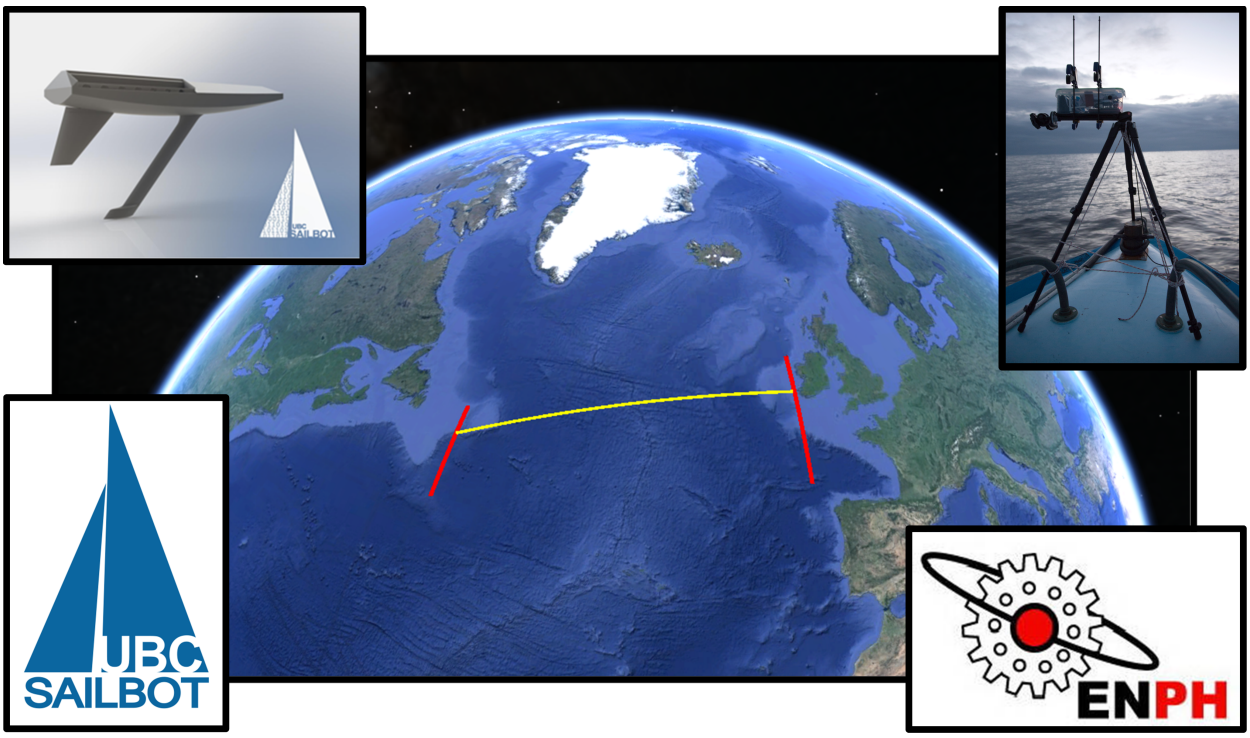
Developing A Long Wave Infrared Autonomous Marine Obstacle Detection System

Ben Hughes, Riley Doering, and Kiel Strang

Group 1507, Engineering Physics, University of British Columbia

(Project Sponsors: Kristoffer Vik Hansen, UBC Sailbot)

(Dated: April 6, 2015)



PREFACE

Our work with the Sailbot team primarily involved collaborating with:

- Paul Cernek, head of Sailbot's Obstacle Avoidance subteam
- Josh Baker, member of Sailbot's Controls and OA subteams
- Josh Andrews, head of Sailbot's Software team
- Derek Lun, member of Sailbot's OA subteam
- Arek Sredski, head of Sailbot's Control team
- Krisoffer Vik Hansen, Sailbot team Captain

While many areas of the work done on the OA system were a result of collaboration between many of the members on this project, certain aspects are more associated with some subgroups than others. Hardware development was primarily taken care of by the Engineering Physics group members, while the working obstacle detection software (the bounding boxes) was developed primarily by Sailbot members (in particular Arek, Paul, and Josh Baker). The code to establish basic Lepton functionality, video encoding, and wireless communication was jointly developed, as was the programming and integration of the IMU. Overall software architecture and pipelining was a joint effort, although this architecture is largely skeletal at the time of writing this report pending the purchase of a Quark2 camera. All testing excursions were equally attended by Eng Phys members and Sailbot members, with planning and logistics credit going to Paul.

Extra thanks are extended to Jon Mikkelsen (Sailbot's Faculty Sponsor), Jon Nakane and Bernhard Zender from the Eng Phys Project Lab for offering advice, insight, and test facilities throughout this project.

Riley Doering Kiel Strang Ben Hughes

EXECUTIVE SUMMARY

Our project is to develop an obstacle detection system for use by the Sailbot autonomous sailboat in its attempt to autonomously transit the Atlantic ocean from Newfoundland to Ireland. Our system is intended to identify hazards and provide the position of the hazards to the Sailbot's route-planning system. Our current system, still under development, uses an long-wave infrared camera (the FLIR Lepton) to identify floating obstacles.

Our team designed and assembled two waterproof and durable test rigs to evaluate the Lepton camera through on-water testing. As well as the camera, these test rigs housed battery packs, a microcomputer (Raspberry Pi 2) and an inertial motion unit (IMU; STM32F4 Discovery Board) to track the orientation of the camera. These test rigs are controlled by a Python script, developed in conjunction with the Sailbot team, which captures images, tags orientation data, and allows wireless access.

We initially planned to detect large obstacles such as container ships and fishing vessels at a range of 100m using an Automatic Identification System (AIS) receiver, but this task has been completed by other members of the Sailbot team. We focused instead on evaluating and optimizing the Lepton system. Through our tests, it became apparent that objects such as buoys and boats are detectable at close distances, less than 100m, and that salt buildup on the lens may pose a problem. As a result, we propose designs for the final camera enclosure to minimize water and salt accumulation on the lens.

The Sailbot team has met with some success in identifying floating obstacles captured on our test footage, however the low resolution of the camera makes many obstacles difficult to detect at great distance. We recommend the use of a higher-resolution camera, which has recently come within the Sailbot budget due to increased sponsorship. We furthermore expound the advantage of zinc selenide over plastic as an infrared-transparent lens. The cost of our design, with the current low-resolution camera, is less than $1000.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1.0. INTRODUCTION

1.1. Background and Significance

This project was sponsored by the UBC Sailbot team, with the goal of overcoming a significant challenge to autonomous robotic vessels in open water: collision avoidance. UBC Sailbot is competing in the Microtransat Challenge [16], which requires participants to build an autonomous sailboat capable of crossing the Atlantic Ocean. There have been nine attempts made at this challenge since its start in 2010, but none have been successful - several of these failed attempts were due to collisions [16]. Our task has been to develop a system capable of detecting obstacles and passing their location to the Sailbot control systems for route determination.

The UBC Sailbot team will make their attempt in late 2015, following the west to east route, which begins off the coast of Newfoundland and ends near the Irish coast, as shown in Figure 2. The boat must sail on its own for 74 kilometers (40 nautical miles) before crossing the start line, eliminating the possibility of towing the boat through the most debris-filled region close to shore.

The boat is a 5.5m scale-up of the 1.5m boat successfully used by the Sailbot team in shorter autonomous navigation challenges. Figure 1 shows this smaller boat. Ocean vessels typically use radar for obstacle detection, but the power and weight requirements of a radar system are prohibitive for such a small boat - our system uses an infrared camera instead. We have found a few groups working on similar systems [20], but there has been relatively little research on IR-based marine obstacle detection.

1.2. Project Objectives

The objective of this project is to create a system capable of analyzing the dynamic nautical environment around a 5.5 meter sailboat. Hazards of any shape, posing a measureable threat to the boat's integrity, must be identifiable by the module at a distance appropriate for avoiding the hazard. The module must provide sufficient time for the sailboat to steer away from any object that might damage the boat; this time varies between obstacles as follows.

Larger objects, such as boats, buoys and icebergs, are the primary challenges for the obstacle detection system. The most massive (and dangerous) of these objects should be

FIG. 1: UBC Sailbot's previous autonomous sailboat [27]

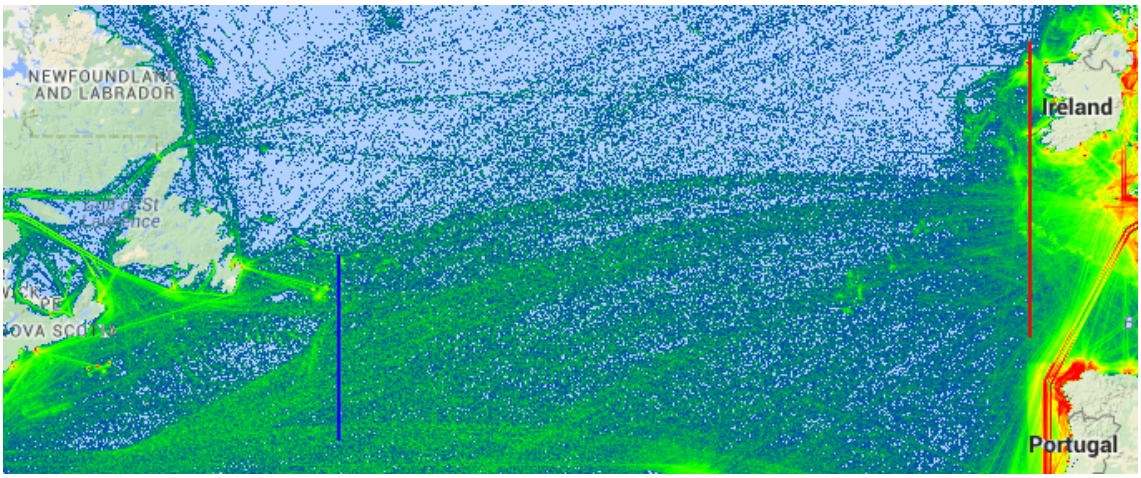


FIG. 2: Start and finish lines for the West to East route of the Microtransat challenge [17], with density map of AIS-equipped vessels. The density is low away from shore and outside major shipping lanes, but vessels are present throughout the expected route.

detected at a range greater than 100m; this range is reserved for large boats, tankers and icebergs. Small boats- in the size of private oceangoing fishing boats- and buoys should be detected at a minimum range of 50m. These ranges have been determined by examining the relative sizes and mobility of these objects and comparing that to the distance that the sailbot (at a maximum operating speed of 12 knots) would need to be sure to avoid the hazard.

