

12-2014

Development of an Exteroceptive Sensor Suite on Unmanned Surface Vessels for Real-Time Classification of Navigational Markers

Christopher Lloyd Kennedy
Embry-Riddle Aeronautical University - Daytona Beach

Follow this and additional works at: <https://commons.erau.edu/edt>



Part of the [Aeronautical Vehicles Commons](#), and the [Mechanical Engineering Commons](#)

Scholarly Commons Citation

Kennedy, Christopher Lloyd, "Development of an Exteroceptive Sensor Suite on Unmanned Surface Vessels for Real-Time Classification of Navigational Markers" (2014). *Dissertations and Theses*. 167.
<https://commons.erau.edu/edt/167>

This Thesis - Open Access is brought to you for free and open access by Scholarly Commons. It has been accepted for inclusion in Dissertations and Theses by an authorized administrator of Scholarly Commons. For more information, please contact commons@erau.edu.

DEVELOPMENT OF AN EXTEROCEPTIVE SENSOR SUITE ON UNMANNED
SURFACE VESSELS FOR REAL-TIME CLASSIFICATION OF NAVIGATIONAL
MARKERS

by

Christopher Lloyd Kennedy

A Thesis Submitted to the College of Engineering Department of Mechanical
Engineering in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Mechanical Engineering

Embry-Riddle Aeronautical University
Daytona Beach, Florida
December 2014

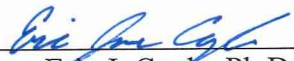
DEVELOPMENT OF AN EXTEROCEPTIVE SENSOR SUITE ON UNMANNED
SURFACE VESSELS FOR REAL-TIME CLASSIFICATION OF NAVIGATIONAL
MARKERS


by


Christopher Lloyd Kennedy

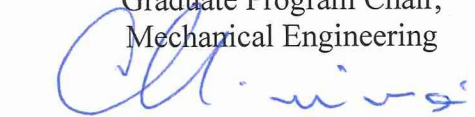
This thesis was prepared under the direction of the candidate's Thesis Committee Chair,
Dr. Eric J. Coyle, Professor, Daytona Beach Campus, and Thesis Committee Members
Dr. Charles F. Reinholtz, Department Chair, Professor, Daytona Beach Campus,
and Dr. Patrick N. Currier, Professor, Daytona Beach Campus, and
has been approved by the Thesis Committee. It was submitted to the
Department of Mechanical Engineering in partial
fulfillment of the requirements for the degree of
Master of Science in Mechanical Engineering

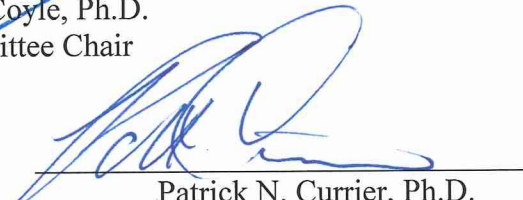
Thesis Review Committee:



Eric J. Coyle, Ph.D.
Committee Chair

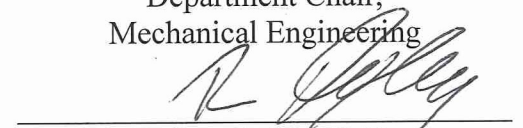

Charles F. Reinholtz, Ph.D.
Committee Member


Jean-Michel Dhainaut, Ph.D.
Graduate Program Chair,
Mechanical Engineering


Maj Mirmirani, Ph.D.
Dean, College of Engineering


Patrick N. Currier, Ph.D.
Committee Member


Charles F. Reinholtz, Ph.D.
Department Chair,
Mechanical Engineering


Robert Oxley, Ph.D.
Associate Vice President of Academics

December 2014

Acknowledgements

I would like to thank everyone who contributed in any way to the completion of this thesis. First and foremost my advisor, Dr. Eric Coyle, who sparked my interests in machine learning, and pushed me to explore new challenging ideas. Whose advice and encouragement through my Graduate education was invaluable. I will always be grateful for your eagerness to see me and every other student succeed.

Furthermore, the incredible team of engineers I was privileged to work with over the past few years and call close friends. Tim Zuercher, who without his technical advice, coding prowess, and continuous challenging, this work would never have had such an incredible success. Hitesh Patel, for giving me access to all of the tools and equipment I needed to succeed, and always being the first one to step-up-to-bat for me and the team whenever the time called, with great magnanimity. Gene Gamble, who's initial coding allowed much of the initial algorithms to be developed. Christopher Hockley, for your continuous encouragement and understanding. Also, for giving me the greatest opportunities throughout my time at Embry-Riddle.

To my committee members, Dr. Charles Reinholtz and Dr. Patrick Currier, who pushed me to create and innovate for many years, and who have advised and taught me many lessons in my years as a student. Thank you for helping finance my education and giving me more opportunities than I could ask for.

I would also like to acknowledge Abraham Feldman, taking time to improve the writing in this thesis.

Finally, I would like to thank my friends and family, who have been there to support me in all of my ventures. Mom, Dad, and Grandparents for your continuous

unwavering encouragement. My good friends for looking after Reba, while I am away at competitions and not forgetting about me when I disappear into the lab for months at time. I am truly thankful to everyone who has encouraged me and pushed me to do what I want and helped me enjoy every moment along the way.

Abstract

Researcher: Christopher Lloyd Kennedy

Title: DEVELOPMENT OF AN EXTEROCEPTIVE SENSOR SUITE ON UNMANNED SURFACE VESSELS FOR REAL-TIME CLASSIFICATION OF NAVIGATIONAL MARKERS

Institution: Embry-Riddle Aeronautical University

Degree: Master of Science in Mechanical Engineering

Year: 2014

This thesis presents the development of an exteroceptive sensor suite for real-time detection and classification of navigational markers on Unmanned Surface Vessels. Three sensors were used to complete this task: a 3D LIDAR and two visible light cameras. First, all LIDAR points were transformed from the sensor's reference frame to the local frame using a Kalman filter to estimate instantaneous vehicle pose. Next, objects were chosen from the LIDAR data to be classified using either Multivariate Gaussian or Parzen Window Classifiers. Both produce 96% accuracy or better, however, multivariate Gaussian ran considerably faster than the Parzen and was simpler to implement and was therefore chosen as the final classifier. Additionally, regions of interest images based on the Multivariate Gaussian classification were extracted from the full camera images to improve marker knowledge. This sensor suite and set of algorithms underwent extensive testing on Embry-Riddle's Maritime RobotX and RoboBoat platforms and greatly improves the ability to quickly and accurately identify multiple navigational markers, which is paramount to the success of any Unmanned Surfaces Vessel.

Table of Contents

Acknowledgements..... 2

Abstract..... 4

List of Tables 6

List of Figures 7

Chapter I - Introduction 10

Chapter II - Review of the Relevant Literature 15

Chapter III - Methodology 18

 3.1 Kalman Filter Angular Corrections for Velodyne 22

 3.2 Classification..... 28

 3.3 Region of interest..... 42

Chapter IV - Results and Discussion 47

 4.1 Kalman Filter Angular Corrections for Velodyne Results 47

 4.2 Classification Results..... 52

 4.3 Region of Interest Results..... 55

Chapter V 59

Conclusions, and Recommendations 59

List of Tables

Table 1: Firing order of lasers on a Velodyne HDL-32E [10].....	20
Table 2: Region of Interest size parameters.....	46
Table 3: Multivariate Gaussian confusion matrix.....	52
Table 4: Parzen Window confusion matrix	54

List of Figures

<i>Figure 1:</i> Minion ASV, during a test in the Halifax river in Daytona Beach, FL.	12
<i>Figure 2:</i> Task one (navigation) at the 2014 MRC. [3]	13
<i>Figure 3:</i> Three 5m bays are identified by three different signs in task three. [3]	14
Figure 4: Task five, detection and avoidance. [3].....	14
<i>Figure 5:</i> Minion’s exteroceptive sensor suite.....	18
<i>Figure 6:</i> Velodyne 32E beam pattern section-cut out to 30m.....	21
<i>Figure 7:</i> Top down view of sensor coverage. Red is the beam pattern for the Velodyne 32E and green represents the field of view of the Microsoft LifeCams.	21
<i>Figure 8:</i> The Velodyne has six accelerometers and three gyroscopes oriented in line and around each axis respectively. [10].....	23
<i>Figure 9:</i> Raw data points of a dock form the Velodyne visualized in VeloView on the left, and a Google Maps view of the same dock on the right. [16].....	28
<i>Figure 10:</i> Initial occupancy grid showing the docks and shore on the right of the image, spurious data returns around the ASV, indicated by a red circle, and returns from the chase boat below the ASV.	29
<i>Figure 11:</i> Histogram of intensity values from a single LIDAR scan.....	30
<i>Figure 12:</i> Non-filtered vs filtered occupancy grid. The red circle is the vessel’s position. This is a scan with the dock the right and a small craft below Minion.....	30
<i>Figure 13:</i> Visualization of occupancy grid manipulations. From left to right 1. Original occupancy grid 2. After Gaussian blur operation 3. After dilate operation 4. After erode operation 5. Final occupancy grid after threshold	31
Figure 14: Example of <i>findContours</i> and the bounding box created	32

<i>Figure 15:</i> A red PolyForm A-3 [13] and a green Taylor Made Sur-Mark Can [14].....	34
<i>Figure 16:</i> MVG probability, for number of rings (r) and number of hits (n). Red are short round buoys and blue are the tall buoys.....	38
<i>Figure 17:</i> Parzen discriminant function output, for number of rings (r) and number of hits (n). Red are short round buoys and blue are the tall buoys. The tall peak is related to the number of samples at taken at that point.....	38
<i>Figure 18:</i> MVG probability for number of rings (r) and range (d). Red are short round buoys and blue are the tall buoys.....	39
<i>Figure 19:</i> Parzen discriminant function output for number of rings (r) and range (d). Red are short round buoys and blue are the tall buoys.....	39
<i>Figure 20:</i> MVG probability for range (d) vs number of hits (n). Red are short round buoys and blue are the tall buoys.....	40
<i>Figure 21:</i> Parzen discriminant function output for range (d) vs number of hits (n). Red are short round buoys and blue are the tall buoys.....	40
<i>Figure 22:</i> Instantaneous scan of three classes of buoy. Green (Taylor Made), Pink (PolyForm A-3) and Blue (PolyForm A-0). The cyan circle is the ASV, the pink ring is an interior exclusion zone, and the light green ring represents the maximum scan area..	41
<i>Figure 23:</i> Plot of three classes of buoys stored in mapper. Green (Taylor Made), Pink (PolyForm A-3), Blue (PolyForm A-0) and Red (N/A). The cyan circle is the ASV, the pink ring is an interior exclusion zone, and the light green ring represents the maximum scan area.....	42
<i>Figure 24:</i> Example of the two points being created on either side of an object.....	44

<i>Figure 25:</i> The left side shows pitching without rolling, and the right two graphs show pitching without rolling.....	47
<i>Figure 26:</i> Linear acceleration is shown to not affect the Kalman filtered estimate of pitch.....	48
<i>Figure 27:</i> Linear acceleration is shown to not affect the Kalman filtered estimate of roll.	49
<i>Figure 28:</i> Pitch all	50
<i>Figure 29:</i> Roll all.....	50
<i>Figure 30:</i> Static Velodyne before (left) and after corrections (right). Both are a side view of a lab. The red points are the ground.	51
<i>Figure 31:</i> Rotated Velodyne, without corrections on the left and with corrections on the right.	51
<i>Figure 32:</i> Objects are observed by the Velodyne at a greater range than it is able to classify, then classified as they come into the theoretical maximum range. Taylor Made Sur-Can (green), PolyForm A-3 (pink), PolyForm A-0 (blue) and Unknown (red).....	53
<i>Figure 33:</i> From top left clockwise: Right camera full image. ROI of red buoy found in the right camera. ROI of green buoy found in the left camera. Left camera raw image. .	56
<i>Figure 34:</i> The light tower sequence as observed through the full camera image on the left and the ROI's on the right. The smaller ROI image surrounded the panel, to decrease the size of the image and help the color classifier by reducing the amount of background noise.	58

Chapter I

Introduction

Every unmanned surface vessel (USV) or autonomous surface vessel (ASV) has a sensor suite on board tailored to the needs of its mission and environment. These sensors either transmit data back to a “ground station” to be interpreted by a human operator, or to a computer, where pre-coded algorithms make decisions based on the incoming data. In either case a software paradigm is selected deriving from the robotic primitives Sense, Plan, and Act. USVs will Sense, relay the gathered information to an operator who Plans and sends Act commands back to the vessel. Humans are very quick to interpret well-formed data, but ASVs can interpret complicated data much more efficiently. The key to moving from USVs to ASVs is the ability to sense and classify the information. Creating a sensor suite allowing USVs to quickly and correctly interpret data to act or relay interpretations to an operator is desirable in order to improve efficiency and reduce the cognitive load on human operators.

There are currently no commercial ASVs capable of completing all maritime navigational challenges. There are, however, many USVs being used to perform dull, dirty, and dangerous tasks. Removing humans from these situations is of particular interest to the Department of Defense where USVs and ASVs are in development for multiple applications including Mine Counter-Measure (MCM), harbor patrol, automated fleet protection and long endurance surveillance. With the appropriate sensor suite, an ASV can sweep vast areas day and night, searching for mine like objects or suspicious vessels with minimal supervision. For such a task, a USV/ASV has multiple types of

cameras, a GPS and usually a surface RADAR [1] to track and occasionally autonomously classify other vessels using machine learning.

