track with cluster outline (in red), previous track motion (yellow path) and the unique track id (white text)

5.4 Static probabilistic grid

Given the Bumblebee ASV's top speed of 5 knots, and it's holonomic manoeuvrability, the perception region of interest is defined to be 300m. This region size is also similar to the competition area size that the Bumblebee ASV is deployed to for the international RobotX competition [12]. This allows us to exclude data outside of this range. As for the grid resolution, given the localization accuracy of the Bumblebee ASV (refer to section 6), a 1m grid resolution will suffice. The terms mentioned in section 4.5.1 will be referenced in this section extensively.

Next, let us define the sensor detection confidence profile. An experimental study was performed by the manufacturer of the Ouster lidar for its older generation model, providing the detection probability profile across ranges for a 10% Lambertian reflective target [13]. However, as we are using a new and more powerful version of the lidar, I have extrapolated an approximate estimate of the probability detection profile based on the Ouster study.

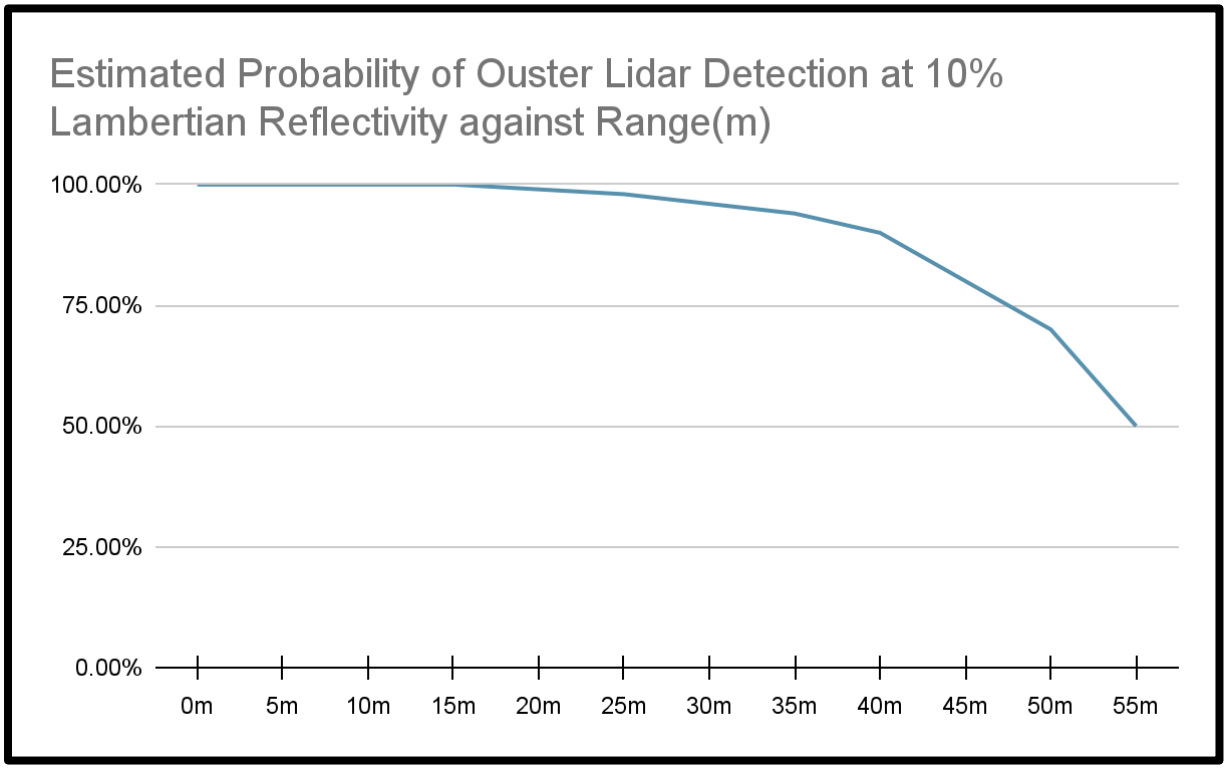

Figure 18: Lidar detection probability against range graph

The sensor detection confidence profile was not available for the marine radar and hence an estimated chart was extrapolated based on the self-collected data. Note that there is a very small dead zone at close distances to the radar, which accounts for the initial low detection probability in Figure 19.