Furthermore, due to power constraints, the module cannot use more than 36 W of power

at any given time, and should not use more than 2 W sustained. Structural considerations prohibit a large mass from being placed high on the mast due to the counterweight moment needed to keep the boat upright, so any parts of the detection system not located in the hull of the boat should be limited to a 0.5kg weight, while also minimzing aerodynamic losses. We also have approximately two 1 ft cubed spaces in the hull of the boat to fit our equipment.

Finally, the module must interface with the Raspberry Pi board that controls the routemaking of the autonomous sailboat to provide the relative position of detected obstacles; the avoidance and pathing system is under independent development by a Sailbot subteam.

1.3. Scope and Limitations

This report examines the Test Rig – a housing designed to keep the an infrared camera safe during testing and data collection – including the hardware and software design work required by the Test Rig. The feasibility of a low resolution infrared camera (FLIR Lepton) for the detection of floating obstacles at sea is assessed, relying on data gathered from multiple days on open and bay waters. As well, the usefulness of higher resolution infrared cameras (such as the FLIR Quark) is analysed in accompaniment with practical considerations such as an appropriate mounting point for the obstacle detection system. It is important to note that the obstacle detection system is designed to detect obstacles in front the boat, so the detection of obstacles approaching from behind or the side is not considered.

Likewise, the feasibility of various computer vision algorithms is not assessed, as the project timeline was lengthened to allow time to construct equipment for data collection and collect the data under realistic open-ocean conditions, and to obtain a higher-resolution infrared camera. Also, the mount point and camera housing for implementation on the final sailbot is considered, but the precise design has not been finalized. Finally, we do not compare the relative detectability of certain objects, as they are exceedingly rare and unlikely to be encountered. Among these obstacles are small floating debris such as logs, seaweed clumps, floating sea birds, and abandoned fishing equipment.

1.4. Organization

This report begins with an overview of previous work on marine obstacle detection, and a discussion of theoretical background information used in the project. We next describe the methods we used to evaluate infrared-vision-based obstacle detection and the Lepton camera, including our test rigs and data collection trips.

We discuss our experimental equipment, including a more detailed description of the test rig hardware and software, and the designs for the final camera mount and integration with Sailbot navigation software. We then describe the results of our data collection trips, and the performance of our obstacle identification software with the resulting data.

Finally, we describe our deliverables, ongoing commitments to UBC Sailbot, and recommendations for future work.

2.0. DISCUSSION

2.1. Theory

2.1.1. Obstacle Detection Overview

While sailors have used navigation aides for many years, only recently has there been an interest in developing autonomous vehicles that could navigate themselves around the ocean for extended periods of time. While there is a wealth of information available on object detection for autonomous surface vehicles [32] [33] [34], many of these techniques involve laser rangefinding or stereo vision in static, sterile environments. None of these techniques are easily applicable to an ocean environment, where the platform and environment are never static, waves and fog mean a constant air/water boundary has to be considered, salt poses a corrosive threat and marine life may grow on or distort the imaging medium.

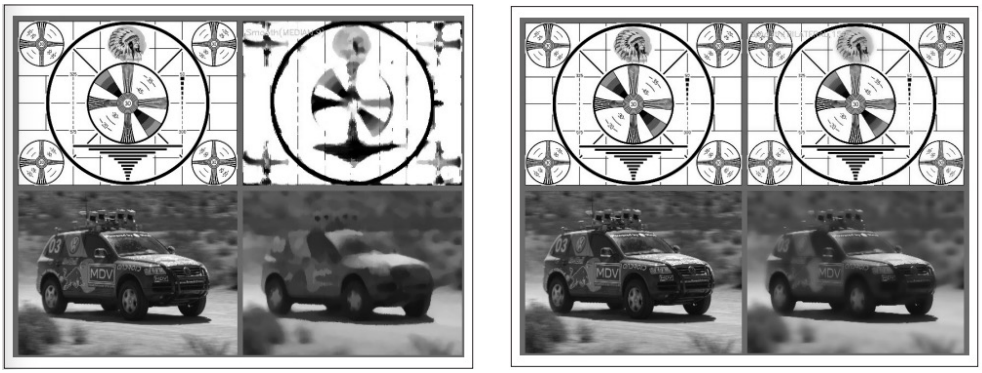
Autonomous underwater vehicles use forward looking sonar but the moving air/water medium for a boat on the surface makes sonar difficult to use when not completely submersed. While profiling sonar has been used in a research capacity on a surface boat[35] the profiling system primarily tested on a small lake with a slow-moving boat, for objects at a distance of less than 30m. This case is a far cry from the conditions that the Sailbot will experience, and furthermore cost considerations rule out sonar obstacle detection systems for us.

The most applicable research we found to our work with the Lepton infrared camera was done at the Old Dominion University in Viriginia, by their Computational Intelligence and Machine Vision Labratory. The ODU lab had developed an algorithm to count boats passing into a harbour by use of an infrared camera, primarily through the use of adaptive thresholding and training a machine learning algorithm on a large database of boat images; however, their work could be fooled on even slightly wavey water, as it was designed with harbour conditions in mind. Watching the sample video on their site shows the algorithm misidentifying water as an obstacle when there are even minor waves, let alone the 4 meter waves the Sailbot is designed to handle [20]. Despite this, ODU's work shows that IR imaging holds promise in marine obstacle detection, and much of our work follows their model.

2.1.2. Software

Acquiring a usable image is only the first step in object detection. In order to train the software to see the objects in the frame, we apply many of the basic algorithms that the human brain uses to identify and partition objects. Our eyes are developed to look for sharp lines and movement against a background; much of the work in machine vision focuses on image segmentation and object tracking, similarly.

In the ideal detection case, we would see a large boat on still water, with a clear sky. In this ideal image, the only sharp lines are those which define the boat and the horizon, at which point linefinding is easy and detection is trivial. However, the imaging conditions that Sailbot will experience are sure to be far from ideal; waves create varying angles of incidence from the sunlight, meaning a varying IR intensity across uniform-temperature water; the boats may be small and far away, and the Sailbot won't ever be still. This means that image smoothing must be performed; filters must be in place to help segment the image and discard noise. While basic smoothing can take place as simple blurring (see Figure 3a, an example from the O'Rielly OpenCV textbook on computer vision), simple techniques such as mean or median blurring (where the pixel value is set to the mean/median of those around it) is a highly lossy technique. Gaussian smoothing (or bilateral smoothing, which is Gaussian with a bias towards similarly-coloured pixels) is much better at preserving edges while smoothing low frequency noise (see Figure 3b).



(a) Blurring by taking the median of the surrounding pixels.

(b) Bilateral smoothing; applying a weighted Gaussian smoothing.

FIG. 3: Two blurring techniques to cut down noise, from the O'Rielly OpenCV textbook

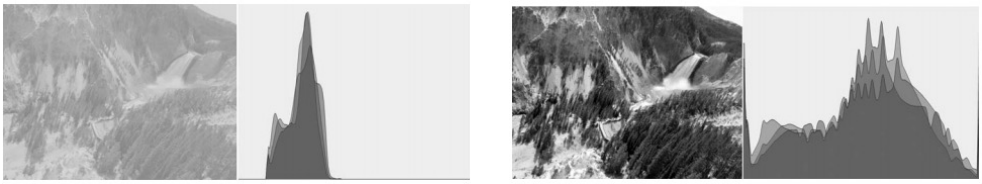
The FLIR Lepton camera we're using for this project is a very low resolution camera – only 80x60 in a 50 degree field of view – so smoothing is not really needed; each pixel

covers several square feet of most objects it's looking at, so the images look like they've been through a heavy mean filtering process already (Figure 4. However, should the Sailbot upgrade to a higher resolution camera (as is being currently considered) then smoothing algorithms will play a more important role in our postprocessing.

FIG. 4: Buoy at 50 meters, no smoothing or processing

After reducing noise, increasing contrast and thresholding help to segment the image. In infrared scenes captured during rain or through heavy fog, much of the IR light will be scattered resulting in washed-out pictures (see Figure 5 taken at midafternoon on a foggy day of testing in Ucluelet). When the obstacles are close, the contrast in the background is not a problem as the object sticks out quite obviously. However, far-off objects will have less contrast, where applying a histogram equilization helps to increase the range of pixels used. Figure 6 illustrates the effect that the histogram equilization can have; by mapping the histogram of pixel intensity values to a cumulative Gaussian distribution, the full range of pixels can be utilized. In scenes with little or no information (such as that in Figure 5) gain no net benefit from increasing contrast. (Note that Figure 5 appears darker not because the camera perceives the surroundings as colder than normal, but mainly because the autoscaling function used in the video encoding of this image takes the coldest pixel and makes it black while the hottest is white; a low contrast scene with one bright spot will appear mostly black apart from the white bright spot).

FIG. 5: Boat at 100m to 150m through fog



(a) Low contrast example. (b) Post-processing.