In the commercial sector, shipping and oil and gas companies are driving the technology to create ASVs. Shipping companies want to remove crew to open more space for goods, and reduce manual control to mitigate the threat of piracy. Oil and gas companies are interested in inspecting pipelines and searching the seafloor for potential drilling locations. USVs and ASVs for both of these industries work closely around docks, drilling platforms, and harbor areas but current sensor payloads usually only involve a camera and GPS. Such limited sensing does not allow an ASV to make intelligent decisions, and puts both ASVs and USVs at greater risk.

To be effective, ASVs need to be able to perform the same tasks as a crewed vessel, with similar or better speed, efficiency and accuracy. Much like the challenges facing driverless cars today, it is not good enough to just know where you are, but also that there are objects around you, and that these objects have meanings. A buoy, for instance, can have meanings from lane keeping, to marking underwater obstacles to speed markers, and the vessel must also be able to interpret what it sees. Current USVs and ASVs do not have the ability to comprehend what they sense beyond whether it is static or dynamic and where it is. With the limited number of sensors, these vessels are cheaper than what they could be but they are also slower, less adaptable and therefore require more supervision. Since more operators are needed, ASVs are limited to simpler tasks.

Developing a sensor suite that can quickly and accurately decipher raw data for decision making is key to making ASVs and USVs much more effective. As there are advantages to using some sensors over others in maritime environments this paper will

present one sensor suite and robust classifying techniques that can improve an USV/ASV's ability to Plan and Act. This will be done by organizing 3D LIDAR sensed objects, collecting feature information about each one and running the feature vector through a classifier. The results are published to a global map, and subscribed to by the cameras. An algorithm creates regions of interest (ROI) around the LIDAR objects in view, and collects more data on the object of interest which is then published back to a global map

This thesis will focus on the development of an exteroceptive sensor suite for unmanned surface vessels which autonomously classifies navigational markers in real-time. The methods developed were implemented and tested on a vessel called Minion, developed at Embry-Riddle Aeronautical University (ERAU) for the Association for Unmanned Vehicle Systems International (AUVSI) Foundation's Maritime RobotX Challenge (MRC). The sensor suite was also implemented for testing on FloatingPoint, ERAU's entry into the 2014 AUVSI Foundation's RoboBoat competition, in Virginia.



Figure 1: Minion ASV, during a test in the Halifax river in Daytona Beach, FL.

The MRC competition field in Singapore was broken into multiple segments or challenges designed to not only increase in complexity but also evaluate specific areas of an autonomous platform's capability. These challenges range from channel navigation, to light sequence recognition, and to a gated obstacle course. [2]

The first task of the MRC had the Minion platform autonomously navigate to a pair of 10 meter wide buoys ("gates") separated by an unknown distance, shown in *Figure 2*. Minion then navigated a linear course between the starting and ending pair of gates, requiring the team to highlight the degree of navigation, control and repeatability inherent in the platform, and was required before any other aspect of the MRC could be attempted. [2]

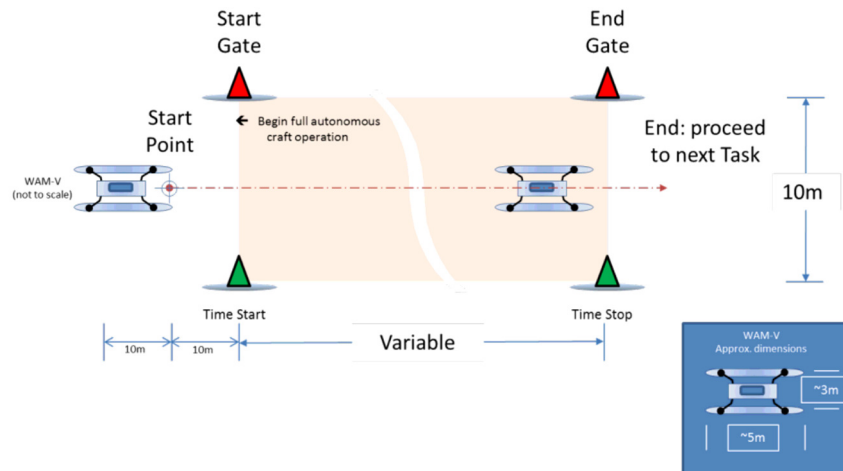


Figure 2: Task one (navigation) at the 2014 MRC. [3]

The Craft Docking and Target Identification Task, required Minion to successfully identify one of three marked docking bays using the provided signage designated by the judges before the competition run for that day. Once the docking bay

had been located, the vessel must maneuver to enter the correct dock, come to a stop, and leave the dock before moving on to the next task. [2]

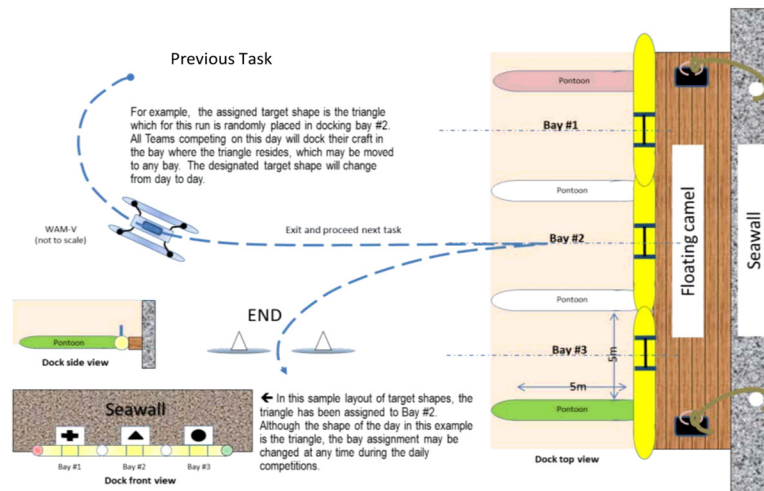


Figure 3: Three 5m bays are identified by three different signs in task three. [3]

In the Detection and Avoidance of Obstacles Task, , Minion must autonomously navigate to the pre-designated entry gate (1, 2, or 3), travel autonomously through a field of floating, stationary obstacle buoys varying in size and color. Completion of this task requires successful traversal of the obstacle field and exit through the designated exit gate (X, Y, or Z) without contacting any of the buoys. [2]

Figure 4: Task five, detection and avoidance. [3]

Chapter II

Review of the Relevant Literature

Classification in a maritime domain has been done with multiple sensors and techniques. Most literature on maritime classification is interested in determining the class of ships. Cameras, LIDAR and RADAR are the most common sensors used. There is no published work on autonomously classifying navigational markers. However the ship identification algorithms can be applied to many maritime scenarios and objects. Each algorithm has advantages, but few work well for 3D object classification in real-time. To improve the quality of classification, sensor fusion has been employed on cars in multiple forms aiding in traffic and pedestrian identification and avoidance.

Using a high fidelity 3D LIDAR, [4], was able to compress the 3D point clouds of ships into a 2D grey scale matrix with intensity representing the height of points on the objects. Displaying the grey scale matrix creates an image of the targets. Using a BP neural network algorithm and Support Vector Machine (SVM) algorithm [4] was able to achieve greater than 95% accuracy on real targets of interest. However, this process is computationally intensive and requires detailed 3D models of the targets to be accurate, which is not generally available for all maritime objects. For these reasons it is unsuitable for real-time classification.

Ref. [5] uses a 2D LIDAR on an ASV for harbor and port protection. The authors were able to recognize that objects were near then, but with limited data, were not able to classify them. This meant that the ASV needed to have predefined actions for any object it encountered, and was not able to adapt or make decisions on its own. Ref. [5] also found the LIDAR was very limited in its range and resolution, and they were therefore

only able to classify objects as dynamic and static. The information was able to be run in real-time allowing for good obstacle avoidance in dynamic environments.

In [6], infrared cameras were used to discover small ships, irrelevant objects and clutter. With a three part support vector machine (SVM) algorithm, they were able to classify the small ships with 97% accuracy. They used 18 features, which they discovered had the greatest separation, over the two stages of the SVM, extracting each feature line by line or pixel by pixel. However, this process is very slow and means that the algorithm is only good for very low resolution images, or in post processing.

An earlier form of naval classification was demonstrated by [7]. Narrow beam RADAR was swept across targets to gain enough size information to guess what type of ship it was. These techniques are also slow, and can easily be deterred by constructive and destructive interference, reflections from the water, and the surface shape of the target. The authors were able to produce decent classification with limited data sets but acknowledged that similar size ships, even with different shapes would most likely fool the classification process, which relied on neural networks.

Ref. [8] researched a sensorial-cooperative system to detect, track and classify entities for intelligent vehicles, using a LIDAR and a monocular camera to detect and classify pedestrians and cars. The detection and tracking was performed solely by the single plane scanning LIDAR and classification was accomplished by both the LIDAR, using a Gaussian Mixture Model classifier, and the camera, using an AdaBoost classifier. The results were combined using a Bayesian sum decision rule, creating a more reliable object classification with a combined accuracy of 82.9% with 6% false positives. The authors were able to create a real-time system to implement on a small ground vehicle.

Ref. [9] created an object classification system designed around a camera and an automotive radar. The system uses a radar to find objects in front of a car, then uses a camera to classify the objects into two classes, vehicle or “other object”. Four classifiers were explored in their research: a Multilayer In-place Learning Network, a k-Nearest Neighbor, a Support Vector Machine, and an Incremental Hierarchical Discriminant Regression. All four classifiers performed with an overall accuracy of 93% or greater. They found that Support Vector Machines ran the fastest and stored the least amount of data, k-Nearest Neighbor ran the slowest was the least accurate and stored the most data, and their Multilayer In-place Learning Network ran fast enough and was the most accurate.

In conclusion, there has been a fair amount of research into classification, and developing systems to perform information gathering. Using, cameras, RADAR and LIDAR, multiple systems have performed some type of classification, whether it was as simple as dynamic vs static, or as advanced as pedestrian identification. Multivariate Gaussian, Support Vector Machines and Neural Networks are the most common classifiers used for active ranging systems while speed, expandability and accuracy all vary depending on the application. Little to no research has gone into autonomously identifying navigational markers in a maritime environment, and for ASVs to be more proficient, the ability to quickly and accurately identify multiple markers is paramount.

Chapter III

Methodology

For the MRC, two sensing modalities were used to provide Minion with the required above water exteroceptive sensing capabilities. These modalities consisted of visual imagery using two Microsoft LifeCam Studio web cameras and active ranging via a Velodyne HDL-32E LIDAR, which gave Minion the ability to use visual discrimination of objects and accurately detect object presence and range in the ASV's coordinate system. These sensors were mounted to an anodized bent aluminum mast above Minion's deck as seen in Figure 5. Each sensor was positioned with at a specific height and angle to yield the desired field of view as discussed below. The sensor's inputs were then analyzed to discern relevant information about the competitions navigational markers and all classified objects were sent to a global mapper module used on Minion, which will be detailed in a subsequent publication.



Figure 5: Minion's exteroceptive sensor suite

The main sensor for this suite was the Velodyne and at 10 Hz uses 32 class 1 eye-safe lasers to scan ~10 degrees above and ~30 degrees below horizontal in a complete revolution around the ASV. Positioned 1.85 meters above the water and slightly behind the center of Minion, the Velodyne is able to see an unobstructed view fully around the ASV. The most downward laser fires first, followed by the interleaved firings from the lower and upper “banks” of 16 lasers, as follows: [10]

Table 1: Firing order of lasers on a Velodyne HDL-32E [10]

Firing order	DSR #	Vertical angle
1	0	-30.67
2	1	-9.33
3	2	-29.33
4	3	-8.00
5	4	-28.00
6	5	-6.66
7	6	-26.66
8	7	-5.33
9	8	-25.33
10	9	-4.00
11	10	-24.00
12	11	-2.67
13	12	-22.67
14	13	-1.33
15	14	-21.33
16	15	0.00
17	16	-20.00
18	17	1.33
19	18	-18.67
20	19	2.67
21	20	-17.33
22	21	4.00
23	22	-16.00
24	23	5.33
25	24	-14.67
26	25	6.67
27	26	-13.33
28	27	8.00
29	28	-12.00
30	29	9.33
31	30	-10.67
32	31	10.67

The interleaving firing pattern is designed to avoid potential ghosting caused primarily by retro-reflectors. [10]

Figure 6 shows the beam pattern out to 30m away from the center of the vessel. This is close to the theoretical maximum distance that the MRC markers will have

enough returns to be classified which is 35m. This is limited by the angular resolutions which is ~ 0.16 degrees horizontally and ~ 1.33 degrees vertically. In one second, the Velodyne can return up to 700,000 points, however in a maritime environment that number is greatly reduced because the 905nm wavelength is mostly absorbed by the water. Therefore, only non-water objects have returns.

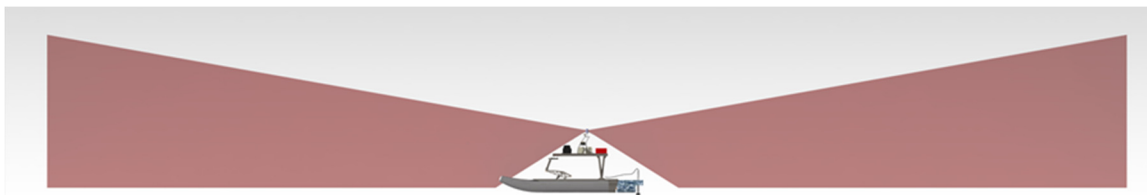


Figure 6: Velodyne 32E beam pattern section-cut out to 30m.