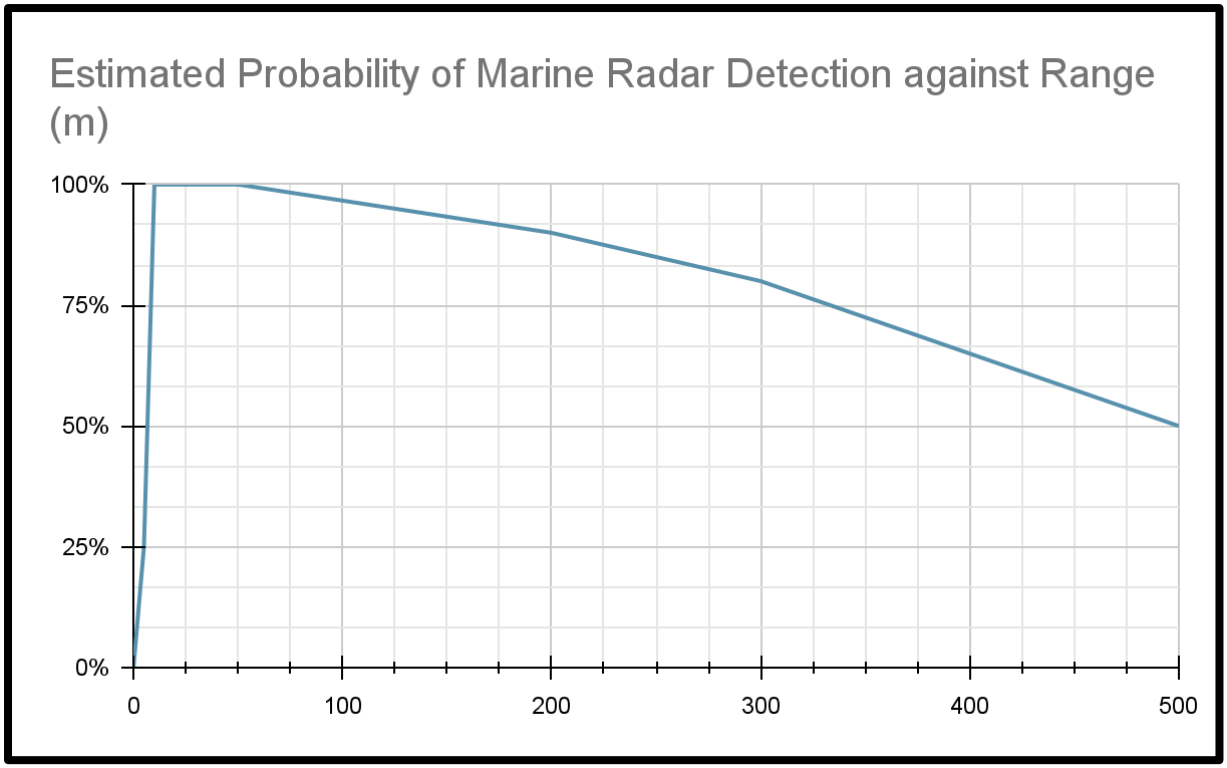

Figure 19: Radar detection probability against range graph

The $intersects(S, c)$ function is implemented such that if any of the sensor readings S at time t were to project and hit grid coordinate c , it would return 1, otherwise 0. Moving on to the rolling window parameter W, $W = 10$ was chosen empirically.

Lastly, the static reward and temporal decay functions need to be defined for each range sensor. Based on data analysis, it was observed that the 3D lidar had very few false positives. However, there were a lot of false negatives due to occlusions. Therefore, the static reward function was chosen with a rapid growth rate while the temporal decay was chosen with slow growth rate. Moreover, due to the low false positive rate, the static reward growth rate was increased inversely proportionally to the sensor detection probability.