FIG. 6: Increasing contrast with histogram equilization, from the O'Rielly OpenCV textbook

The next step in most traditional object detection algorithms is to apply a thresholding technique: blocking off all pixels less than a certain intensity value is an easy way to prioritize close, large, easy-to-identify objects. Particularly with infrared imaging, the pixel intensity corresponding to an object is proportional to its heat; in the ocean, boats are drastically warmer than the ocean surrounding them. In addition, a ceiling threshold helps to eliminate the sun from detection as a potential obstacle, as the recorded intensity of the sun is orders of magnitude higher than those of most any terrestrial objects. Rather than apply a static threshold, as the recorded pixel values can vary by large amounts depending on the time of day, position of the sun, cloud cover, and fog, an adaptive threshold is much more useful. Adaptive thresholds are "smarter" thresholds, given that they take into account the overall

pixel intensities in the image and apply a differing threshold to each part of the image based on its surroundings. Adaptive thresholds excel at accounting for lighting differences, but in our application they can help account for sun position and waves. Figure 7 illustrates the benefit of adaptive thresholds over traditional binary thresholds.

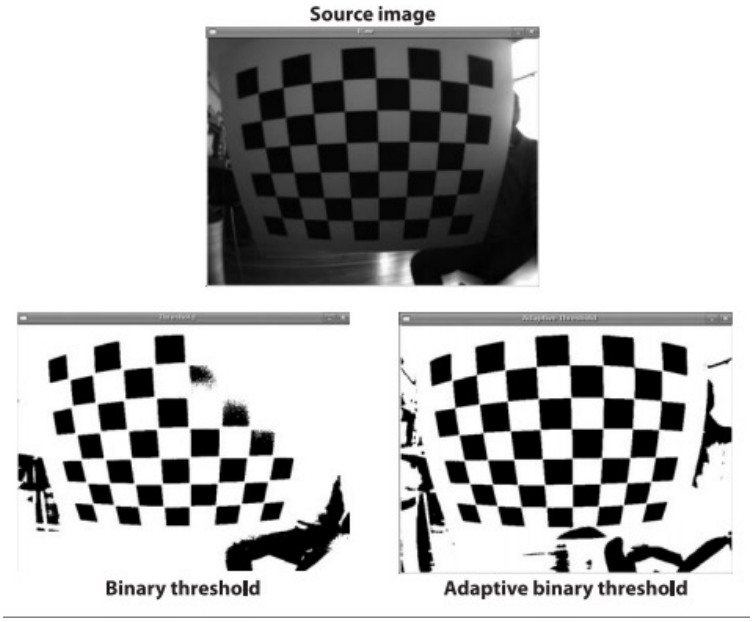


FIG. 7: Simple vs Adaptive Thresholding, from the O'Rielly OpenCV textbook

Finally, after as much extraneous information has been removed as possible, edge detection algorithms can be applied to find areas of complex geometry in the image. The most commonplace edge detection algorithm is Canny edge detection, which maps the first and second derivatives in both horizontal and vertical directions over the image before looking for local maxima and minima; by pairing these extrema with a gradient analysis and applying a threshold to filter for high rates of change, the Canny algorithm is very effective at finding edges. In the Sailbot case, we can use Canny edge detection on our filtered images to identify areas with many edges.

High concentrations of edges combined with lines at sharp angles to the horizon have a high likelihood to be an obstacle, while very rarely being triggered by ocean waves or other noise. Since obtacles prove to be persistent where noise doesn't trip the algorithm often enough, weighting an obstacle's likelihood of existence by it's frequency of appearance is a strong method to filter out false positives while limiting the chance of false negatives.

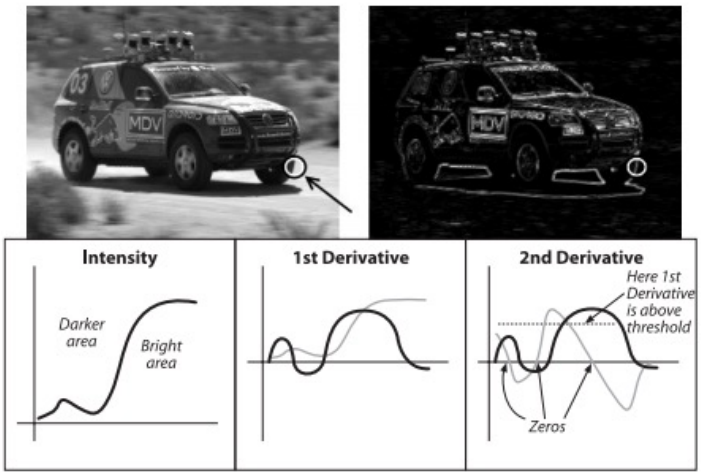


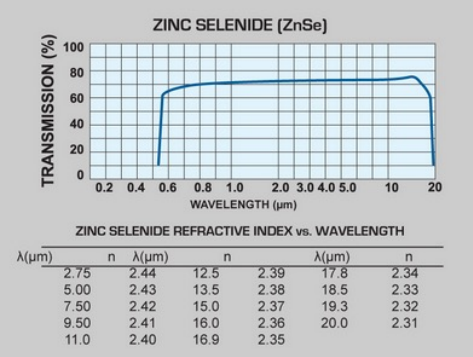
FIG. 8: Canny edge detection, from the O'Rielly OpenCV textbook

2.1.3. Hardware

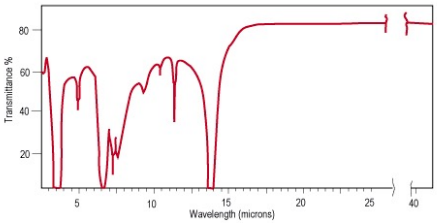
The rig containing the IR camera will be exposed to a heavily corrosive, destructive environment when travelling across the Atlantic ocean. Salt water will evaporate to leave deposits on all surfaces of the equipment, heavy waves will be assumed to crash over it on a semi-regular basis, and marine life could pose additional problems. In order to alleviate these problems, the rig must be both strong and waterproof; several theoretical considerations must be taken into account during the design.

Firstly, the lens of the camera must be strong and water resistant. The initial version of the test rig used a thin plastic film as the lens; while the HDPE film had excellent long-wave IR transparency, this is not a suitable solution for a long-term, durable build. Many materials were considered for the lens of the final rig design, with rigidity and transparency in the 7-13 micron range (the wavelength of long-wave infrared light) being the most important factors. While certain materials such as sapphire crystals are often used in long-wave infrared imaging, most are too prone to scratching or, worst of all, are water soluble. An approprite solution was found with Zinc Selenide, which has excellent, even transmission across the entire spectrum our camera operates in (see Figure 9).

Secondly, any water that sits on the lens of our camera will evaporate to leave behind salt deposits, obscuring and degrading our image quality significantly over time. While not a robust subject of research, anecdotal evidence from several boat owners tells us that within two weeks at sea, any unattended surface becomes heavily encrusted. So, any method of reducing salt deposits is highly desired.



(a) Zinc Selenide transmission spectra.



(b) Typical IR-transparent plastic spectra.

FIG. 9: Comparing transmission spectra of various materials.

Hydrophobic coatings seem to be an obvious choice; if water never stays on the lens, then it can't evaporate there. However, there are no commercial coatings built with the intent to remain IR transparent. We found several basic spray-on coatings, such as Rain-X or the like, to be minimally detrimental to the transmission spectra of our lens but these coatings have a short lifetime; they're meant to be reapplied every week or two (something clearly impossible on the Sailbot's voyage). While one of these coatings will be applied at the beginning of the voyage, it can only be assumed to be useful for a few days after launch.

A second, less beneficial choice comes in the form of a passive system. Similar to the aerodynamics of bug deflectors on flat-front vehicles, by disrupting the airflow in front of the lens in a manner so that the air flows sharply upwards in front of it, a sort of "air wall" helps to intercept airborne droplets and move them past the lens before they land [36]. As the Sailbot's expected to be operating in areas of high wind and travelling at speed, this effect may significantly reduce the amount of water droplets that land on the lens without consuming any power or adding any significant weight or complexity to the system. Wind tunnel testing is planned for our model in late April.

2.2. Methods and Testing Protocol

The performance of the Lepton infrared camera was the largest unknown when starting this project, so much of our testing methods revolved around rapidly implementing basic functionality, testing, then adding additional functionality with each testing iteration. Several large testing excursions were used to try to implement as much as possible in one integrated system.

2.2.1. Test Rig 1.0

Test Rig 1.0 was the first testing excursion of the term. The goal of this testing round was to implement basic Lepton functionality (image capture, scaling from 16-bit images to the more standard 8-bit, and video encoding) along with the communication protocols so that the Raspberry Pi in the test rig's box wirelessly backed up the data to another computer on board. The Test Rig 1.0 was then taken to Ucluelet over Reading Week (February 15-19, 2015). Brian Congdon of Subtidal Adventures, a whale-watching company in Ucluelet, took us out for close to 8 hours of total time at sea over two days, taking us on close drive-bys of bouys, logs, boats and marine life. As we approached each obstacle, the captain would read out the our distance using the onboard radar system which we would then timestamp into each recording; this lets us play back through the footage we captured and interpolate the distance to the obstacle at each point in time. Pairing this with time readouts of when our obstacle detection code first positively identifies an object, we know the approximate distance at which our code identifies various obstacles. With the test rig strapped to the front of the boat, this outing gave us a massive amount of invaluable footage, guiding the development of Test Rig 2.0 and giving insight into the final rig design. Results from this outing, as well as the others, will be shown and discussed in the Results section of this paper.