Figure 7 shows a top down view of the sensor's field of view out to 30 meters at the water's surface with red indicating the ground intercept of the Velodyne and green, the camera's field of view. The overlap in sensors allows them to work in support of each other and ensures there are no blind spots in a potential direction of travel.

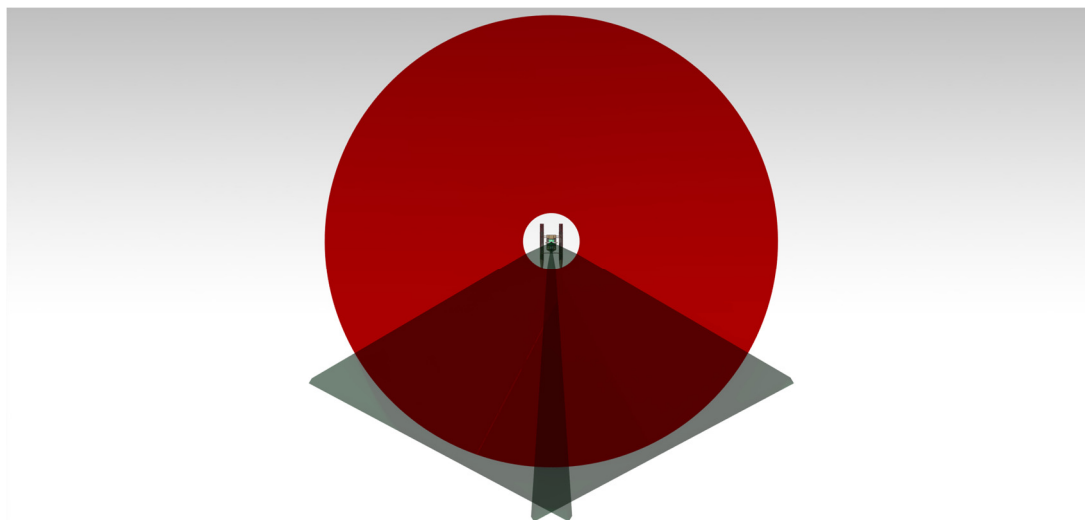


Figure 7: Top down view of sensor coverage. Red is the beam pattern for the Velodyne 32E and green represents the field of view of the Microsoft LifeCams.

Each Microsoft LifeCam camera captures 1920x1080 pixels at 30fps. Mounted 1.7m above water level, the cameras were rotated approximately 33 degrees away from Minion's center, and approximately 16 degrees down from horizontal. The result was a ten degree horizontal overlap between the two cameras directly in front of Minion and a 122 degree field of view. The bottom of the image started directly in front of the ASV and the top of the image was set to see slightly above a 2m object 40 meters away. The cameras were only sampled at 5fps to decrease the processing load. This was deemed acceptable because, even at full speed, Minion would only move 2m, or less than half a hull length, and rotate up to 36 degrees between captures.

3.1 Kalman Filter Angular Corrections for Velodyne

Every Velodyne return corresponds to a point in Minion's local frame that needs to be mapped in the global frame of reference. If Minion were constantly level, the points could be easily translated by subtracting the local position from the global position of the ASV. However, wave disturbances cause Minion to pitch and roll, the range data from the Velodyne needs to be transformed with a rotation and translation. To transform the points, the pitch and roll during the scan must be determined. For this purpose, the Velodyne has 6 internal accelerometers and 3 gyroscopes are included inside the casing shown in *Figure 8*.

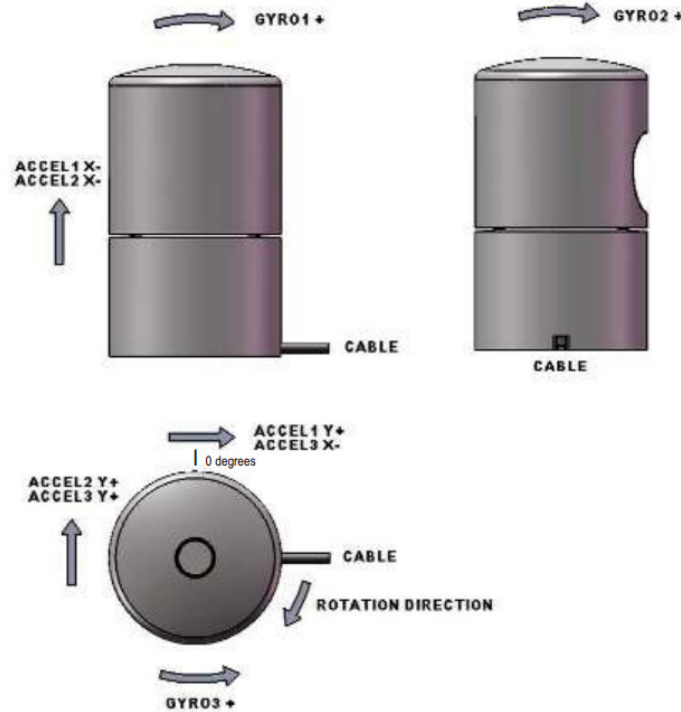


Figure 8: The Velodyne has six accelerometers and three gyroscopes oriented in line and around each axis respectively. [10]

The accelerometers, which are reasonable and accurate for a short period and work well under low acceleration conditions, can be used alone to estimate the orientation of the Velodyne. Two angle estimations can be made per axis due to the two accelerometers along each. The gyroscopes measure the rate of rotation about each axis.

Under dynamic conditions, the accelerometer and gyroscope measurements are extremely noisy. Therefore one Kalman filter for roll and another for pitch, were utilized to combine the accelerometer and gyroscope measurements to create a more stable and accurate state estimation. The procedure began by calculating two angles using just the accelerometers. This is simply done by taking the atan2 of the vertical and directional acceleration components. As Minion accelerates, the measurements will become less

accurate because the directional acceleration will change, even if there is no tilt.

Gyroscopes do not have this problem, and are therefore ideal to combine with the accelerometer estimates in a Kalman filter. These equations are found in [15]

Equation (1) predicts the current state, $\hat{\mathbf{x}} = [\theta, \dot{\theta}, \ddot{\theta}]^T$ (angle, angular velocity, angular acceleration), of the Velodyne using the previous state and the state transition model. This is the estimation for what should have happened between the previous and current sensor readings.

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}\hat{\mathbf{x}}_{k-1|k-1} \quad (1)$$

Where:

$\hat{\mathbf{x}}_{k|k-1}$ = Current predicted state given the previous state.

\mathbf{F} = State transition model.

$\hat{\mathbf{x}}_{k-1|k-1}$ = Previous predicted state given the previous state.

Error estimate using previous error and process noise.

$$\mathbf{P}_{k|k-1} = \mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}^T + \mathbf{Q}_k \quad (2)$$

Where:

$\mathbf{P}_{k|k-1}$ = Priors error covariance matrix.

$\mathbf{P}_{k-1|k-1}$ = Previous error covariance matrix.

\mathbf{Q}_k = Process noise.

Find difference between measurements z_k and estimated priori state

$$\hat{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1} \quad (3)$$

Where:

$\hat{\mathbf{y}}_k$ = Innovation.

z_k = Measurements.

H = Observation model.

$\hat{\mathbf{x}}_{k|k-1}$ = Current predicted state.

Predict how much to trust the measurement

$$\mathbf{S}_k = \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \mathbf{R} \quad (4)$$

Where:

\mathbf{S}_k = Innovation covariance.

\mathbf{R} = Measurement covariance matrix.

Update Kalman Gain

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}^T\mathbf{S}_k^{-1} \quad (5)$$

Where:

\mathbf{K}_k = Kalman gain.

Update state

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\hat{\mathbf{y}}_k \quad (6)$$

Where:

$\hat{\mathbf{x}}_{k|k}$ = Updated current predicted state.

$\hat{\mathbf{y}}_k$ = Priors error covariance matrix.

Update error covariance

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}) \mathbf{P}_{k|k-1} \quad (7)$$

Where:

$\mathbf{P}_{k|k}$ = Updated priori error covariance matrix.

\mathbf{I} = Identity matrix.

$\mathbf{P}_{k|k-1}$ = Priori error covariance matrix.

The values of \mathbf{F} , \mathbf{R} , \mathbf{Q} and \mathbf{H} were determined based on the measurements, or found empirically, and were the same value for the roll and the pitch Kalman filters.

State transition model, \mathbf{F} , multiplied the “known” value of the angle plus the rotation rate multiplied by the time between updates minus the angular acceleration times the time between updates.

$$\mathbf{F} = \begin{bmatrix} 1 & 0.005 & -0.005 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

\mathbf{R} , the measurement covariance matrix, can be determined by recording multiple samples from each sensor in a steady state, then finding the variance in the data. The two accelerometers had low variance when in steady state. However, when the system accelerated the estimated angle would change even when there was no rotation. \mathbf{R} was tuned until sharp linear accelerations stopped effecting the filter.

$$\mathbf{R} = \begin{bmatrix} 100000 & 0 & 0 \\ 0 & 100000 & 0 \\ 0 & 0 & 0.00015 \end{bmatrix}$$

\mathbf{Q} is the process noise, which dictates how much the measurements should be trusted. The larger the values, the less trust is placed on the measurement. Here, the two accelerometer angle measurements are trusted less than the gyroscope measurement. The

values were found empirically by an iterative procedure and verifying the response of the filter.

$$Q = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.0001 \end{bmatrix}$$

H is the observation model. Any sensor that measures one of the states is indicated here. The Velodyne sensor gives two angle, one angular rate, and no angular acceleration measurements. Therefore, there are two ones in the angle column, and one in the rate column. The order in which the sensors are labeled in H is the same order as the measurement vector z .

$$H = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

The angle from the output $\hat{x}_{k|k}$ is used in a rotation matrix and multiplied by each return, to translate the each return from the Velodyne frame to the local frame. The Velodyne's frame originating at the lens, with forward opposite side the cable seen in *Figure 8* and down positive vertical. It should be noted that this equation does not account for the change in displacement of the Velodyne that would occur due to roll and pitch, as the small angular changes and short distance between the Velodyne and point of rotation are deemed to be insignificant.

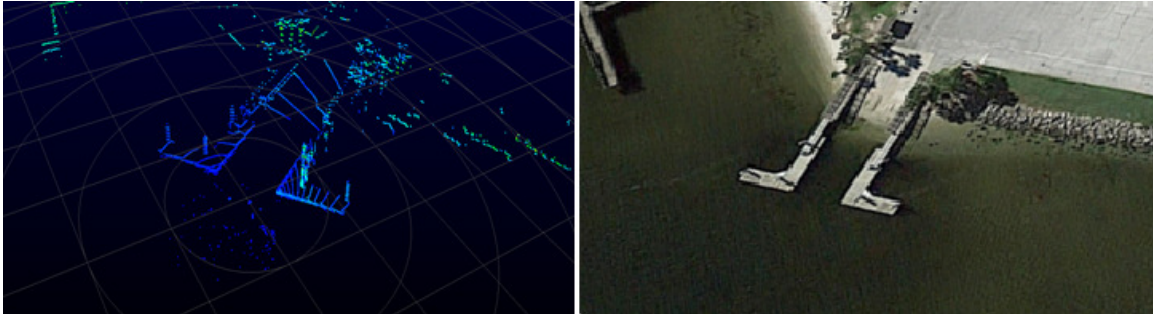


Figure 9: Raw data points of a dock from the Velodyne visualized in VeloView on the left, and a Google Maps view of the same dock on the right. [16]

3.2 Classification

As previously stated, the Velodyne can produce up to 700,000 data points per second, which creates difficulties in processing the data in real time. To help with this challenge, a novel way of storing, sorting and quickly accessing data was created by combining many common techniques. To begin, the occupancy grid and multiple arrays equal in size to the occupancy grid were created, with the occupancy grid holding return data only and the five arrays storing return information about each grid cell including number of LIDAR returns, average intensity of returns above 100, average intensity of return below 100, highest ring number and lowest ring number. The grid size was set to 0.1m x 0.1m for 30m on either side of Minion creating 361,201 cells in a 601x601 element array. The grid size can be finer or coarser without changing the featured data, depending on the needs and computing power of the ASV. An example of the generated occupancy grid can be seen in *Figure 10* where the red circle is the ASV's location, and every white pixel is a grid cell that had at least one return. The other grids were not intended for visual representation, but instead to allow for easy positional representation of collected data.



Figure 10: Initial occupancy grid showing the docks and shore on the right of the image, spurious data returns around the ASV, indicated by a red circle, and returns from the chase boat below the ASV.

The large number of returns seen close to Minion in *Figure 10* are due to white caps and the platform's wake, which are not objects of interest and therefore needed to be removed from the occupancy grid. To do this, only data points that returned an intensity value above a threshold were stored. The threshold was found by plotting a histogram of the intensity value of all returns as seen in *Figure 11*.