In contrast, the marine radar was observed to have more false positives due to sensor noise, but less false negatives due to robustness against occlusions from the radio signal. As

a result, a slower growth rate reward function was chosen while a more rapid temporal decay function was chosen.

$$static\ reward\ \propto \frac{p_s}{conf}$$

$$temporal\ decay \propto\ cost(c, t-1) -\ cost(c, t-1)^{p_f}$$

The individual sensor growth rates were controlled more precisely for each sensor type using some empirically found constants for each equation. By combining the cost updates from both sensors on to the same grid map, the disadvantages for both sensors are greatly reduced and the resulting grid map (in the region overlapping the radar and lidar) is more accurate than what each sensor could have provided.

6. Evaluation

6.1 Experiment methodology

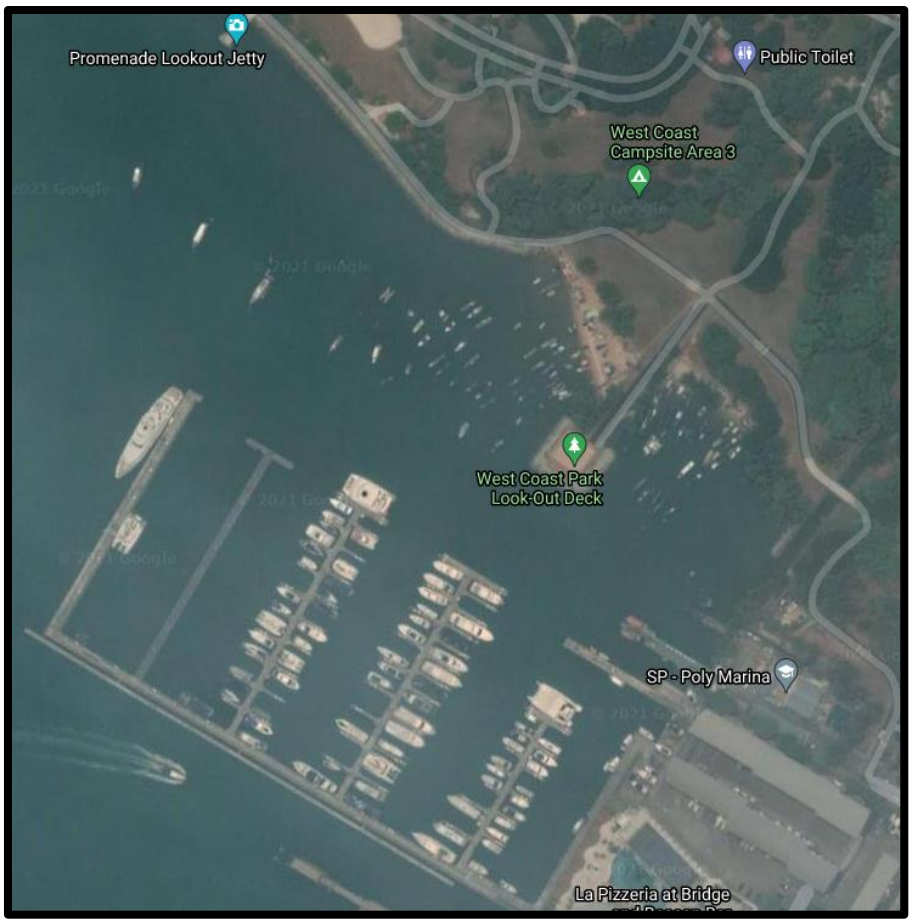

Figure 20: Google Maps satellite image [14] of RSYC testing location for ASV

Despite the repeated lockdown and restrictions in Singapore due to COVID-19, real-life data was collected from the Bumblebee ASV system at the Republic of Singapore Yacht Club (RSYC) with the help of the Bumblebee Team. To evaluate the proposed perception system, 2 test scenarios were generated. The first test scenario would involve a ASV parked along the dock with it trying to get out of the dock area with stationary ships surrounding it. To carry out autonomous testing at RSYC, a safety boat is required and hence this safety boat was utilized as the stationary or dynamic obstacle that we could control as the rest of the RSYC marina had boat traffic that was not in our control. The second scenario would be a

repeat of the first scenario except with a moving obstacle present in the crowded dock. Due to the prevalent weather conditions during the entire month of data collection, almost all of the data collected were in cloudy or rain conditions.