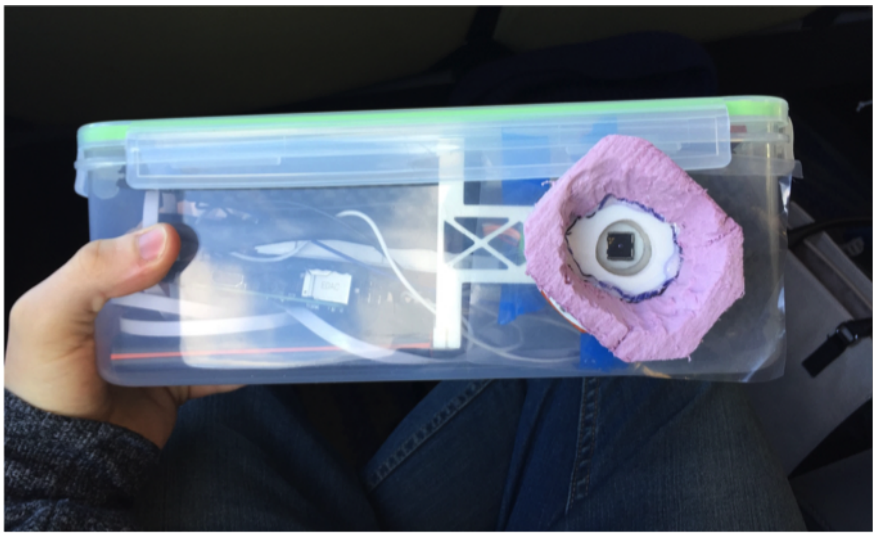
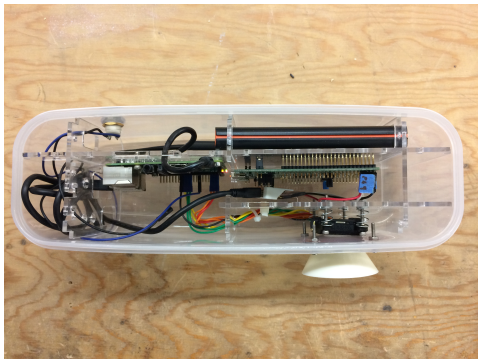


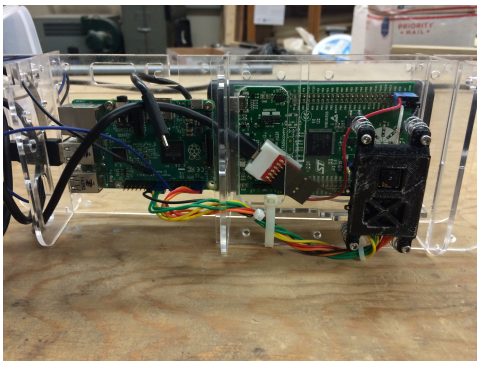
FIG. 10: Front of Test Rig 1.0

2.2.2. Test Rig 2.0 and Obstacle Detection

Running in parallel to the development of Test Rig 2.0 was the building of the overall software architecture and the obstacle detection software. Using video data gathered on the Ucluelet trip, many different filtering, thresholding, line finding and pattern recognition functions were experimented with and tested. The pipeline and initial functionality of the object detection software was developed for March 14, 2015 when the Sailbot team held their Sync-the-Boat hackathon. However, much of the object detection development was paused on March 21 when our presentation of our obstacle detection system won the top prize among oral presentations at UBC's Multidisciplinary Undergraduate Research Conference. As a direct result of this prize we acquired several new sponsorships, opening the door to the possibility of buying a higher-quality camera (such as the FLIR Quark2, the best model of which has a 640x 512 resolution). Since cameras of this quality would drastically improve our detection range and accuracy, but the detection software would need entirely different functions from those used for the Lepton, much of our software focus shifted to evaluating the capabilites of the Quark2 and the Raspberry Pi's ability to process images at higher resolutions.



(a) Top-down view.



(b) The interior structure, with the RPi, Discovery Board, and Lepton (in housing) in view.

FIG. 11: Test Rig 2.0.

The second main outing took place on Wednesday, April 1, 2015. With the Test Rig 2.0, in which we had developed more advanced Lepton functionality (in the form of controllable video scaling, more advanced correction algorithms and general bug-fixing), a solid internal structuring, and implemented an STM32 Discovery board to act as an interial measure-

ment unit (IMU). Stamping every gathered frame with a 9 degree of freedom IMU readout (accelerometer, gyroscope, and magnetometer) gave us a database to use when we implement horizon-finding and orientation tracking from IMU data; this is one of the ongoing commitments. These tests were performed on a boat taken out in North Vancouver.

2.2.3. Final Rig

The final version of the obstacle detection system, the one which will be placed on board the Sailbot, will take a drastically different physical form from the test rigs. While the test rigs held the Raspberry Pi, battery, Lepton, and IMU all in one casing, on the final rig ideally only the lepton is mounted high on a tripod at the front of the boat while the rest of the electronics are in a secure compartment in the hull. This limits the weight at the top of the tripod, reducing the induced moment and minimizing the effect on the original mechanical design specs (which did not anticipate any obstacle detection systems). In addition, the final rig must be durable and resistant to sea water damage, incorporating a solid IR-transparent window and some method to combat salt buildup. To ensure the success of this distributed design, testing already performed has demonstrated that Lepton can interface with the Raspberry Pi over long cables, up to 3 meters in length.

The final rig is, at the time of writing this report, still in the design stages. At the beginning of the term, building the final rig was a desired deliverable but this has changed as Sailbot extended the project timeline and team members committed to working with them past the early April deadline. The design concept is complete for the final rig, though building and testing will take place through April and possibly into May. The finalized software deadline has also been extended with the evaluation of the Quark2, though we can still discuss the planned testing methods for each.

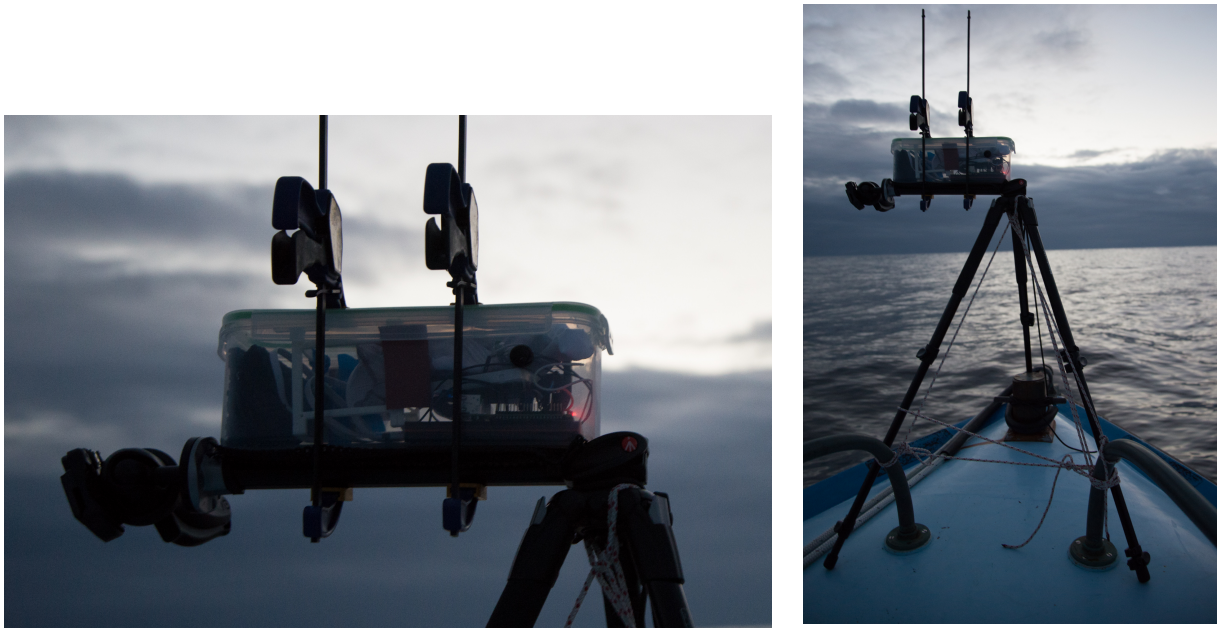
The finalized rig will contain a housing for the infrared camera (either the FLIR Lepton or Quark2), with electrical connections down to the hull of the boat. The exterior will have an extended conical lens protector to minimize splashes (already built and tested on Test Rig 2.0), an airflow shaper to add passive protection from water droplets (as discussed in the Theory section previously) as well as a loose, external windmill-like shell which can harness wind power to spin a wiper across the lens face. These components will be assembled and built mid April, with wind tunnel testing in the Rusty Hut pending for the end of April.

Moving forward with the software, integrating the IMU data with orientation tracking

and horizone finding will be tested with the data sets obtained on the April 1 outing. Once the choice of infrared camera is finalized (and purchased, if the Quark2 is chosen) then our obstacle detection software will be refined by playing our existing test data through the pipeline and optimizing for identification at longer and longer distances; this is measured by timestamping when the software first identifies the object, and optimizing the code for the shortest timestamp.