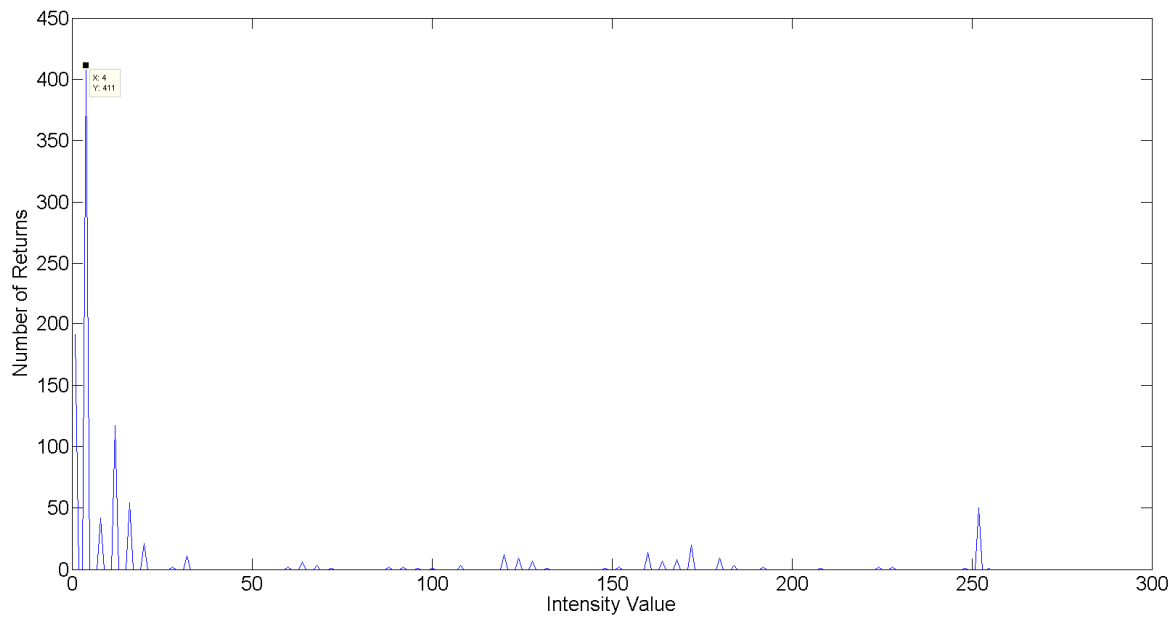


Figure 11: Histogram of intensity values from a single LIDAR scan.

Sixty percent of the one-thousand eleven returns were below an intensity value of eight. A simple threshold was applied at 8 reducing the saved values to 409. The filtered data can be seen compared to the initial occupancy grid in Figure 12.

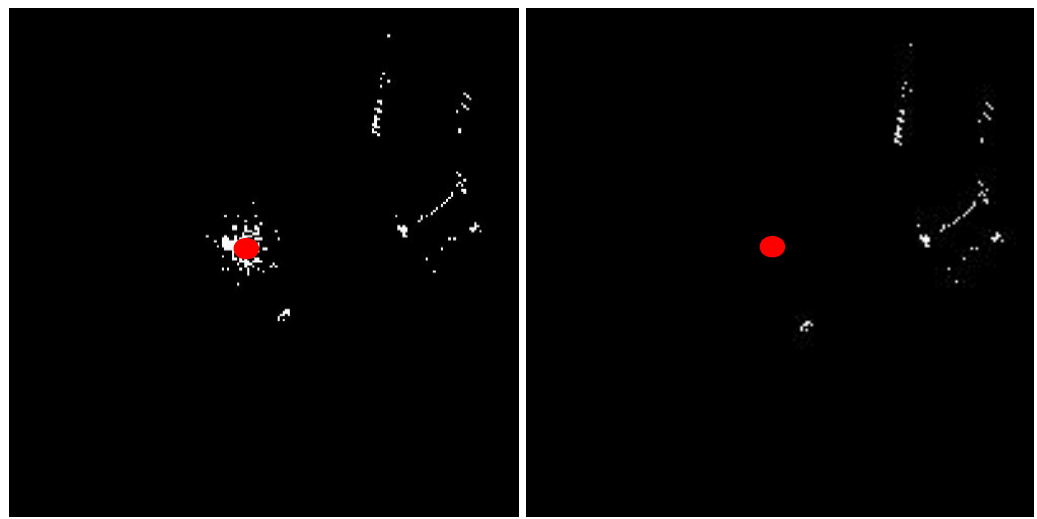


Figure 12: Non-filtered vs filtered occupancy grid. The red circle is the vessel's position. This is a scan with the dock the right and a small craft below Minion.

In order to extract objects of interest from the raw occupancy grid, the entire binary array, as shown in *Figure 12*, is treated as an image. This allowed the use of the entire and computationally efficient OpenCV image processing library for filtering objects from the raw data. The first step in this filtering process was to clean up the grid further by running a Gaussian blur (*Size(3,3)*), then a dilate (*Size(9,9)*) on the image to connect close components, and an erode (*Size(5,5)*) to eliminate elements and reduce the number of cells to check. Finally a threshold is run on the image resulting in a binary array. All remaining non-zero connected components are then said to be objects of interest, whether they are navigational markers, other vessels, docks, or the shoreline. *Figure 13* shows an example of these operations. The top right is similar to what a dock or the shore looked like, the larger point that takes up four pixels is similar to a buoy, and the lone point is spurious data. After the operations, it is desired to have the “dock” as one object, the spurious point removed and the “buoy” remaining.

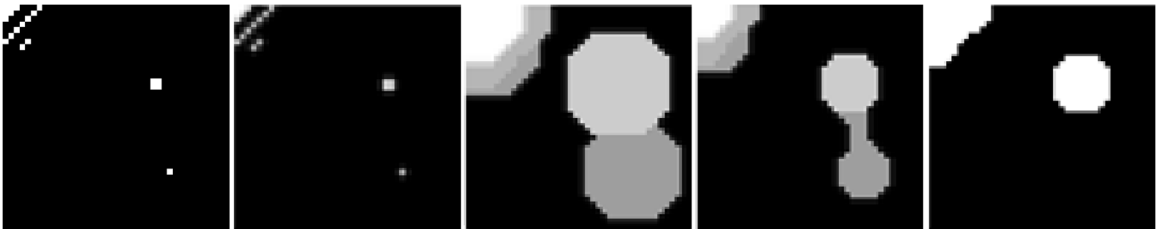


Figure 13: Visualization of occupancy grid manipulations. From left to right 1. Original occupancy grid 2. After Gaussian blur operation 3. After dilate operation 4. After erode operation 5. Final occupancy grid after threshold

To determine which, if any, of the objects of interest are navigational markers, a feature vector for each object must be created and passed through a classifier to estimate class. Using OpenCV’s *findContours*, as show in *Figure 14*, each object is separated into its connected components and using the contour bound as an array index, each feature array can be accessed quickly to pull out the features of the object. To find the markers

for MRC three features were identified as being the most distinguishing: the Range to the object d , number of returns from an object n , and number of rings that comprise the observed returns r .

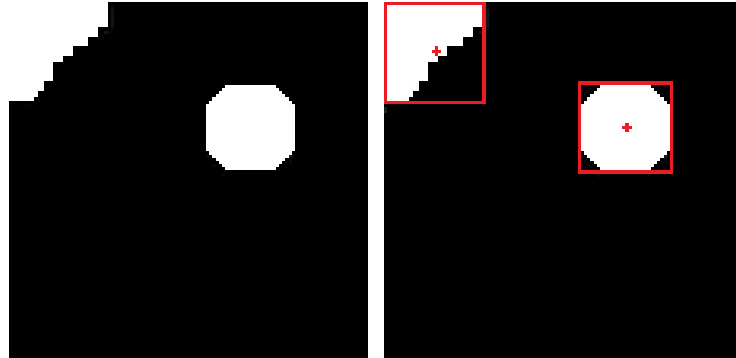


Figure 14: Example of *findContours* and the bounding box created

To understand the benefits of these features, first consider that range alone is not a distinguishing feature of navigational markers as it depends on navigation and control of the vehicle. However, range will determine the number of returns and number of rings that would be expected from the MRC navigational markers. Number of returns gives an idea of the frontal area of any object seen, which again is only true when associated with range data. The number of rings essentially gives the height of the navigational markers. Thus, when used with both range and number of hits, different shaped and sized objects, yield distinct feature sets. Additional features considered here include intensity, radius, curvature and height variance and while not used for classification due to ineffectiveness, may be beneficial for navigational markers not seen in MRC.

The three values are placed in a feature vector $v = [d \ r \ n]$ which can be used for classification. To accommodate classification of objects from both known and unknown classes, the use of probability density functions was pursued. This enabled the

use of a minimum conditional probability in order to deem the observed object as known. Two types of probabilistic classifiers were explored for this thesis, Multivariate Gaussian (MVG) and Parzen Window Estimation. MVGs are a conventional classifier that assumes a Gaussian distribution of underlying features. [11] Although range is not expected to have Gaussian distribution (more likely uniform), the remaining two features were. It was assumed that the effect of one uniform distribution would not adversely affect the results of the MVG with two strong Gaussian features, because the affect would be expected to be small enough to justify the relatively fast implementation of MVGs.

An MVG works by parameterizing each class's mean and covariance matrix; a test sample of which is then said to be from the class with the greatest estimated conditional probability $p(class|x)$. [11] The performance of an MVG decreases if it is under-trained (too few training samples) or the actual distribution does not follow an MVG. The Parzen Window Estimator, on the other hand, is a non-parametric density estimation classifier that makes no assumptions on the underlying feature distribution. It utilizes a Gaussian kernel function with a smoothing coefficient as an activation function and classifies by summing the feature vector's weighted distance from all training data. Its performance degrades if the training data is limited, or is too great. [11] The equations for the Parzen Window Estimator can be found later in this thesis.

Two classes were used for the MRC. The first was the PolyForm A-3 and the second was the Taylor Made Products Sur-Mark Can Buoy as seen in *Figure 15*. MVG and Parzen classifiers require data about a class to train or run against. Minion therefore collected data on the two buoys and completed the required setup for each classifier in Matlab. MVG requires a mean vector and covariance matrix for every class so hundreds

of recordings were analyzed to extract enough well distributed data to create these two values. Parzen requires many feature vectors and the known class for each to compare against. As stated before, if there are not enough data points, then the classification is poor. If there are too many, then the processing becomes slow. Each classifier was prototyped in Matlab, to quickly determine the accuracy and speed. From testing, it was decided that MVGs were to be further developed in OpenCV and implemented on Minion.



Figure 15: A red PolyForm A-3 [13] and a green Taylor Made Sur-Mark Can [14]

The classification procedure then executes as follows. First, every object within the occupancy grid with a radius of less than a meter has a feature vector \mathbf{x} assembled. This feature vector is sent to the two MVGs whose parameters were trained a priori. The MVG implementation found in this thesis was found in [17]. The MVG is defined by:

$$\Delta = \frac{-(\mathbf{x} - \mu)\Sigma^{-1} * (\mathbf{x} - \mu)^T}{2} \quad (8)$$

Where:

Δ = Mahalanobis distance between \mathbf{x} and μ

\mathbf{x} = Feature vector

Σ = Maximum likelihood estimate of Covariance matrix

μ = Maximum likelihood estimate of Mean vector

$$N = \frac{e^{-\Delta}}{(2\pi)^{p/2} |\Sigma|^{1/2}} \quad (9)$$

Where:

N = Output.

p = Number of features.

The covariance matrix and mean vector for each class was found from collected data of each class; 144 samples from the Taylor Made and 115 samples from the PolyForm A-3 buoys.

PolyForm A-3

$$\mu = \begin{bmatrix} 17.25 \\ 1.423 \\ 14.63 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 0.0403 & 0.2235 & 0.0037 \\ 0.2235 & 25.17 & -0.8341 \\ 0.0037 & -0.8341 & 0.0335 \end{bmatrix}$$

Taylor Made Sur-Mark Can

$$\mu = \begin{bmatrix} 15.06 \\ 3.153 \\ 18.03 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 0.1358 & 0.6599 & -0.0117 \\ 0.6599 & 12.61 & -0.8763 \\ -0.0117 & -0.8763 & 0.0794 \end{bmatrix}$$

The order of the mean vector is range, number of rings then number of returns.

Finally the output of each class's probability density function is compared to one another. The highest value over an empirically found threshold of 0.000001 was declared the winning class. If neither of the outputs exceeded the threshold, the object was declared unknown.

The Parzen Window Estimator was implemented in Matlab with 115 samples of the PolyForm A-3 and 144 samples of the Taylor Made Sur-Can. A random 60% of the samples were used for training and the other 40% for testing. The training samples were not simplified to a covariance matrix and mean vector like with the MVG and were instead stored to be compared to test feature vectors. Each training sample has the class it belonged too associated to it and the distance from the feature vector to every training sample was found. Then using equation (10) the discriminate function value, $g_j(x)$, for each class, j , was found for test feature vector x . [18] The discriminant function value $g_j(x)$ is proportional to $p(x|j)$.

$$g_j(x) = \frac{1}{m_j} \sum_{k=1}^{m_j} e^{-\frac{\|x-x_k\|}{\sigma^2}} \quad (10)$$

Where:

$g_j(\mathbf{x})$ = The conditional probability distribution

m_j = Number of test samples per class

σ = Smoothing function

The smoothing function was found experimentally by increasing it by 0.1 from 0.1 to 1 and running through all of the test samples to check the accuracy. It was determined that the best value for the smoothing function was 0.6. The class that was closest to the input feature vector was then selected as the winning class with a threshold of 0.00001 applied to distinguish when objects were unknown.

As the conditional probability distribution produced by MVGs and Parzen Window Estimation can only be plotted when using two or fewer features, *Figure 16* through *Figure 21* show the conditional probability distribution these methods produce on each pair of features. These figures clearly show that both MVGs and Parzen window estimation produce distinct decision boundaries between the two classes. As previously discussed, the MVG distributions, when compared to the Parzen window estimation distribution show the Gaussian assumption is violated due to the range feature and the discretization of number of rings r . However, it can also be seen that the decision boundaries for both implementations lie in nearly the same location. As MVGs are more computationally efficient than Parzen Window Estimation, only MVGs were implemented in real-time on Minion's hardware.