Each of these 2 scenarios would be replicated by tele-operating the Bumblebee ASV near the RSYC dock via radio control along with instructions to the safety boat driver to adjust the obstacle path accordingly. The localization stack of the ASV involves a navigation grade GPS-INS system with onboard Kalman filter processing for filtering. As such, the local localization output is of high accuracy (even without RTK) and is generally accurate to around 1m for positioning. In the context of the ASV (which is 5 meters long) and the large perception area of interest (300 meters), this localization error can be considered relatively negligible.

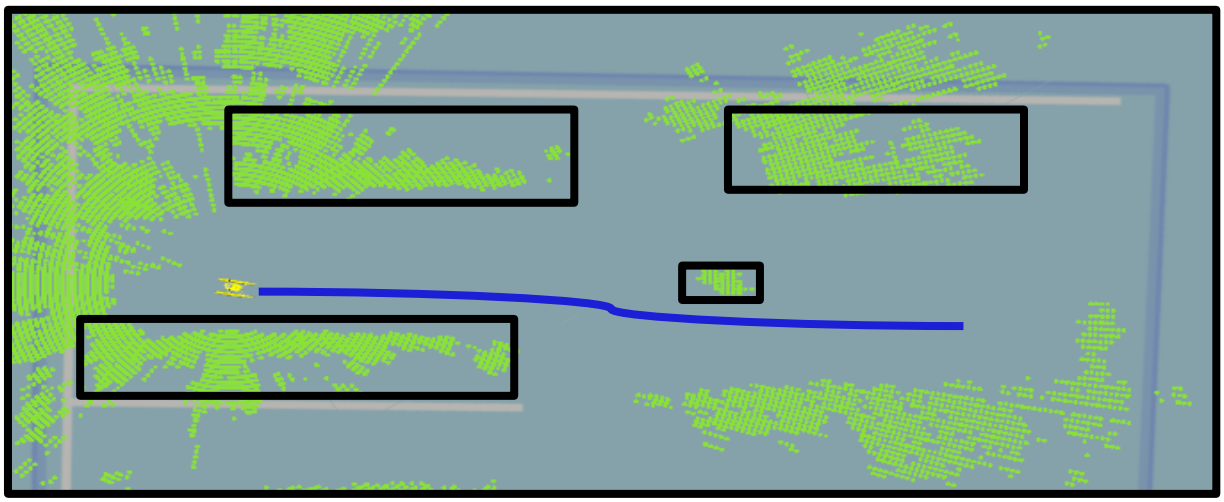

Figure 21: Scenario 1, all stationary ships and boats in a crowded dock. Black bounding boxes represent parked ships, blue path indicates path taken by ASV

For scenario 1, all the ships are parked adjacent to the dock platform and the safety boat is stationary in the middle of the dock. The ASV then moves from its parked location to near the end of the dock. Without a robust method to generate ground truth data, manual identification of the known scene with obstacles was performed on the raw sensor data. The length of this dock is roughly 150m.

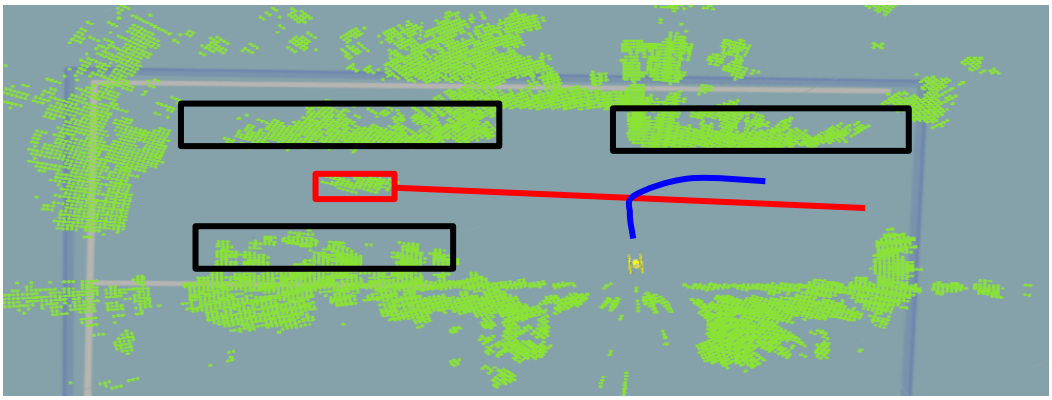

Figure 22: Scenario 2 moving obstacle and moving ASV in crowded dock. Black bounding boxes represent parked ships, red box represents moving obstacle with red path as the path taken, blue path indicates path taken by ASV

For scenario 2, all but one of the vessels are parked adjacent to the dock platform and the safety boat is moving in a straight-line motion. The ASV then moves from its parked location to near the end of the dock while (manually) avoiding collision with the safety boat.