2.3. Experimental Equipment

2.3.1. Test Hardware

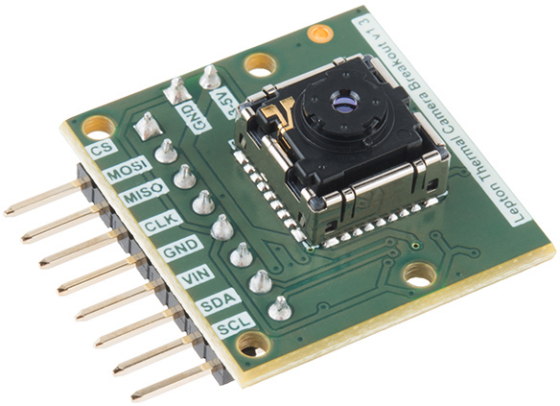


(a) Close up of Test Rig 1.0

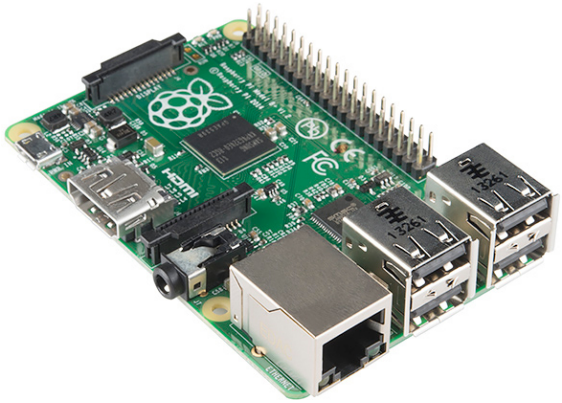
(b) Test Rig 1.0, clamped to a tripod secured to the bow of a small boat.

FIG. 12: Two views of an open-ocean test at Tofino of the FLIR Lepton, contained in Test Rig 1.0.

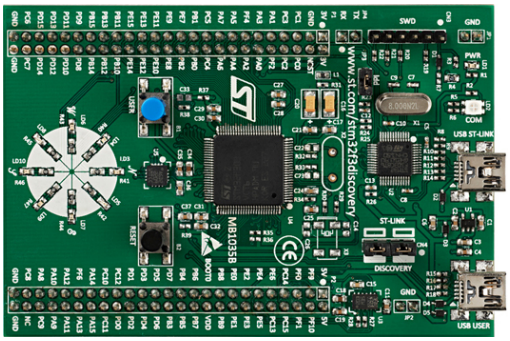
In order to evaluate the FLIR Lepton infrared camera, it was important to secure it in a waterproof housing. This housing, dubbed the Test Rig 1.0, was constructed from a large oval plastic container with a waterproof gasket-sealing lid. A single hole was drilled in the side to accommodate a button to start and stop the recording of video from the Lepton.



(a) The FLIR Lepton infrared camera (black) mounted in a green breakout board developed by Pure Engineering [22]. (Image CC BY-NC-SA 3.0 by SparkFun.)



(b) The Raspberry Pi Model B+, a microcomputer used by the Sailbot team for all control functions, as well as by our group to run the Test Rig [25]. (Image CC BY-NC-SA 3.0 by SparkFun.)



(c) The ST Discovery Board, with integrated IMU unit. (Image from ST.)

FIG. 13: The Raspberry Pi and FLIR Lepton used in the Test Rig.

Figure 12 shows Test Rig 1.0 being tested at Tofino in February. (The Test Rig interior has since been refined and restructured in version 2.0). This test included data collection in sunshine and rain, and in broad daylight, evening twilight, and early night.

As well as providing water protection, the plastic container was large enough to hold a battery pack, Raspberry Pi computer, and the Lepton camera. We cut a window in the container and covered it with infrared-transparent film to provide a viewport for the camera - this will be replaced with a more robust zinc selenide window in the complete obstacle detection system. Version 2.0 of the Test Rig features a laser-cut plastic interior support structure to organize all cables and devices, and accommodate a large inertial measurement

unit (IMU).

2.3.2. Test Software

The Test Rig is controlled by a straightforward Python script, shown in Figure 14. The script responds to button presses, starting and stopping the recording of video from the Lepton. 60 by 80 pixel frames are grabbed from the Lepton approximately 20 times per second at 16 bit depth. These high bit depth images, which high tonal gradation resolution, are then converted to an 8-bit video stream. Version 2.0 of the Test Rig hardware includes an IMU, and this data is polled and logged for each frame captured.

As information is lost in the conversion of 16-bit source images to 8-bit video, the original 8-bit images are stored alongside the video output. As well, the bit-mapping for the video is changed dynamically - the darkest and brightest regions in the video are assigned minimal and maximal values respectively in the 8-bit encoding, whether or not these regions do not take on maximal values in the 16-bit images. In effect, therefore, the video exposure is computed frame-by-frame. This maximizes the contrast ratio in the video, though this approach does not always produce the best frames for analysis.

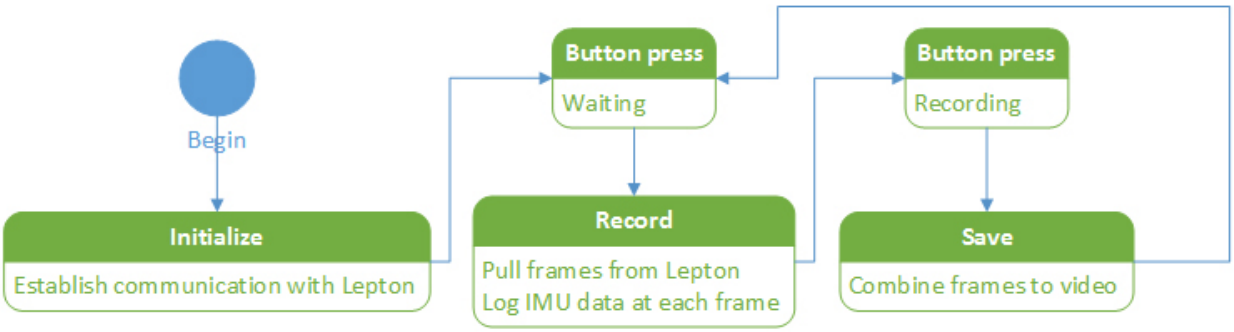


FIG. 14: The flow of the Test Rig control software.

2.3.3. Software Architecture and Routemaking Integration

a. Obstacle Detection Our software starts by identifying obstacles in a single frame of video. First, it applies Erosion and Dilation filters to reduce noise, then uses OpenCV's Canny Edge Detection to find contours. Finally, it draws bounding boxes around areas with a high density of contours - these are considered to be obstacles.

Work on this part of the system is currently on hold, as UBC Sailbot has been offered a donation of a higher-resolution infrared camera, the FLIR Quark. This will significantly improve our detection capabilities, but is likely to require different filters and detection techniques from the Lepton.

b. Obstacle Location After identifying an obstacle in an image, the next step is to determine its location in space to allow Sailbot to avoid it. We will not attempt to estimate the distance to an obstacle, only the direction. Figure 15 shows our procedure for determining the direction of an obstacle - each obstacle's position in the image is projected onto the horizon line, giving its angle relative to Sailbot's path.

This approach depends on reliable identification of the horizon line. Waves, rain, and fog make finding the horizon from the image unreliable at best. Instead, we will use the IMU to determine Sailbot's orientation, allowing us to infer the position of the horizon - the Test Rig 2.0 includes an IMU to support this. We will combine angular rate integration of the gyroscope data (to respond rapidly to changes in orientation) with a moving average of the direction of gravity as measured by the accelerometer (to provide a reliable reference point to correct for drift in the gyro measurements) to determine orientation.

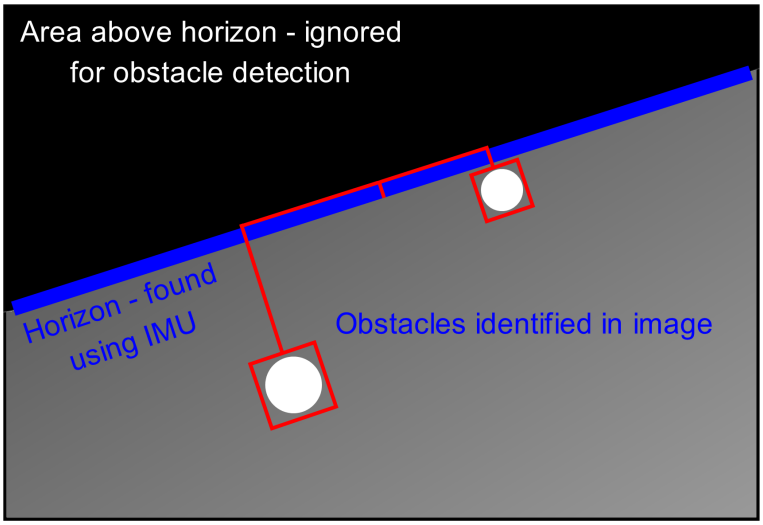


FIG. 15: Direction of obstacles is determined by projecting them onto the horizon line

c. Routemaking Integration After locating obstacles in space, Sailbot needs to plot a course around them. The UBC Sailbot team is currently developing the routemaking software that will control Sailbot's navigation. This software assigns "hazard values" to a grid of points and plots a course that minimizes the encountered hazards, allowing the Sailbot to avoid known, fixed obstacles such as islands. Our obstacle detection system will be

integrated into this hazard map, allowing the same routemaking software to avoid detected obstacles.

When an obstacle is identified in a frame, we will add to the hazard value of points in a cone centered on its estimated angle, as shown in Figure 16. These hazard values will decrease over time - false detections will fade out, but real obstacles will be visible over several frames, increasing their hazard value enough to cause routemaking to steer around them.