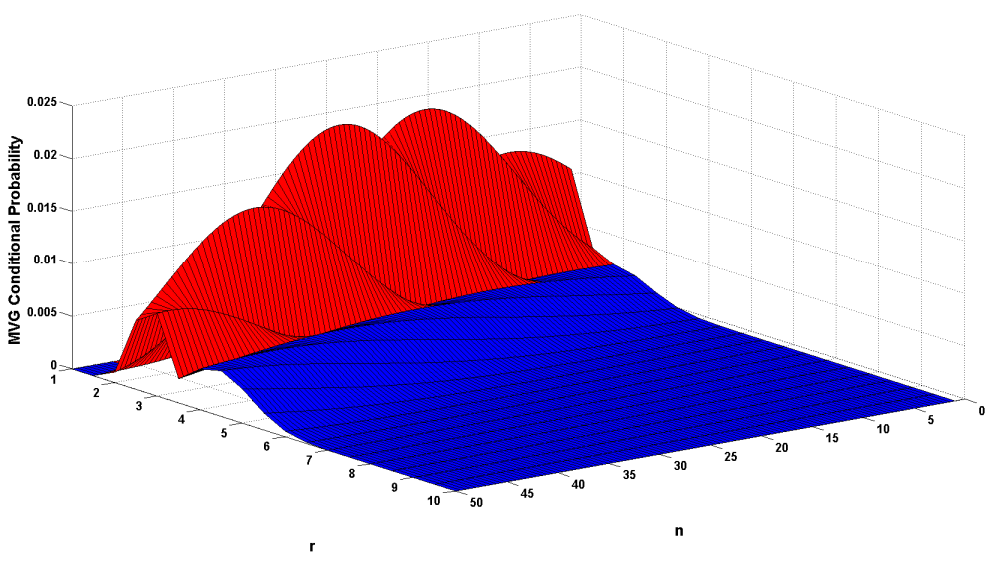


Figure 16: MVG probability, for number of rings (r) and number of hits (n). Red are short round buoys and blue are the tall buoys.

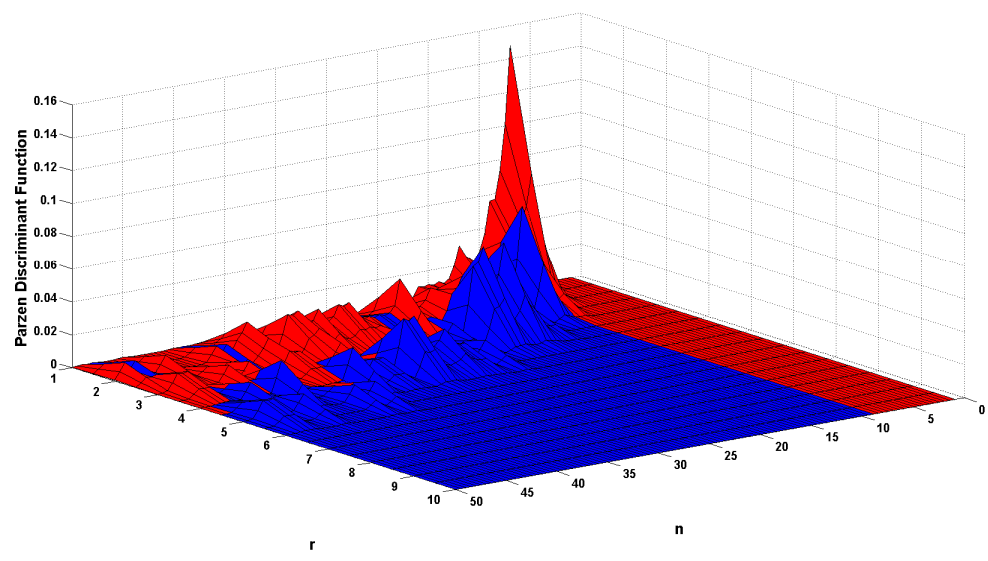


Figure 17: Parzen discriminant function output, for number of rings (r) and number of hits (n). Red are short round buoys and blue are the tall buoys. The tall peak is related to the number of samples at taken at that point.

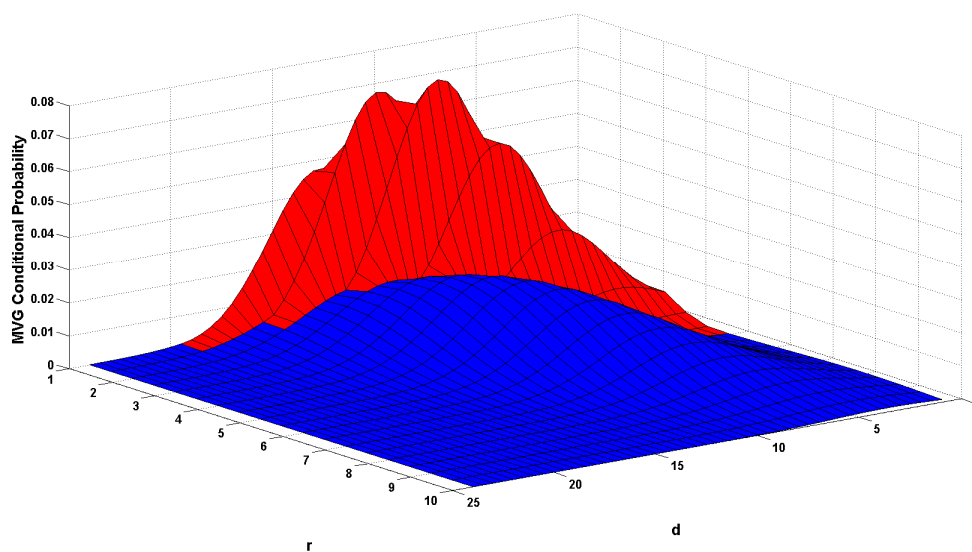


Figure 18: MVG probability for number of rings (r) and range (d). Red are short round buoys and blue are the tall buoys.

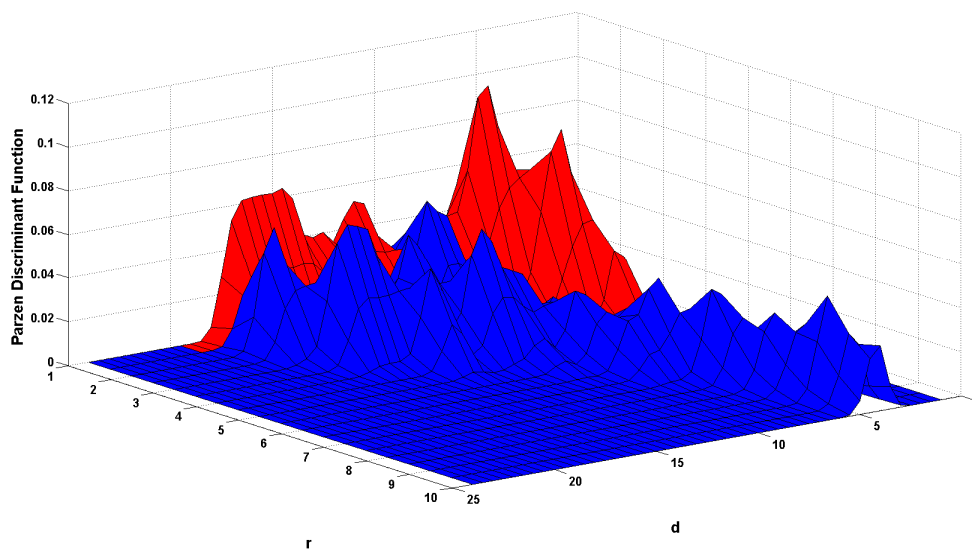


Figure 19: Parzen discriminant function output for number of rings (r) and range (d). Red are short round buoys and blue are the tall buoys.

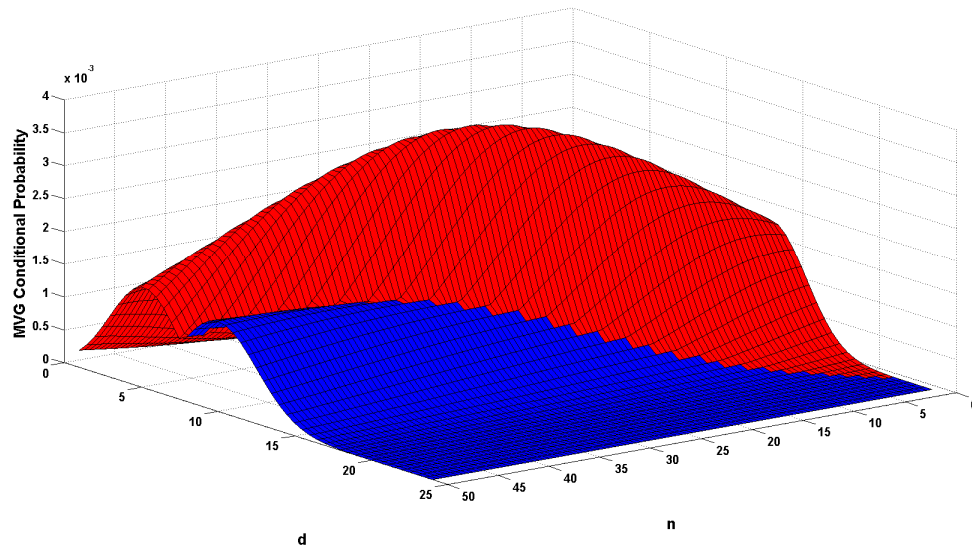


Figure 20: MVG probability for range (d) vs number of hits (n). Red are short round buoys and blue are the tall buoys.

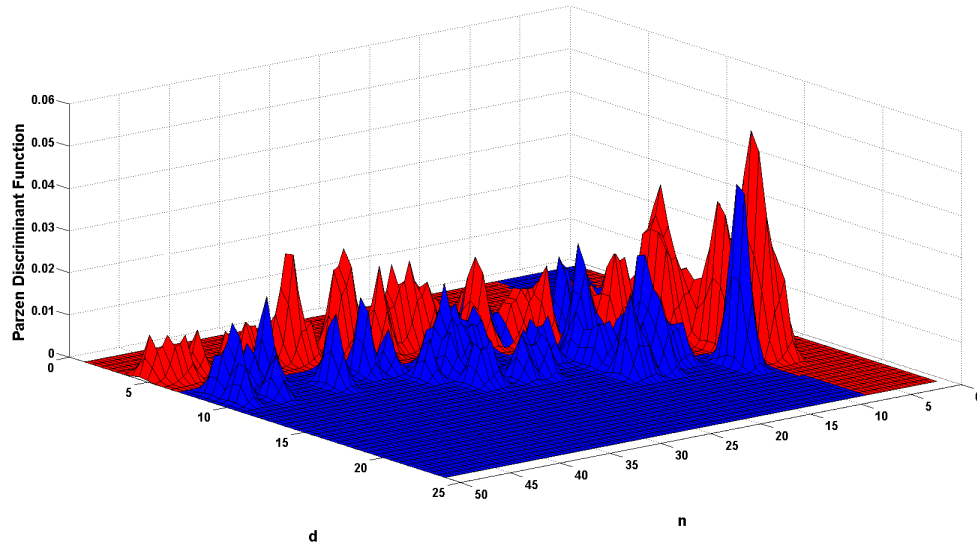


Figure 21: Parzen discriminant function output for range (d) vs number of hits (n). Red are short round buoys and blue are the tall buoys.

An instantaneous scan around an ASV will look like *Figure 22*. Each class of marker has been colored based on the classification from three classes. This scan is of the start gate and entrance to the obstacle field at the 2014 AUVSI RoboBoat competition,

which required classification of 3 classes of buoys. This particular scan had 100% accuracy.

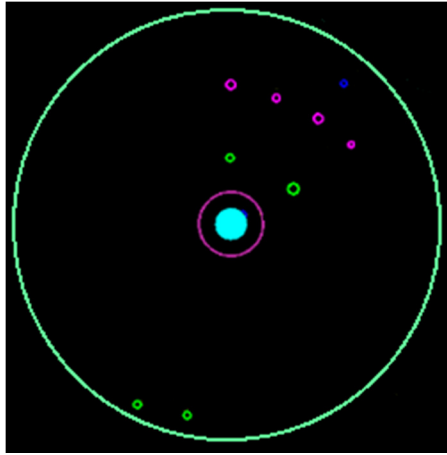


Figure 22: Instantaneous scan of three classes of buoy. Green (Taylor Made), Pink (PolyForm A-3) and Blue (PolyForm A-0). The cyan circle is the ASV, the pink ring is an interior exclusion zone, and the light green ring represents the maximum scan area.

Every classified object, whether it is a navigational marker, dock or unknown is sent to a global map. This map stores a time history of the instantaneous scans of the Velodyne classifier, cameras, and other sensors, to be used by other systems on Minion. A visual representation of the map can be seen in *Figure 23*. This is a more complete picture of the RoboBoat competition field.