The reliable identification of static obstacles and dynamic obstacles will be utilized as the evaluation metric for this perception system. For static obstacles, landmass and dock obstacles will be identified via satellite images and overlaid for comparison. For stationary ships, the black bounding boxes are applied to each of the scenarios. For dynamic obstacles, the red bounding box with the red path is identified in scenario 2. The final evaluated dynamic object track path will be evaluated along with the reliability of the track.

6.2 Results

The perception algorithm generally performed well for both the crowded port scenarios, producing an accurate static map and a dynamic obstacle track capable of providing accurate position and velocity.

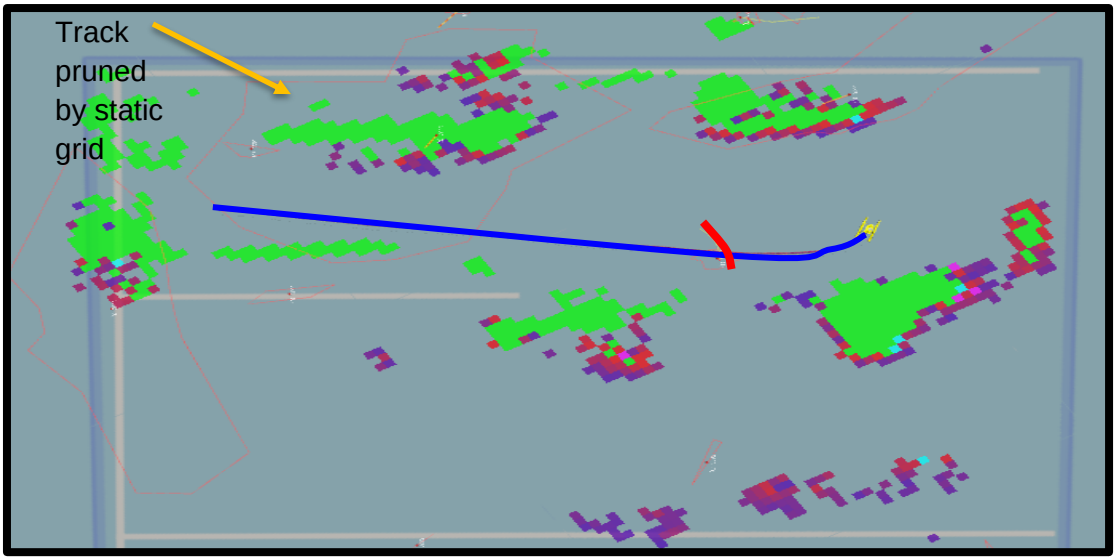

Figure 23: Scenario 1 result. Dark purple indicates low probability of being static, while green indicates almost 100% probability of being static. Red line shows path of "static" obstacle boat parked in the middle

For scenario 1, the ASV drove from a parked location to near the end of the port while avoiding a static boat parked in the middle. While the boat was intended to be stationary to test the robustness of the static probabilistic grid, the result showed that the boat driver actually struggled to maintain the position in the waves hence actually drifted away from its starting pose. Hence, the red line can be seen tracked on the above figure drawn by the obstacle tracking algorithm. As a result, the probabilistic grid map avoided mapping that obstacle as a static obstacle and hence that part of the grid is empty (0% static probability). The rest of the 3 boats were docked physically to the port and hence are guaranteed to be stationary. As such, the probabilistic grid picked it up and it is displayed as green pixels which are almost 100% probability of static obstacles.

Additionally, there are many red clusters drawn in the above figure. These indicate that the tracking algorithm alone is trying hard to predict a position for a cluster that changes centroid position and shape based on the occlusion. Hence, the wrong velocity and position are being tracked for the cluster. But as the static probability grid classified the points as highly

likely to be static, these tracks are pruned away, and we are left with only the 1 semi-moving obstacle and the static probabilistic map.