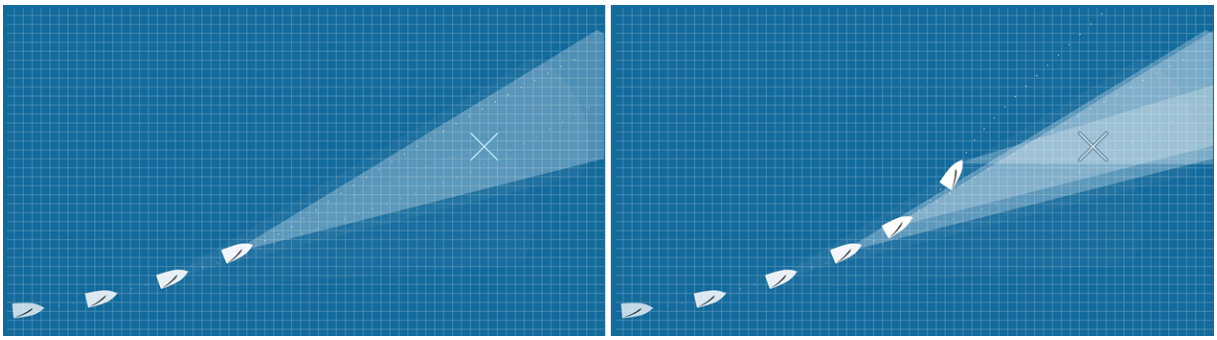
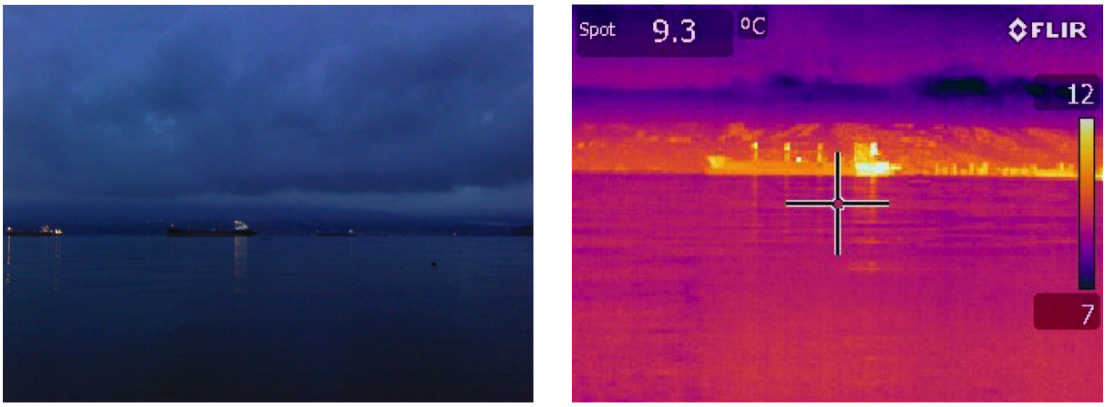


FIG. 16: Routemaking

2.4. Results

At the beginning of the term, we borrowed a high-quality infrared camera from the UBC Mechanical Engineering department to test the appearance of obstacles in the long-wave infrared spectrum. These images appear in Figure 17 and Figure 18.



(a) Optical camera. (b) Infrared camera.

FIG. 17: Comparing images of a distant freighter.

(a) Optical camera.

(b) Infrared camera.

FIG. 18: Comparing images of a person and a log on the water.

These images gave us great enthusiasm for the potential of infrared imaging in our obstacle detection. Not only is the person clear and crisp against the water, but the log as well and the freighter has large hotspots to make detection relatively easy. However, the camera these sample images were taken on was very high quality, and we knew the cameras in our price range would be much lower resolution.

2.4.1. Ucluelet Testing

The first testing with Test Rig 1.0 in Ucluelet took place over three outings spread over two days. With basic image capture and video encoding code running on the FLIR Lepton camera we purchased, we shot video during a clear sunny afternoon, just before and after sunset, and during foggy overcast weather the next day. With the Lepton wirelessly transmitting video back to a laptop we kept in the cabin of the boat, it was immediately obvious the difference that the camera's resolution makes. While the initial images shown earlier in this section were taken on a high-resolution IR camera, the Lepton had only 80x60 pixel resolution in a 50 degree field of view. This means that when looking at objects 100 meters away, each pixel covers well over a square meter of space. Nevertheless, this round of testing gave valuable insight into the function of the camera in a diverse set of atmospheric conditions. While on approach to each obstacle in turn, the captain of the boat gave distance readouts from his onboard radar system which we timestamped to the videos to interpolate the approximate distance to each object at each point in time.

(a) Boat at 200 meters on a sunny day

(b) Boat at 150 meters at night

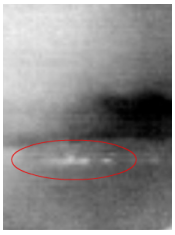
FIG. 19: Comparison of a distant boat during the day and at night.

Figure 19 shows the Lepton's view of a distant boat. When watching the video or viewing the images, the boat is easy to track by eye but at this point the boat appears only two or three pixels wide, too low of a threshold for the detection algorithm to pick it up. The video of a boat approach at night shows very similar results; while the night video does show a lens flare characteristic of the video shot at that time, aside from the flare defect the boats appear similarly visible during the day and night both.

Buoys were also readily visible during the day. Figure 20a shows a buoy on approach during the day. Since buoys are almost stationary, we could approach very close by them and obtain much higher quality images than what was possible for boats. Not only are the buoys distinctively shaped and slow moving, but they also have mounted lights that make them highly visible. The images of buoys taken on this trip give us high confidence that the Sailbot will be able to identify and avoid buoys with relative ease.

Images taken during the second day of Ucluelet testing are dramatically different from those taken the first day, however. While the first day has clear skies, the second day was overcast and foggy. This weather, possibly combined with problems in our field correction code, led many of the videos to appear highly washed-out; the horizon is not clearly visible, which tells us that any horizon finding technique can't be optically driven, and objects are only visible at greatly reduced ranges. Figure 20b shows an image of a boat taken through fog; while still visible, all context clues are removed from the image.

(a) Buoy at 50-75 meters



(b) Small boat at 100m through fog

FIG. 20: Objects observed during Ucluelet testing

Aside from information gathered on the infrared appearance of objects at sea, this round of testing also exposed several image defects that pose problems for our detection software (shown in Figure 21). Scaling issues made sections of the images appear lighter or darker than they should be, consistent flaring on the left side of the camera (regardless of our heading or the position of the sun) often obscured objects, and reflections from the sun created fleeting "objects" that could cause false positives.

(a) Image artefacts left by incorrect scaling

(b) Flaring that obscures objects

(c) Sparkling caused by the sun reflecting. Potential false positives.

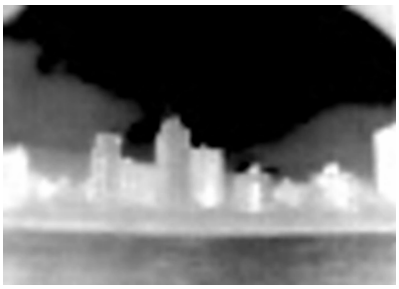
FIG. 21: Defects observed during Ucluelet testing.

One of the main priorities for the development of Test Rig 2.0 was eliminating these defects; after creating more reliable connections and robust communication with the Lepton, none of these defects were seen again in the second round of on-ocean testing.

2.4.2. North Vancouver Testing

The second round of on-water testing took place on Wednesday, April 1 in North Vancouver. The goal of this test was to integrate the inertial measurement unit and eliminate as many image defects as possible. The IMU was seen to be necessary during the first round of testing, when fog obscured the horizon and many long-distance objects. Since knowing the position of the horizon is important for eliminating clouds (and the sun) from being identified as obstacles, tracking orientation with an IMU is the next best way to track the horizon. Each frame was associated with a set of IMU readings - three axes of each an accelerometer, gyroscope, and magnetometer. This provides a base of test data for future development of the orientation tracking code. Furthermore, the image scaling algorithm was changed to better handle very bright spots (such as the sun) without blacking out the rest of the image. During this entire round of testing neither the artefact nor flaring defects were observed; we're confident that they were fixed by a combination of better cabling and connections and reprogramming of the Lepton software for better pixel drift corrections. This round of testing was much shorter than the first Ucluelet outing, with fewer boat and buoy approaches available. In addition, the Lepton was oriented correctly in this rig design so that the wider field of view ran horizontally, rather than vertically as in Test Rig 1.0.

(a) A distant freighter and the Sun



(b) Image of the shoreline at several hundred meters

FIG. 22: Images captured during the second round of testing.

2.4.3. Realtime Obstacle Detection

Our two rounds of testing produced a wealth of data to feed into our obstacle detection system. After the Ucluelet round of testing, our project ran in two parallel veins; one to improve the hardware and produce Test Rig 2.0, and one to pipeline and produce a working version of obstacle detection software. The detection software currently evaluates the video frame by frame (i.e without any optical flow algorithms) by segmenting the image to discard inappropriate regions (e.g the sky), then finding lines and contours, then identifying regions of high contour density to locate an object. When tested with the Lepton images, the software in it's current form can identify objects at a maximum range of 100 meters. Figure 23 shows the software identifying a buoy at approximately 50 meters, and outlining the area where it locates that object in a white bounding box.

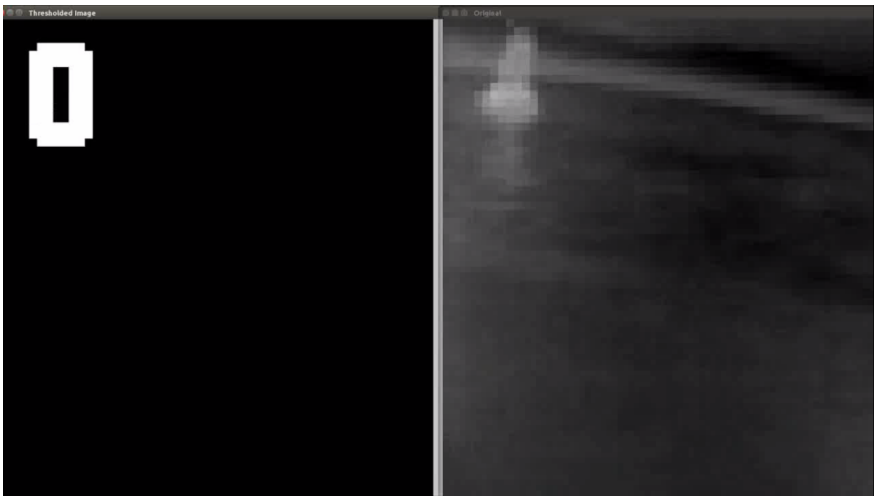


FIG. 23: Identifying a buoy at 50 meters

The identification software currently runs in real-time (between 20-27 frames per second, depending on the information in the picture and responsiveness of the Lepton) while gathering but not using the data from the IMU. None of the boats in our database could not be picked up by the current detection software, but in all our testing we never approached another boat close enough to become a collision hazard (in order to comply with maritime law, as well as respecting the space of other vessels who couldn't know our testing purpose). Based on the success when identifying buoys, we have confidence that the system could identify a boat in its current state, should it approach closely. Further, we had a very low false positive rate when masking off above the horizon (a process which should be automated by the IMU in our ongoing work with Sailbot, slated to end late April).