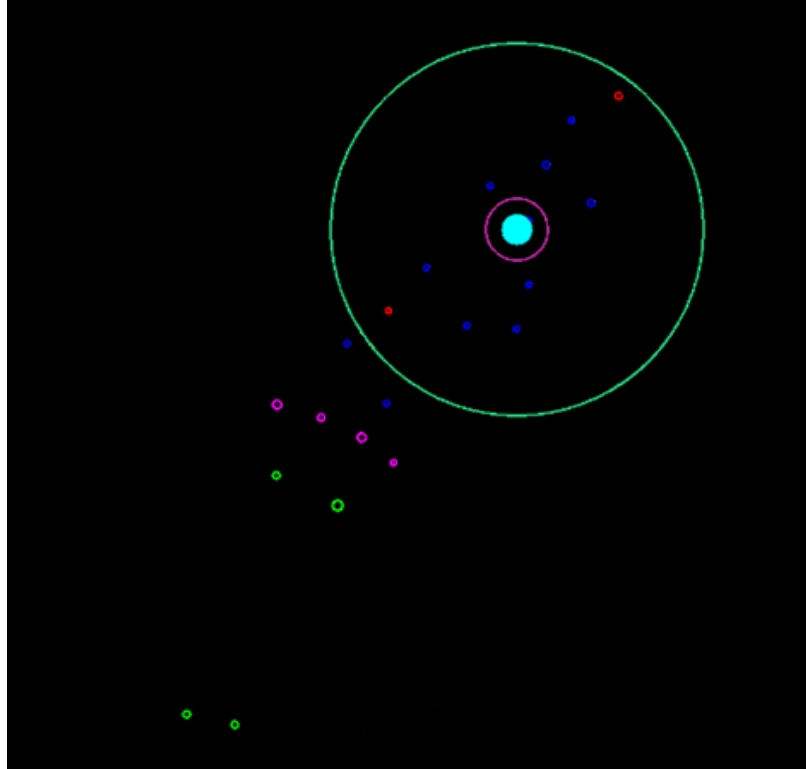


Figure 23: Plot of three classes of buoys stored in mapper. Green (Taylor Made), Pink (PolyForm A-3), Blue (PolyForm A-0) and Red (N/A). The cyan circle is the ASV, the pink ring is an interior exclusion zone, and the light green ring represents the maximum scan area.

3.3 Region of interest

There are characteristics of markers that the Velodyne does not have the capability to determine. The most important of these is Color, which determines the meaning of navigational markers. To distinguish between the green, red and white buoys for MRC, the previously described cameras are utilized. It is complicated, processing intensive, and inaccurate to use cameras to search for markers within an image using only computer vision techniques. For instance, consider that under normal circumstances a majority of an image is not of interest, but still needs to be analyzed. Minion's sensor suite uses the cameras as an enhancement instead of a critical device. Using the map

objects discovered by the Velodyne, small rectangular regions of interest (ROI) around the markers are created from the large image which are then used to determine object color.

The implementation comes from the idea that the camera has a certain horizontal and vertical field of view, and that a ray traced from the center of the camera to a point, will have an angle from the top and side of the field of view. If the pixels are evenly spaced, then it would be easy to correlate a pixel to point. Because of the limited knowledge about the cameras used, some measurements were taken to estimate the field of view, which was found to be ~66 degrees horizontal and ~33 degrees vertical. This technique does not require knowledge about the size of the sensor or focal length, just the size of the image and the field of view, making it easier to implement with any camera. Having a low distortion lens, like the Microsoft LifeCams, eliminates the need to correct for distortion at the edge of the image, although it can be implemented in this algorithm.

To find the four corner pixels to make up the rectangular ROI, it was necessary to find these points in the global frame. The mapped object's location is at the center of the object and at its base ($\mathbf{point} = [X_G, Y_G]$). To find the points on either side of the object, a vector was found from Minion's origin to the object.

$$[\Delta x, \Delta y] = [X_G, Y_G] - [X_{ASV}, Y_{ASV}] \quad (11)$$

Where:

$[\Delta x, \Delta y]$ = Vector from ASV to object.

$[X_G, Y_G]$ = Object position in the global frame.

$[X_{ASV}, Y_{ASV}]$ = ASV's position in the global frame.

The perpendicular angle to this vector is calculated. This was done by changing $[\Delta x, \Delta y]$ to polar then adding π to the polar angle to get the right side point and $-\pi$ to get the left side point. The polar points were then changed back to Cartesian points $[X_{left}, Y_{left}]$ and $[X_{right}, Y_{right}]$. Height is determined by the type of object as classified by the Velodyne. The new points are copied, and given appropriate height information. One set moves up above the maximum height of the object, and the other set moves slightly below resulting in four points. These local locations are shown below in *Figure 24*.

$$[X_{left}, Y_{left}, Z_{bottom}]$$

$$[X_{left}, Y_{left}, Z_{top}]$$

$$[X_{right}, Y_{right}, Z_{bottom}]$$

$$[X_{right}, Y_{right}, Z_{top}]$$

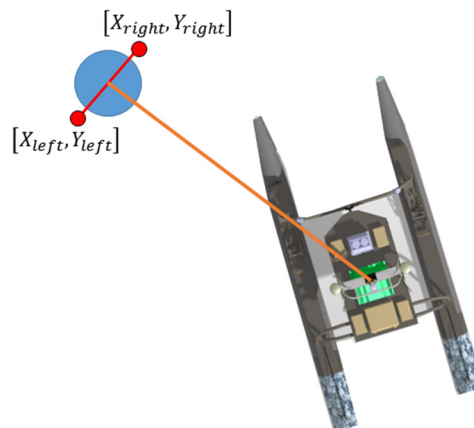


Figure 24: Example of the two points being created on either side of an object.

After the four points are determined, they need to be rotated into the ASVs frame of reference. Remember, since the $[\Delta x, \Delta y]$ vector was used to calculate the location of the four ROI points, instead of the original $[X_G, Y_G]$, the points are already translated into the local frame $[x_l, y_l, z_l]$. Therefore a rotation matrix can rotate the points individually into the local frame, using the ASV's current roll, pitch and yaw $[\alpha, \beta, \gamma]$.

$$Rot = \begin{bmatrix} \cos(\beta)\cos(\gamma) & \cos(\beta)\sin(\alpha)\sin(\gamma) - \cos(\alpha)\sin(\beta) & \sin(\beta)\sin(\alpha) + \cos(\beta) * \cos(\alpha) * \cos(\gamma) \\ \cos(\gamma)\sin(\beta) & \cos(\beta) * \cos(\alpha) + \sin(\beta)\sin(\alpha)\sin(\gamma) & \cos(\alpha)\sin(\beta)\sin(\gamma) - \cos(\beta)\sin(\alpha) \\ -\sin(\gamma) & \cos(\gamma)\sin(\alpha) & \cos(\alpha)\cos(\gamma) \end{bmatrix}$$

$$[x_{roi}, y_{roi}, z_{roi}] = Rot * [x_l, y_l, z_l] \quad (12)$$

Lastly, the points are translated from the center of the vehicle to the cameras by subtracting the displacement of the camera from the Minion's origin.

In order to associate the points with pixel locations, the angle from each edge is determined. The horizontal is simple, as it is just a subtraction of yaw angle to the left side of the camera's field of view and the yaw angle to the point. The vertical angle is determined by the range to the point.

$$vp = (atan2(h, d) - \delta)\tau \quad (13)$$

Where:

vp = vertical pixel value.

h = height of point.

d = range

δ = maximum vertical camera angle

τ = vertical pixels/radian

This process always results in the same ROI size in the global frame for the same class of objects, but different size ROIs within the image depending on the range to the object d . This technique idea was used for the Taylor Made, PolyForm and the light tower. Each marker had its own ROI size as shown in Table 2.

Table 2: Region of Interest size parameters

	Taylor Made	PolyForm	Light Tower
Z bottom	-0.2m	-0.2m	1.2m
Z top	1.3m	0.5m	1.9m
Width	0.8m	0.8m	0.9m

The light tower ROI was focused just around the panel, to aid in color matching. A simple color recognition algorithm was run on the ROIs to gather data for the mapped objects, or the light sequence. It should be noted, that the ROI process used here can be implemented with any camera as long as the targets location is known. Additionally, this method has an advantage over ground plane interpolation, because it requires less knowledge of the camera specifications.

Chapter IV

Results and Discussion

4.1 Kalman Filter Angular Corrections for Velodyne Results

The developed Kalman filter was found to be highly successful with several tests done to show its effectiveness. The first showed that the roll and pitch corrections were decoupled and was done by slowly rotating the Velodyne about one axis at a time. *Figure 25* shows a time plot of pitch, on top, and roll, on the bottom. It can be observed that as one angle changed the other stayed at approximately the same angle.

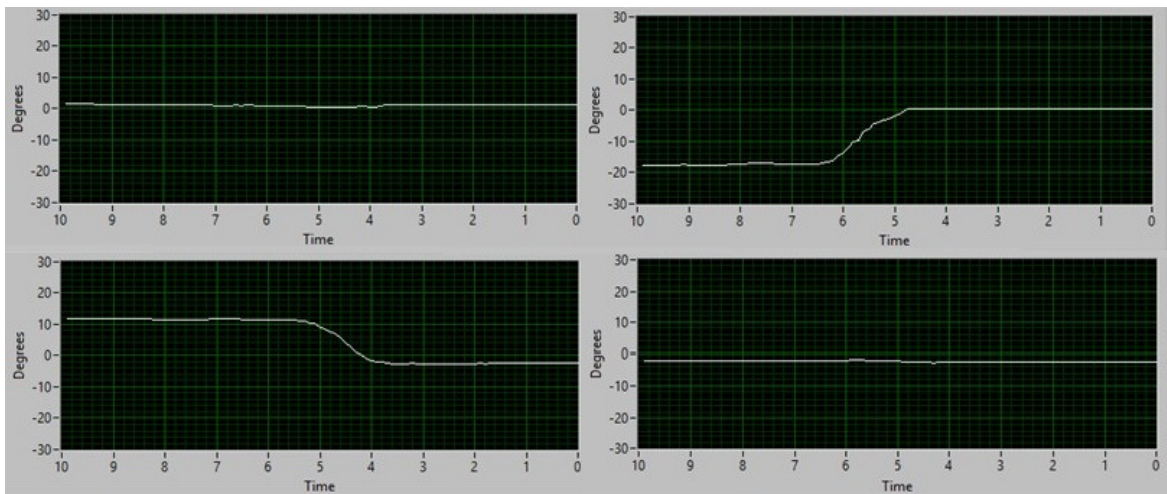


Figure 25: The left side shows pitching without rolling, and the right two graphs show pitching without rolling

The other major test made sure Q and R were tuned correctly. If they were not, then when the Velodyne was rotated sharply, or accelerated in a direction, the rotation estimates would drift (due to gyroscope integration) or over rotate (due to acceleration spikes). Acceleration spikes can be seen in *Figure 26* where the plot shows angle vs time of the accelerometer's estimate of pitch (Red) and the Kalman filter's estimate (White).

In this case the Velodyne was not rotated, but translated linearly to change the magnitude of the accelerometer output without changing the rotation rate measured by the gyroscope.

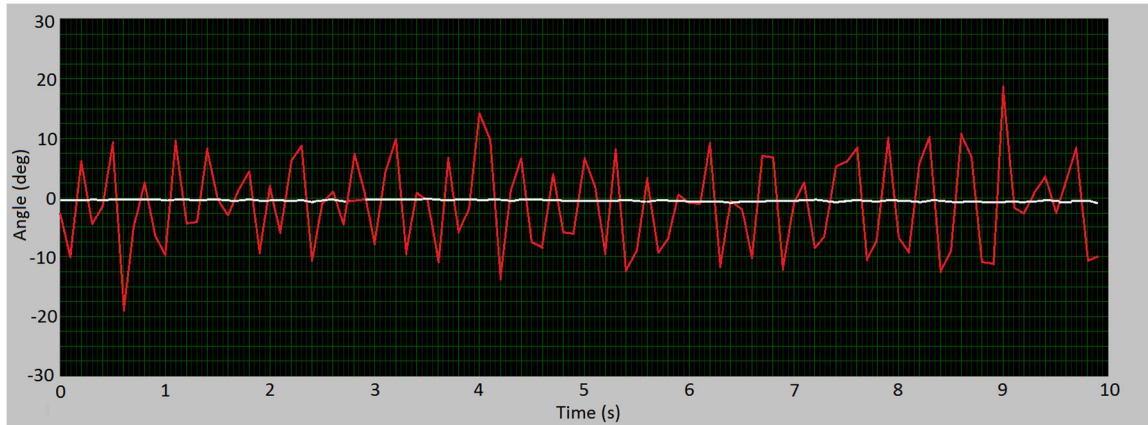


Figure 26: Linear acceleration is shown to not affect the Kalman filtered estimate of pitch.

This figure shows no appreciable effect of acceleration spikes on the rotation estimate when no rotation rate is measured. The same test was run on the roll filter by oscillating the Velodyne linearly left and right. Similar results to the pitch test can be seen in *Figure 27*. Again the Kalman filter was able to adjust for the poor rotation estimate measured by the accelerometer.