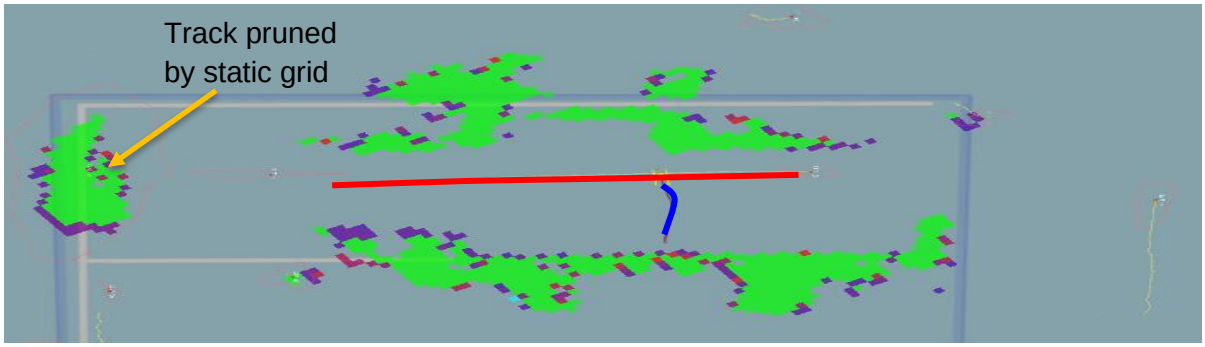

Figure 24: Scenario 2 result. Dark purple indicates low probability of being static, while green indicates almost 100% probability of being static. Red line shows path of dynamic obstacle boat moving in crowded port, blue line indicates ASV path

For scenario 2, the ASV moved from the park location in a motion perpendicular to the dynamic obstacle. The track for the dynamic obstacle was sustained for the entire duration and hence successful. Moreover, on the right-hand side of the figure, there was a larger ship much further away that it had managed to track (as shown in red ellipse and yellow line on the right-hand side of the figure). This further shows the robustness of the tracker to pick up dynamic obstacles. As in scenario 1, the static probabilistic grid correctly identified all the static areas within the port and was able to prune redundant tracks from the obstacle tracking module, hence reducing the number of false positive tracks.