3.0. CONCLUSION

The development of an automated machine vision-based obstacle detection system for the Sailbot's 2015 MicroTransatlantic Challenge is a difficult problem. Our project has helped the Sailbot team evaluate the effectiveness of and implement an infrared-based obstacle detection system, initially selected due to its low cost and ease of implementation, despite the dearth of literature describing its use in similar challenges.

Throughout the term, we developed Test Rigs, to protect the FLIR Lepton infrared camera during testing, and providing the requisite power as well as hardware and software control in a convenient, waterproof package. These Test Rigs will also inform the final design of the distributed housing, where the infrared camera will be mounted to a tripod on the boat and the electronics will be mounted in the hull of the boat, several meters away.

Our testing showed the effectiveness of the tripod mount for the camera, confirmed the functionality of serial communication between the Raspberry Pi and Lepton over several meters of cable, and our design and development led to a sturdy baseplate to which a small infrared camera can be secured. This baseplate can then be secured to any other compatible module, such as a Test Rig.

We also provided several battery packs, an inertial motion unit, and plan to supply a zinc selenide infrared-transparent lens to the Sailbot team, to assist in testing and to be incorporated into the final design. In particular, we are currently assisting the Sailbot team in decoding data from the inertial motion unit, and design a lens cleaning system to keep clear the camera's field of view.

In cooperation with the Sailbot team, we were able to identify some floating obstacles using machine vision algorithms on the low-resolution Lepton camera images acquired with our Test Rig. After the presentation of our joint work with Sailbot on obstacle detection won the top prize among oral presentations in UBC's 2015 Multidisicplinary Undergraduate Research Conference, Sailbot acquired new sponsorships which allow for purchasing much higher quality IR cameras than the Lepton used in our testing; and as such, we finally recommend the use of the higher resolution FLIR Quark camera, which will provide greater detection range of floating objects.

4.0. PROJECT DELIVERABLES

4.1. List of Deliverables

Throughout the past three months, the scope of our project has changed. Our deliverables are grouped according to those that will be handed off to the sponsor at the end of term (Section 4.1.1) and those that were initially planned but either scrapped or are ongoing at the time of this report (Section 4.1.2) and not included in the project completion report.

4.1.1. Current Deliverables

Test Rig hardware 2.0: The second design of a plastic waterproof housing, with laser-cut internal plastic supports and mounting structure. A 3D-printed housing secures the FLIR Lepton infrared camera and breakout board, and the internal mounting structure supports this part as well as a Raspberry Pi computer, a rechargeable battery pack, an inertial measurement unit (IMU), and a button to start and stop recording. The Test Rig can be clamped to a tripod. Pieces of the test rig (particularly the 3D-printed camera mount) will serve as a model for the final housing of the infrared camera.

Test Rig software: The Test Rig software records IMU data and infrared images at about 20 frames per second, and allows this data to be viewed and transferred wirelessly from the Test Rig hardware. The software is written in Python and C++ and is synchronized to a GitHub repository; it was developed in conjunction with the Sailbot Obstacle Avoidance team.

Support equipment: We obtained an IMU and two battery packs for the Sailbot team to use with the Test Rig and in the final obstacle detection system.

4.1.2. Former Deliverables

AIS signal processor (receiver, connectors, software library):

- written in Python
- interpret data from AIS receiver
- report position and speed of nearby vessels to Sailbot navigation software

Responsibility for this item was taken over by the Sailbot Electrical team.

IR signal processor (camera, connectors, software library):

- written in Python
- capture IR images ahead of Sailbot
- detect and track features in image
- determine position and speed of features in image
- report position and speed of detected objects to Sailbot navigation software

The pieces of this task that have been completed are outlined in Section 4.1.2. The image analysis code is still in development in part by us (see Section 4.3) and in part by the Sailbot Obstacle Avoidance team.

Report on performance characteristics for all systems:

- detection distance
- detection likelihood
- best weather conditions for operating
- power draw
- input data required from Sailbot sensors
- frequency of data output

As the image analysis code is incomplete, and the final infrared camera has not been chosen, this item will not be completed.

4.2. Financial Summary

Table I shows a breakdown of the total cost of the project. The items purchased by the team members are intended to be eligible for later reimbursement by the Project Lab. All costs from international websites have been converted to CAD on April 5, 2015.

TABLE I: Financial summary (all costs in CAD)

Item	Quantity	Unit cost	Purchaser	Vendor
Battery pack	2	24.99	Team members	Amazon
Raspberry Pi Model B+	1	49.85	Sailbot	Sparkfun
FLIR Lepton IR Camera	1	436.53	Sailbot	Sparkfun
IMU	1	18.16	Team members	Newark
Test Rig internal structure	1	30.00	Project Lab	N/A
Graphics LCD Screen	1	70.00	Team members	Lee's Electronics
IR transparent window	1	87.99	Project Lab	EKSMA Optics
Total		742.51		

4.3. Ongoing Commitments

Housing design with water wiper: This is a modification of the existing Test Rig housing including a small wiper to clear the window in front of the infrared camera lens. The wiper is actuated by air moving over the housing, and will be tested in a wind tunnel. **Expected completion: May 1, 2015.**

Final housing: This is a housing that can be mounted on the autonomous Sailbot during its transatlantic crossing. **Expected completion: August 1, 2015.**

IMU decoder: Written in Python or C++, this code will communicate with an inertial motion unit (IMU) to extract and decode the pitch and roll of the Sailbot in real time. **Expected completion: May 1, 2015.**

IMU gravity identification: Written in Python or C++, this code will examine the accelerometer and gyroscope data obtained from the IMU in order to determine the direction of gravity. This is essential to accurately determine the pitch and roll of the Sailbot. **Expected completion: May 1, 2015.**

Horizon predictor: Written in Python or C++, this function will accept pitch and roll data from the IMU decoder, and identify the set of pixels on the image from the infrared camera where the horizon should lie. **Expected completion: April 10, 2015.**

5.0. RECOMMENDATIONS

1. Switch from the FLIR Lepton to the Quark2 camera. The Quark2's resolution of 640x480 pixels is a major improvement over the Lepton's 80x60, and will substantially increase the range and reliability of obstacle identification. The housing for the infrared camera was designed with the intent that it's easy to modify to accomodate for a Quark camera, and the software flow to take images into obstacle detection has been built. New interfacing protocol will be needed, as the Quark camera has seperate drivers and communication protocol from that used for the Lepton, but FLIR provides software to handle this interface. Upgrading to this camera will allow for obstacle detection from much greater ranges, be less prone to failure (as the Quark has a shutter, and so better flat-field correction functions to account for pixel drift) and, as it has greater resolution, will be able to work better in foggy or rainy conditions as it can gather more information from a given field of view.

2. Refine obstacle identification software based on video from new camera, to make use of improved resolution. The current iteration of the obstacle detection software uses minimal smoothing/ noise reduction techniques, as the information from the Lepton is already in a highly lossy state. The Quark will require more robust smoothing functionality to accommodate for the increased information density, particularly in the water. Higher resolution pictures of the water show more of the high-intensity sun reflections that the images taken with the Lepton (see Figure 21c for more information).

6.0. APPENDICES

A. Obstacle detection

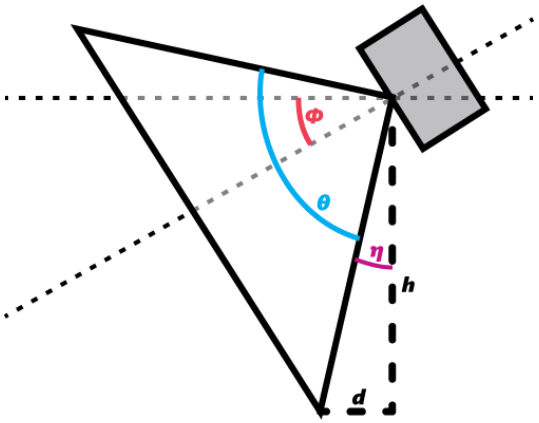


FIG. 24: Minimum detection distance for a directional emitter.

A.1. Minimum detection distance

Constraints on the viewing area appear with any method of detection, whether radar, lidar, on infrared. For directional emitters, the minimum detection range is given by trigonometry and depends only on the vertical beam width and the mounting angle of the device. $d = h/\tan(\eta)$, where $\eta = \pi/2 - (\theta/2 + \phi)$ given by θ, the vertical beam spread, and ϕ, the emitter mount angle. Figure 24 details the trigonometric analysis.

A.2. Frequency of floating obstacles

Given the vast size of the Atlantic ocean, floating obstacles are rare. Nevertheless, it is instructive to provide a reasonable guess of the space between obstacles of different types.

From [9], we can say that there are about 50000 container ships in the world at any given moment. These travel at about 25 knots (50 km/h) [12], making the 3000km Atlantic crossing in about 20 days. The turnaround time (the length of time spent in port is about 1 day [3], which will be neglected here).