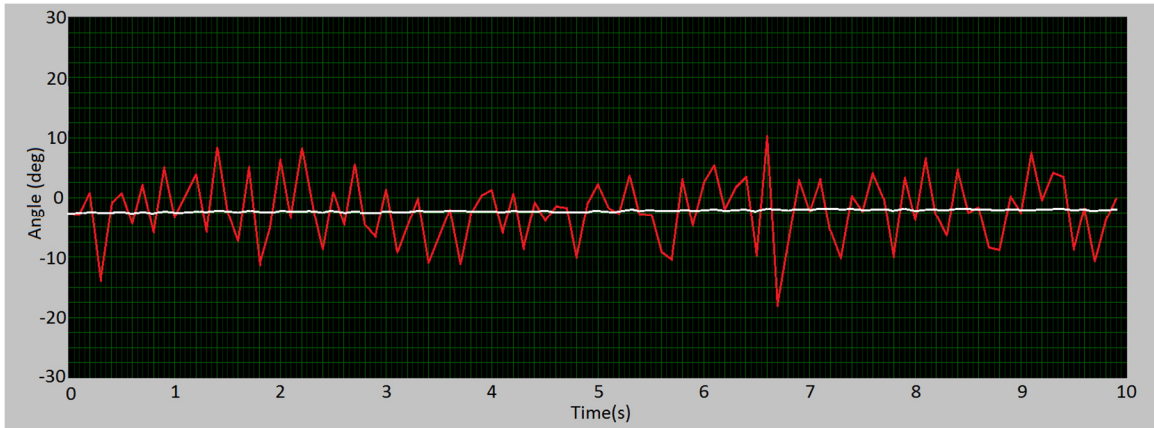


Figure 27: Linear acceleration is shown to not affect the Kalman filtered estimate of roll.

In *Figure 28* and *Figure 29* the Kalman filter was subjected to multiple steps, tracking the delay and estimate. For a full comparison of filter response and delay, multiple impulses were applied to the Velodyne in multiple directions individually. In *Figure 28* the two accelerometer angle estimates (green and red), the integrated estimate (blue) and Kalman filter estimate (white) are compared. The plots in *Figure 28* show that outside of static conditions, the accelerometers experience noise, peaks, and do not settle quickly. The gyroscope is much smoother, settles instantly, but drifts significantly. This is evident in *Figure 29*, where over the ten seconds shown in the figure, the settling point from the gyroscopes dropped 12.5 degrees. It also demonstrates why gyroscopes should not, and often cannot be used as a sole source of angle estimation.

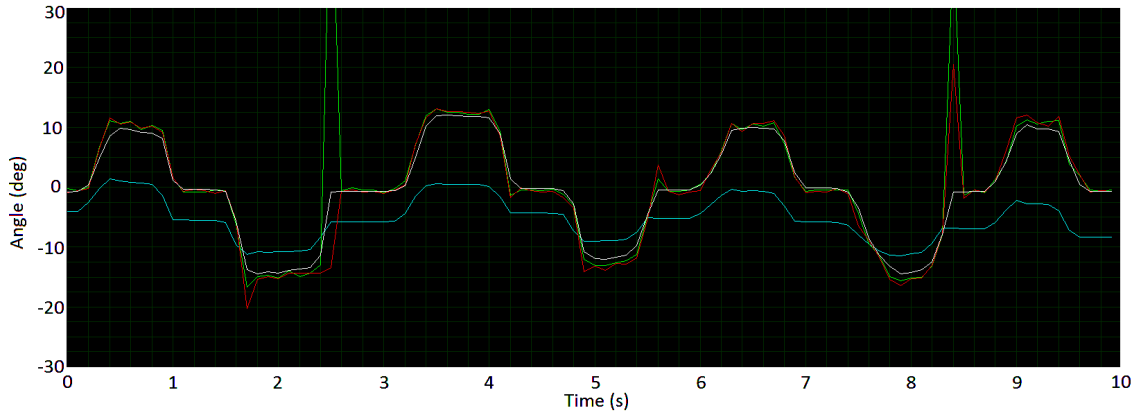


Figure 28: Step function showing the response of the accelerometer estimate (red and green) the gyroscope (blue) and the Kalman filter (white) in pitch.

By combining the two into the Kalman filter, an angular estimate that does not drift, has smooth transitions, and settles quickly was obtained as seen in *Figure 28* and *Figure 29*.

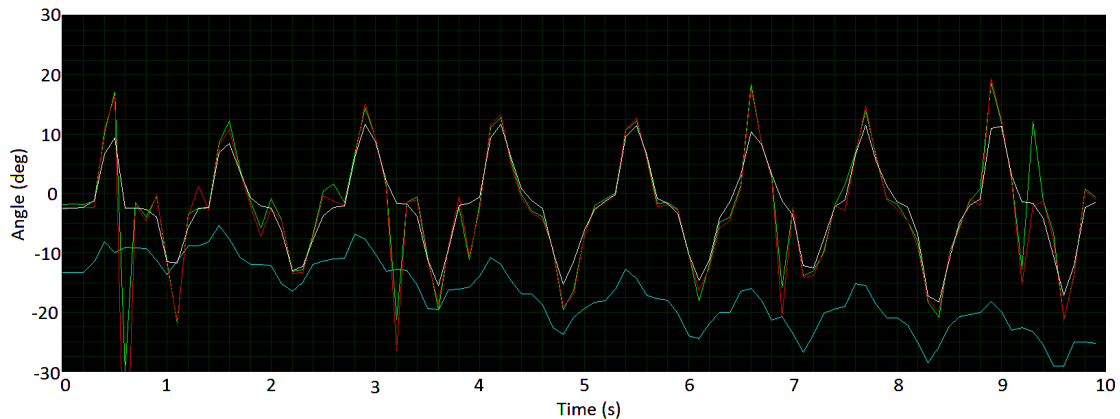


Figure 29: Step function showing the response of the accelerometer estimate (red and green) the gyroscope (blue) and the Kalman filter (white) in roll.

In a static case, the corrections had an immediate effect. *Figure 30* shows a profile view of a lab. The floor is colored red, and the ceiling is purple. The ground, in the non-corrected images show on the left side of *Figure 30* as a 3 degree down slope for the entire 15 meters length of the room.

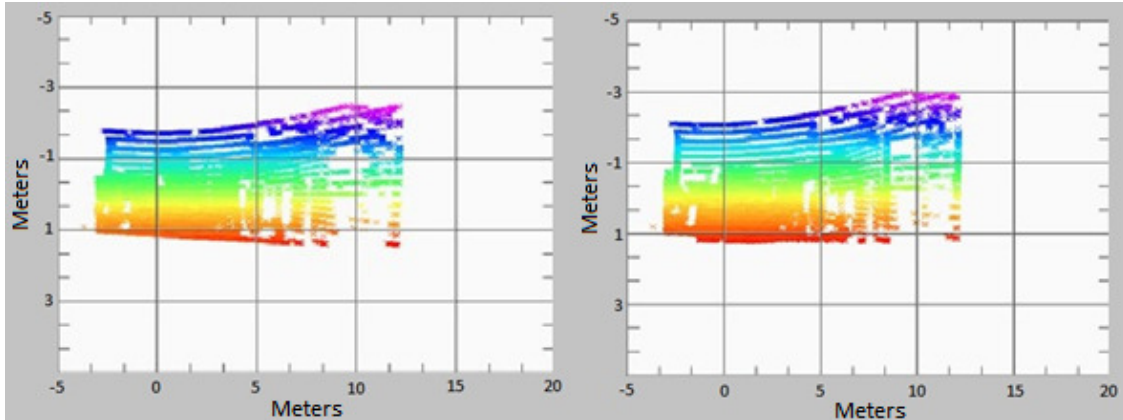


Figure 30: Static Velodyne before (left) and after corrections (right). Both are a side view of a lab. The red points are the ground.

Moreover, as the Velodyne is rotated, the floor “rotates” in the Velodyne’s frame when there are no corrections applied as seen in the left image of *Figure 31*. In the ASV’s frame, as seen on the right image in *Figure 31*, the ground is re-aligned. When compared to the static case, the ground is at a similar level and height demonstrating the repeatability and reliability of the Kalman filtered corrections. The full length of the room is not shown, because the Velodyne was rotated beyond the upper limits it is capable of observing.

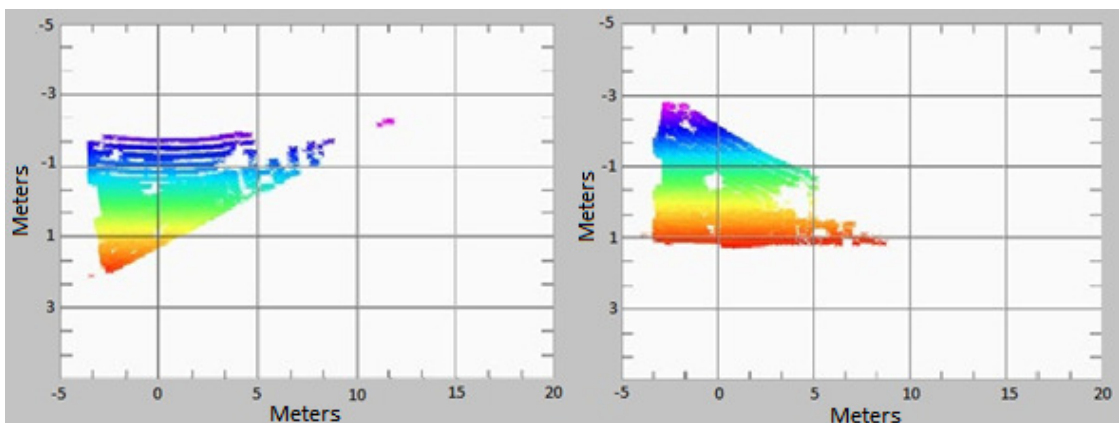


Figure 31: Rotated Velodyne, without corrections on the left and with corrections on the right.

4.2 Classification Results

Both the MVG and Parzen Window Estimation classifiers proved to be effective. The results of the accuracy tests are shown in two confusion matrixes. Table 3 shows that the MVG was consistent in picking the correct class, with a 93.84% overall accuracy and only 10.89% false positives for all tested objects.

Table 3: Multivariate Gaussian confusion matrix

		Actual		
		PolyForm A-3	Taylor Made Can	Unknown
Predicted	PolyForm A-3	100%	0.99%	2.15%
	Taylor Made Can	0	95.55%	7.75%
	Unknown	0	3.46%	90.1%

In Table 3 four hundred thirteen real samples were run against the two classifiers. Forty-five PolyForm A-3, two-hundred-two Taylor Made Sur-Can and one-hundred-sixty-six unknown Objects. For MRC, the objects seen by the Velodyne included both Taylor Made Sur-Can and PolyForm A-3 buoys, for which the classifier is trained, and only a few, unknown objects such as the chase boat and course border markings. This pushed the level of false positives even lower to 0.99%, with a true positive rate of 96.36. In fact, the 3.46% of the Taylor Made samples that were false negative were mostly due to approaching the theoretical maximum range of classification caused by objects being further out. The further out the objects move, the fewer returns received and therefore, the harder it is to distinguish between objects.

With multiple correct classifications of the objects the ASV Mapper, was able to build a high level of confidence in the objects being observed. *Figure 32* shows an

example of how, as the ASV approaches markers, they are detected by the Velodyne and quickly and accurately classified as they move further into the classification range. This is the map generated by the RoboBoat, using the MVG classifier for a different set of classes. Additionally, even in cluttered environments all classifications are completed in less than the Velodyne's scan time of 100ms.

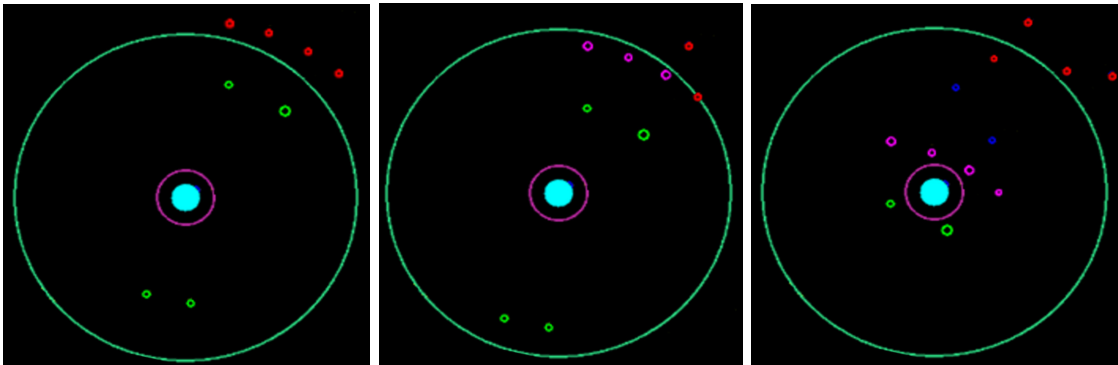


Figure 32: Objects are observed by the Velodyne at a greater range than it is able to classify, then classified as they come into the theoretical maximum range. Taylor Made Sur-Can (green), PolyForm A-3 (pink), PolyForm A-0 (blue) and Unknown (red).

Figure 32 also demonstrates how well this classifier can be expanded. At the RoboBoat competition, this algorithm was used with great success to identify three different types of buoys. Two of these, PolyForm A-3 and PolyForm A-0 are the same shape, but slightly different sizes. The MVG was able to distinguish between all three, after being trained for a different height and more classes. This ability gave the ASV detailed knowledge about the competition field, an advantage that was apparent when the ASV aptly maneuvered through the entire course on its way to a first place finish. A video of the final run can be seen at <http://www.auvsifoundation.org/competitions/roboboat/videos>. [19]

The Parzen Window Estimator was found to be highly accurate as well. With an overall accuracy of 98.2% and a false positive rate of 0.2%, this is an even more accurate way of classifying than MVG. The Taylor Made was never classified as a PolyForm A-3, and only had 1.2% false negative classifications. No PolyForm A-3 went unclassified, and the only instance was a false positive. No objects of unknown class were tested in the initial development phase.