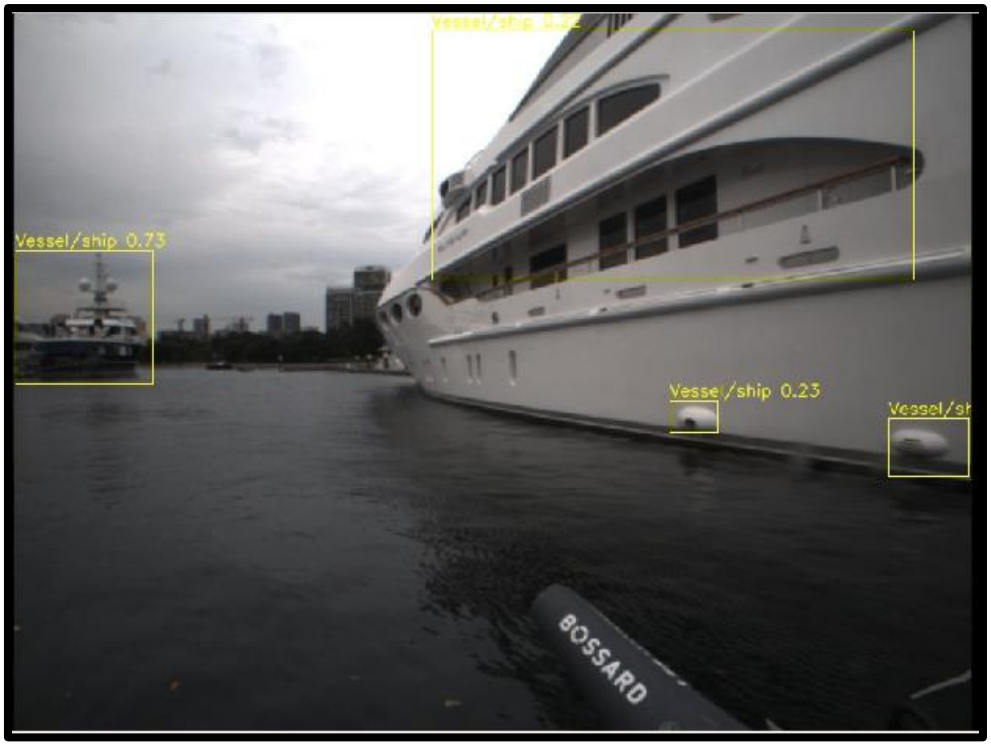

Figure 25: Difficulty in image object detections due to ship being too close

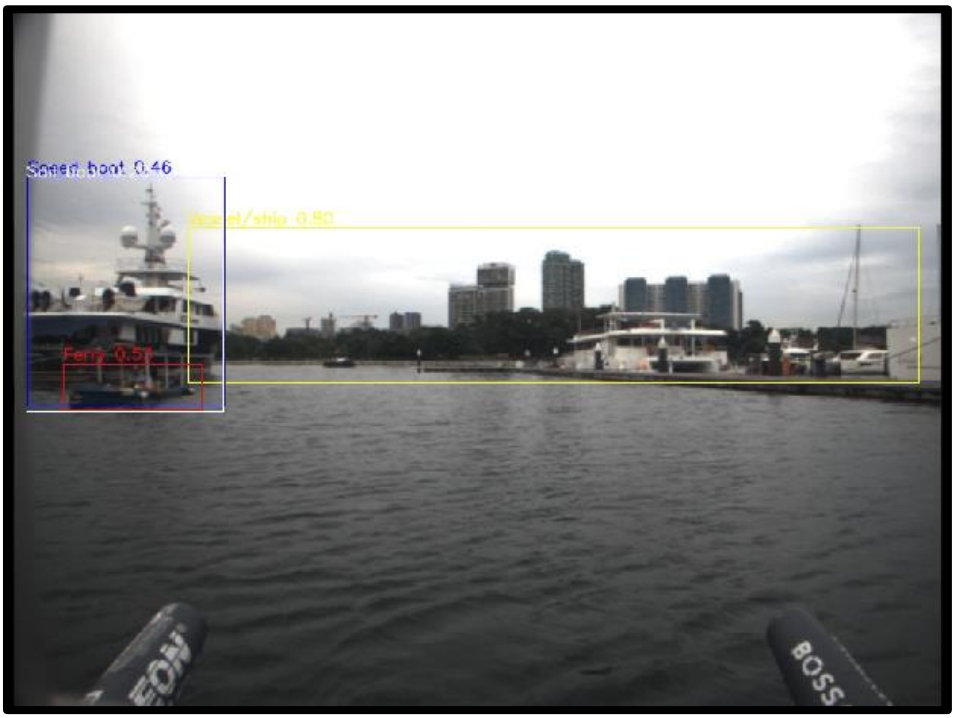

Figure 26: Detection of moving ferry (red bounding box on left)

In both scenarios, the camera object detector managed to detect the moving safety boat and identify it correctly, allowing for fusion with the radar obstacle tracker. However, this only typically occurred at closer distances, and many of the parked vessels were too large to

be captured in a single camera image, hence failing to provide a detection. Moreover, as seen in Figure 26, there were times when both vessels had a similar yaw angle, making it difficult to fuse with the radar as either vessel could be the target. Fortunately, the vessel width criteria allow the correct vessel of interest to be tracked.

7. Conclusion and future work

7.1. Conclusion

In this project, a perception system for the Bumblebee ASV was developed to aid in navigation and obstacle avoidance. A new dataset for near shore navigation was also collected via the Bumblebee ASV. The project had a greater focus on nearshore and port scenarios given the lack of literature available for it. An algorithm was developed to segment the static obstacles on a probabilistic grid map along with a dynamic obstacle tracker. These 2 frameworks exchanged data to complement each other, which improved the reliability and accuracy of tracks and static obstacles. This perception system allows for obstacle avoidance capabilities of the Bumblebee ASV to be developed.