Let's assume 20% of the world's shipping occurs somewhere in the north atlantic, near our route. The Microtransat organisers estimate that it will take our boat two to three months to complete the crossing, so that a given ship can be assumed to cross our route

TABLE II: Obstacles and risk

Obstacle	Location	Inverse Frequency	Damage
Shipping vessels	Entire route	4 days	Severe
Icebergs	Entire route	400 days	Severe
Logs	Entire route, drawn to gyres	40 days	Moderate

five times during our trip. Eyeballing, the area of ocean of interest, bounded by the most northern and southern routes the Sailbot is likely to take, is 1/10 of the full 100 Mkm size of the Atlantic [31].

So, we have $\frac{10000000}{0.2*50000*5} = 200$. This is the average area of ocean, in km^2, containing one shipping vessel.

To convert to a frequency, we can assume that ships are stationary and expand the width of the Sailbot to a typical length of a ship, say 100m. Traveling at 20km/h, the Sailbot then sweeps an area of $2km^2/h$, leading to one ship encounter per 100 hours or roughly 4 days.

Now, during peak season, there are about 1000 icebergs in the North Atlantic [18]. This is about 50 times less than the number of ships, leading to an inverse area density of one per 10000 km^2, and a frequency of 1 encounter per 400 days

B. Software Testing Methodology

This appendix outlines a particular form of test driven development, inspired by Peter Provost [23] and James O. Coplien [2].

The correct functioning of software is checked at every stage of development, using three levels, two of which are automated. The three levels are unit tests and functional tests, performed by testing software; and user tests, performed manually.

Software code not depending on hardware interaction is examined through unit tests, which verify that functions produce the correct output under all anticipated conditions. These tests are written before or simultaneously with the code to be tested.

The emergent, holistic functionality of the software (including user interface elements are outputs) can be tested using functional tests that provide simulated realistic input, possibly simulating hardware components, and examine the output.

The final test level is user tests, which include testing on the actual vessel in real and simulated environments, as well as any other manual testing.

These test levels are not performed in sequence; all testing levels should interact with the development process, and be re-evaluated whenever a notable change is made. Figure 25 shows the connection of testing and development.

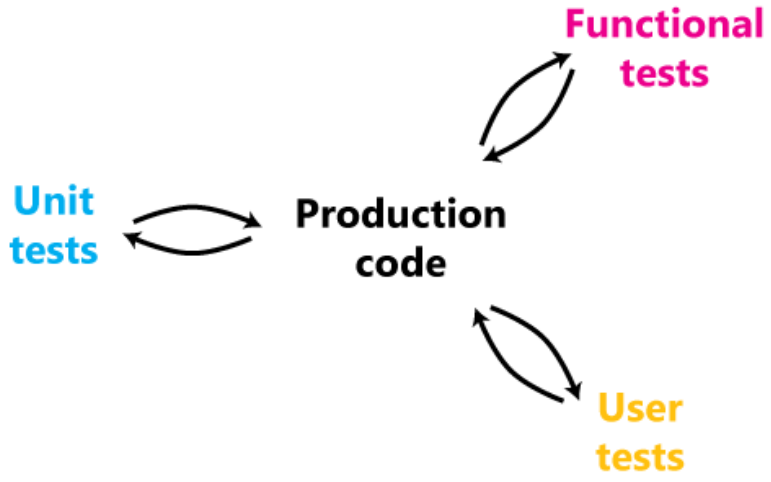


FIG. 25: Development cycle. Tests are written before or simultaneously with production code, and both automated and manual testing is run intermittently during development.

[1] Elkins, L., Sellers, D., & Monach, W.R. (2010). *The Autonomous Maritime Navigation (AMN) Project: Field Tests, Autonomous and Cooperative Behaviors, Data Fusion, Sensors, and Vehicles*. Retrieved from http://onlinelibrary.wiley.com/doi/10.1002/rob.20367/pdf

[2] Coplien, J.O. *Why Most Unit Testing is Waste*. Retrieved November 19, 2014 from http://www.rbcs-us.com/documents/Why-Most-Unit-Testing-is-Waste.pdf

[3] Ducruet, C., & Merk, O. (2013, September). Examining container vessel turnaround times across the world. *Port Technology International*, 19-20.

[4] Flir. (2014). *FLIR MD-Series Thermal Imager*. Retrieved from http://www.flir.com/cvs/americas/en/maritime/view/?id=59356&collectionid=1129&col=59377

[5] Furuno. (2008). *Operator's Guide to Marine Radar*. Retrieved from http://www.getfeetwet.com/FurunoRadarGuide-LR.pdf

[6] GAM Electronics. (2014). *SS-2 Mini VHF Antenna*. Retrieved from http://gamelectronicsinc.com/product/ss-2/

[7] GPSD Project. *About GPSD*. Retrieved November 27, 2014, from http://www.catb.org/gpsd/

[8] Jackson, C.R., & Apel, J.R. *SAR Users Manual*. Retrieved from http://www.sarusersmanual.com

[9] Jallal, Craig. (July 31, 2012). *How Many Ships Are There in the World?* Retrieved from http://shippingresearch.wordpress.com/2012/07/31/how-many-ships-are-there-in-the-world/

[10] Kaehler, Adrian, & Bradski, Gary. (October 2013). *Learning OpenCV: Computer Vision in C++ with the OpenCV Library*. O'Reilly Media.

[11] Mace, T. (2011). *At-sea detection of marine debris: Overview of technologies, processes, issues, and options*. Marine Pollution Bulletin. doi: 10.1016/j.marpolbul.2011.08.042

[12] Maersk. (2014). *Triple-E Vessels*. Retrieved from http://www.maersktechnology.com/stories/stories/pages/triple-evessels.aspx

[13] Marine Traffic. (2014). *AIS Traffic and Locations*. Retrieved from https://www.marinetraffic.com/

[14] Maritime and Coastguard Agency - UK. (September 14, 2006). *Maritime Guidance Note (MGN) 324*. Retrieved from https://mcanet.mcga.gov.uk/public/c4/solasv/m_notice/mgn/mgn324.pdf

[15] Milltech Marine. *AMEC CYPHO-150 AIS Receiver*. Retrieved December 29, 2014 from http://www.milltechmarine.com/AMEC-CYPHO-150-AIS-Receiver_p_263.html

[16] Microtransat Challenge. (2014). *History of the Microtransat Challenge*. Retrieved from

http://www.microtransat.org/history.php

[17] Microtransat Challenge. (2014). *West to East Route Start and Finish Lines*. Retrieved from http://www.microtransat.org/tracking/westeast.html

[18] National Geographic. (2008). *Iceberg Frequency in April*. Retrieved from http://education.nationalgeographic.com/education/photo/icechart-frequency/?ar_a=1

[19] National Marine Electronics Association (NMEA). *Standards (NMEA 2000, 0183)*. Retrieved November 27, 2014, from http://www.nmea.org/content/nmea_standards/nmea_standards.asp

[20] Old Dominion University Vision Lab. (2009). *Small Boat Detection for Port Security Applications*. Retrieved from http://eng.odu.edu/visionlab/research/boat.php

[21] Optotherm Thermal Imaging. (2014). *Emissivity Tables*. Retrieved from http://www.optotherm.com/emiss-table.htm

[22] Pure Engineering. *Lepton*. Retrieved from http://www.pureengineering.com/projects/lepton

[23] Provost, P. (September 2013). *What Really Matters in Developer Testing*. Retrieved from http://www.peterprovost.org/blog/2013/09/23/what-really-matters-in-developer-testing/

[24] Raymarine by FLIR. (2014). *Raymarine Marine Radome* Retrieved from http://www.raymarine.com/view/?id=2966&collectionid=96&col=3021

[25] Raspberry Pi Foundation. (2015). Retrieved from http://www.raspberrypi.org/

[26] UBC Engineering Physics Project Lab. (2014). *Technical Info 2013*. Retrieved from docs.google.com/document/d/1WU_duqr20lqg48ukjsuhUJPJhsSNkrWiEVf3uU4xS0A/pub#h.f2e6v0ormnoc

[27] UBC Sailbot Team. (2014). *Why We Do This*. Retrieved from http://ubcsailbot.org/

[28] United States Coast Guard. *AIS REQUIREMENTS*. Retrieved November 27, 2014 from http://www.navcen.uscg.gov/?pageName=AISCarriageReqmts

[29] Velodyne High Definition Lidar. (2014). *Velodyne VLP-16*. Retrieved from http://www.velodynelidar.com/lidar/hdlproducts/vlp16.aspx

[30] WeatherDock. (2014). *EasyAis 025*. Retrieved from http://www.easyais.com/en/products/easyais-a025/

[31] WorldAtlas. (2014). *Atlantic Ocean Details*. Retrieved from http://www.worldatlas.com/aatlas/infopage/oceans/atlanticocean.htm

[32] Manduchi, R. et al. (2005). *Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation*. Retrieved from http://link.springer.com/article/10.1023/B:AURO.0000047286.62481.1d

[33] Mallot, Hanspeter A. et al. (1991). *Inverse perspective mapping simplifies optical flow computation and obstacle detection*. Journal of Biological Cybernetics. Retrieved from http://link.springer.com/article/10.1007/BF00201978

[34] Matthies, L. et al. (1996). *Obstacle Detection for Unmanned Ground Vehicles: A Progress Report*. Robotics Research. Retrieved from http://link.springer.com/chapter/10.1007/978-1-4471-0765-1_52

[35] Heidarsson, H. & Sukhatme, G. (2011). *Obstacle Detection and Avoidance for an Autonomour Surface Vehicle using a Profiling Sonar*. Retrieved from http://robotics.usc.edu/publications/media/uploads/pubs/693.pdf

[36] Schifano, W. 1975. *Device for Forming a Protective Airflow Forward of Flat-Fronted Vehicles*. Retreived from https://www.google.com/patents/US3862777