Table 4: Parzen Window confusion matrix

		Actual		
		PolyForm A-3	Taylor Made Can	Unknown
Predicted	PolyForm A-3	99.4%	0%	N/A
	Taylor Made Can	0.2%	98.8%	N/A
	Unknown	0%	1.2%	N/A

Using the inherent Matlab classifier, the algorithm ran 166 test samples in 0.04seconds. This was a promising result, so a version similar to what would be developed for OpenCV was created, but struggled to run the same samples in 0.12 seconds. A slight enhancement in accuracy was achieved over the MVG, but was not worth the slow speed and the difficulty of the implementation.

The Parzen Window classifier never made it out of the testing phase, because of the difficulty of creating an OpenCV implementation and the slow processing rate. The MVG was chosen for real-time implementation because it was the simplest to implement, easily expandable for different classes and the least processor intensive. With over 100 hours of testing on ASVs Minion and RoboBoat's Floating Point, the robustness of this classifier has been demonstrated on multiple occasions and has been found to be highly reliable.

To increase the chance of classifying a marker, it is recommended in future implementations to place the Velodyne in a position where the number of returns on objects is maximized. This generally occurs when the Velodyne is mounted such that the center, zero angle, beam is guaranteed to strike an object. This mounting increases the chances of multiple lasers hitting an object at greater ranges, allowing for better classification. This was seen at the RoboBoat competition, when the Velodyne was placed only 0.6 meters above the water.

Research was also conducted to find the color of the buoys based on the intensity of the returns. Each point could have an intensity of 1-100, or 101-255. The lower range was a normal return, and the higher range could only come from a retro-reflector, or a surface that causes minimal scattering. With large sets of data, a separable mean value could be determined for each color of buoy (red, green and white buoys), but the variance in the readings made distinguishing between buoys extremely difficult ($\mu_{\text{red}} = 14$, $\mu_{\text{green}} = 16.8$, $\mu_{\text{white}} = 12$, $\sigma_{\text{red}} = 68.2$, $\sigma_{\text{green}} = 110$, $\sigma_{\text{white}} = 6.5$). Values above 100, did tell the classifiers that that point was from a red or green Taylor Made buoy, since they were the only competition items with retro-reflective strips. Unfortunately, this only worked when a Velodyne beam hit the small strip on the buoy, making it unreliable, especially at longer ranges. Because of this, the feature was removed from the classifier and the ROI method had to be used with the color imagery to detect buoy color.

4.3 Region of Interest Results

The concept of using ROIs to track markers was found to work extremely well. In *Figure 33* a buoy on either side of Minion is tracked tightly. When Minion moved, the

delay between the camera capture and the ASV's state output was often too large, causing the buoy ROIs to shift to one side, or in some cases not be in the ROI at all. Time syncing the cameras with the state of the vessel is therefore paramount to successfully utilizing ROIs. However, the time delay was not consistent and its length was dependent on the size of the ROIs and the number analyzed. On Minion, a delay was placed in the code before the request for state information, which worked well in the most situations.

Figure 33 shows the user interface to the ROI implementation. The full view of the right camera is shown in the top left, and the ROI generated around the red Taylor Made buoy is shown in the top right. The smaller ROI image was sent to a color identification program to be analyzed. With fewer pixels and less spurious data around the markers, the classification was faster and more accurate.



Figure 33: From top left clockwise: Right camera full image. ROI of red buoy found in the right camera. ROI of green buoy found in the left camera. Left camera raw image.

Another issue with the ROI implementation was a noticeable drift to the top of the image as the targets moved further away. This could be caused by many things, including inaccurate measurements of the maximum angle (δ), insufficient knowledge of the field of view of the camera, or image distortion. To correct for this a gain associated with range was added to the vertical pixel location equation.

Although the light tower object was not classified by the Velodyne, due to a lack of empirical data for training, an ROI was implemented around the panel. The closer the bounds of the ROI are to the sides of the panel the better color classifier results will be. The images below show a test apparatus that was used to test the ROI and light panel detection algorithms. At the MRC, Minion was able to consistently track the light panel and on multiple occasions was able to determine the sequence.

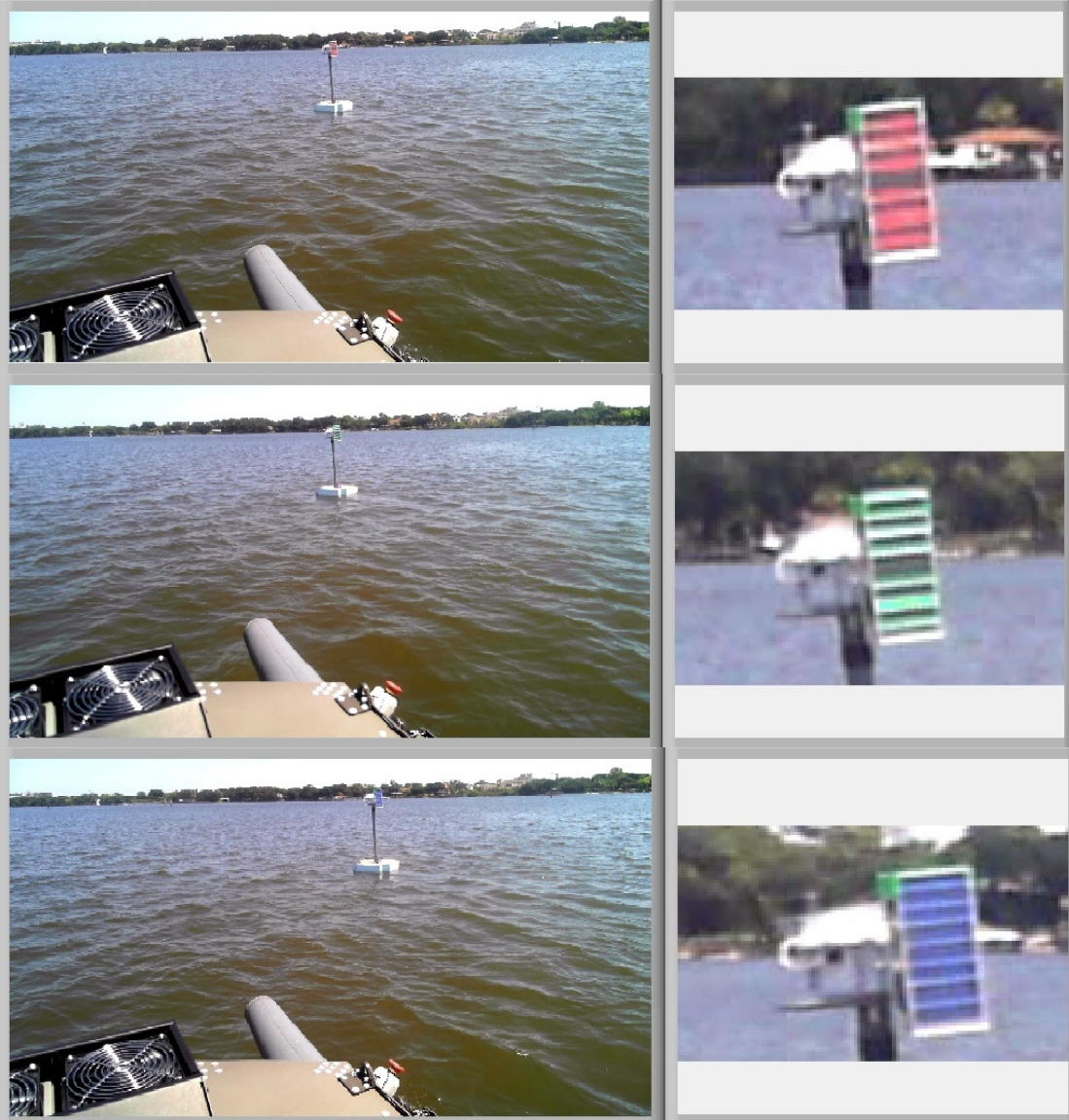


Figure 34: The light tower sequence as observed through the full camera image on the left and the ROI's on the right. The smaller ROI image surrounded the panel, to decrease the size of the image and help the color classifier by reducing the amount of background noise.

Chapter V Conclusions, and Recommendations

A highly successful sensor suite was developed for unmanned surface vessels to find navigational markers. Using and combination of a Velodyne HDL-32E and off the shelf web cameras, detailed information was able to be quickly, reliably and accurately obtained. Correcting for the disturbances in maritime environments was successfully completed using onboard accelerometers and gyroscopes and the fusion of these sensors through a Kalman filter enabled the ASV to accurately map the location of objects between multiple scans. Adjusting for the speed of the vessel within the scan and exploring other filtering techniques could improve the abilities of the sensor suite.

Classification of objects with a Multivariate Gaussian had a 95.6% accuracy with less than 1% false positives for class objects. This was surpassed for accuracy by the Parzen Window Estimator which achieved 98.8% accuracy and 0.2% false positives for the Taylor Made and PolyForm A-3 buoys classes. Unfortunately the Parzen window was not able to run in the allotted time, and never made it past prototype testing. It showed a lot of potential, and had the ability to classify more complicated objects with potentially bimodal features such as the light tower, because the classifier is non-parametric. This however would have required much larger training sets, and would have slowed the system down even more. MVG was chosen because of its reliability, accuracy, expandability, ability to add more classes with simple data collection, and speed, finishing the calculations in under 0.04s. In future works, exploring the use of 3D features could vastly improve the USV's classification ability.

With such accurate classification, the cameras were used as a support sensor to collect more detailed data for the ASV that the Velodyne did not have the capability to ascertain: primarily the color of objects. Because the Velodyne classification could be trusted, customizable regions of interest ROIs could be made to decrease the computational load.

An ASV with this sensor suite has data collection capabilities that give it reliable information to make decisions on; a vast improvement over current unmanned maritime systems. Minion and Floating Point demonstrated what kind of avoidance and decision making can that is made when more, accurate information is available.

Greater computing power will allow for the more accurate more robust Parzen Window classifier to be used. There are also many ways to combine classifiers to be more accurate. These options should and will be explored more in the future.

References

- [1] C-Sweep Multi-Role MCM USV, ASV Unmanned Marine Systems, 22, November 2014, <http://www.asvglobal.com/military-and-security/c-sweep>
- [2] C. Hockley, "The development of the ERAU Maritime RobotX Challenge Autonomous System," Association for Unmanned Vehicle Systems International, Orlando, FL, 2014
- [3] (2014, April 11) *Maritime RobotX Challenge Preliminary Rules and Tasks Descriptions (2.1 ed.)* [Online]. Available: <http://robotx.org/files/mrc-rules-and-tasks-4-11-14.pdf>
- [4] Z. Shen, "New approach of imagery generation and target recognition based on 3D LIDAR data," in *International Conf. on Electronic Measurements & Instruments*, 2009, pp. 612-616.
- [5] T.J. Pastore, "Laser Scanners for autonomous surface vessels in harbor protection: Analysis and experimental results," in *Waterside Security Conf.*, Carrara, 2010, pp 1-6
- [6] M. Teutsch, "Classification of small boats in infrared images for maritime surveillance," in *Waterside Security Conf.*, Carrara, 2010, pp 1-7
- [7] M.R. Inggs, "Ship target recognition using low resolution radar and neural networks," in *IEEE Transactions on Aerospace and Electronic Systems*, 1999, pp 386-393
- [8] C. Premebida, *A Lidar and Vision-based approach for pedestrian and vehicle detection and tracking*, [Online]. Available: http://www2.isr.uc.pt/~cpremebida/files_cp/A%20Lidar%20and%20Vision-based%20Approach%20for%20Pedestrian%20and%20Vehicle%20Detection%20and%20Tracking.pdf
- [9] Z. Ji, *Radar-Vision Fusion for Object Classification* [Online]. Available: https://www.cse.msu.edu/~jizhengp/Fusion08_Ji.pdf
- [10] (2012, November) *Velodyne HDL-32E Manual* [Online] Available: (http://velodynelidar.com/lidar/products/manual/63-9113%20HDL-32E%20manual_Rev%20E_NOV2012.pdf)
- [11] N. Thammakhoune (1999) *Long range acoustic classification*, [Online] Available: www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA390072
- [12] K. Murphy (2007) *CS340 Machine learning Gaussian classifiers*, [Online] Available: <http://www.cs.ubc.ca/~murphyk/Teaching/CS340-Fall07/gaussClassif.pdf>
- [13] "Polyform A series Buoy," Amazon.com Available: <http://www.amazon.com/Polyform-Buoy-17-Inch-Diam-23->

Inch/dp/B005N7UN02/ref=sr_1_2?ie=UTF8&qid=1418112556&sr=8-2&keywords=PolyForm+A-3

[14] “Taylor Made Products Sur-Mark Can Buoy (Green),” Amazon.com Available: http://www.amazon.com/Taylor-Made-Products-Sur-Mark-Green/dp/B000MUD4TW/ref=sr_1_1?ie=UTF8&qid=1418111181&sr=8-1&keywords=Taylor+made+sur-can

[15] L. Kleeman, *Understanding and applying Kalman filters*, [Online] Available: http://biorobotics.ri.cmu.edu/papers/sbp_papers/integrated3/kleeman_kalman_basics.pdf

[16] Photos courtesy of map.google.com

[17] R. O. Duda, “Bayesian parameter estimation: Gaussian Case,” in *Pattern Classification*, 2nd ed. New York, A Wiley-Interscience Publication, 2000, 3.4.3, pp. 95-97.

[18] R. O. Duda, “Parzen Windows,” in *Pattern Classification*, 2nd ed. New York, A Wiley-Interscience Publication, 2000, 4.3, pp. 164-168.

[19] AUVSI Foundation (2014), *2014 RoboBoat Videos*. Available: <http://www.auvsifoundation.org/competitions/roboboat/videos>