However, it was challenging to evaluate the effectiveness of camera fusion with the radar tracking in this dataset as the low light conditions and bad weather made it difficult to spot ships. Moreover, even ships that were spotted were too close and only parts of it could be seen, hence not producing a camera object detection. Fortunately, the radar obstacle tracking and probabilistic static grid map were sufficient for the test case scenarios.

7.2. Future work

While the results of the perception system have been decent, more testing needs to be done to evaluate its robustness in different types of nearshore scenarios. Additionally, the next step would be to build upon this perception system to develop autonomous obstacle avoidance capabilities for the Bumblebee ASV.

Moreover, the radar target tracking algorithm can be improved to use more advanced techniques such as JPDA (Joint Probabilistic Data Association Filter) to help with the data association between frames of radar detections. The current simple method will suffer from a swapping of track ids if multiple vessels cross paths at close distances.

Lastly, the camera object detector trained via YOLOv5 can be improved significantly by providing more training data from the Bumblebee ASV system. Most of the data from the Singapore Maritime Dataset was collected from on-shore, and in good weather whereas the conditions faced by the Bumblebee ASV during data collection had much lower light conditions along with some rain. This was partially the cause of the poor object detector performance. Basic data augmentation techniques could also be performed. The radar centered dynamic obstacle tracking framework could also incorporate lidar data at closer distances to improve the accuracy and speed of the track given that the lidar provides data updates 10 times more often than the radar.

8. Bibliography

[1] Y. Gu, J. Góez, M. Guajardo and S. Wallace, “Autonomous vessels: state of the art and potential opportunities in logistics,” *International Transactions in Operational Research,* vol. 28, March 2020.

[2] J. Fayyad, M. Jaradat, D. Gruyer and H. Najjaran, “Deep Learning Sensor Fusion for Autonomous Vehicle Perception and Localization: A Review,” *Sensors,* vol. 20, p. 4220, July 2020.

[3] J. Han, J. Kim and N.-s. Son, “Persistent automatic tracking of multiple surface vessels by fusing radar and lidar,” in *OCEANS 2017 - Aberdeen*, 2017.

[4] M.-H. Haghbayan, F. Farahnakian, J. Poikonen, M. Laurinen, P. Nevalainen, J. Plosila and J. Heikkonen, “An Efficient Multi-sensor Fusion Approach for Object Detection in Maritime Environments,” 2018.

[5] X. Mou, X. Chen, J. Guan, B. Chen and Y. Dong, “Marine Target Detection Based on Improved Faster R-CNN for Navigation Radar PPI Images,” 2019.

[6] B. Bingham, C. Agüero, M. McCarrin, J. Klamo, J. Malia, K. Allen, T. Lum, M. Rawson and R. Waqar, “Toward Maritime Robotic Simulation in Gazebo,” in *OCEANS 2019 MTS/IEEE SEATTLE*, 2019.

[7] M. Ejaz and M. Chen, “Sliding mode control design of a ship steering autopilot with input saturation,” *International Journal of Advanced Robotic Systems,* vol. 14, May 2017.

[8] K. Granström and M. Baum, “Extended Object Tracking: Introduction, Overview and Applications,” *CoRR,* vol. abs/1604.00970, 2016.

[9] R. B. Rusu and S. Cousins, “3D is here: Point Cloud Library (PCL),” in *2011 IEEE International Conference on Robotics and Automation*, 2011.

[10] G. Bradski, “The OpenCV Library,” *Dr. Dobb's Journal of Software Tools,* 2000.

[11] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monrroy, T. Ando, Y. Fujii and T. Azumi, “Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems,” in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, 2018.

[12] Robonation, *RobotX Challenge,* 2021, p. 15.

[13] *The OS0 wide-view Lidar Sensor: Deep Dive,* 2021, p. 20.

[14] “. Rsyc Satellite Image, *Google maps,* 2021, p. 3.

[15] D. K. Prasad, D. Rajan, L. Rachmawati, E. Rajabally and C. Quek, “Video Processing From Electro-Optical Sensors for Object Detection and Tracking in a Maritime Environment: A Survey,” *IEEE Transactions on Intelligent Transportation Systems,* vol. 18, pp. 1993-2016, 2017.