2016 RobotX Journal Paper

Aricia Argyris, Ileana Argyris, Amy Bentz, Karla Cortez, Steven Cory, Jonathan De Leon, Raina Ann Duenas, Richard Eidswick, Nicole Clare Hortizuela, Michael Huang, Kelan Ige, Christianne Izumigawa, Minshik Kang, Steven Kim, Paulo Lemus, Michael Loui, Aaron Nagamine, Nathan Park, Kobe Taylor, Eric Welton, Allison Wong, and Darren Wong

FACULTY ADVISOR: A ZACHARY TRIMBLE
GRADUATE ADVISOR: BRENNAN YAMAMOTO

***Abstract*—Team Kanaloa, comprised of students from the University of Hawaii at Manoa, was given an opportunity to compete in May of 2016. With a six month time period, many design decisions were driven by this constraint. The goal was to execute the Demonstrate Navigation and Control, Find Totems and Avoid Obstacles, and Detect and Deliver. Unfortunately, Team Kanaloa was unable to successfully complete a single task; However, the experience and opportunities gained from this competition would establish a strong foundation for future RobotX competitions.**

I. INTRODUCTION

Team Kanaloa will be representing the University of Hawai‘i at Mānoa in the 2016 Maritime RobotX Challenge. The team was given an opportunity to participate in this competition in May of 2016. Due to the limited time budget of six months, all design decisions were heavily influenced by this constraint. This also drove the decision to allocate resources to execute certain tasks rather than all of them.

Team Kanaloa's mission statement declared that the team will perform the following tasks: Demonstrate Navigation and Control, Find Totems and Avoid Obstacles, and Detect and Deliver. Demonstrate Navigation and Control, also referred to the Preliminary Task, was selected as this was a requirement to compete in the competition. Find Totems and Avoid Obstacles was selected because the requirements to complete this task, were fundamentals of an autonomous vehicle. As a team, successfully executing this task would demonstrate that the WAM-V has the core functionalities to become an autonomous vehicle and would establish a foundation for future RobotX competitions. Detect and Deliver was selected due to the dominant presence of mechanical engineering majors on the roster. It was best suited to execute a task that tended to the team’s strengths rather than its weakness.

Team Kanaloa's definition of success for this project was to perform these tasks robustly and reliably by obtaining at least 80% of the total points for each task. Eighty percent was selected as the team concluded that the percentage stated, quantified a competitive score. Meeting this expectation at competition will demonstrate competency and prove the team's success. The team's long-term goal for this project was to establish a foundation within the University of Hawai'i for future Maritime RobotX competitions. As such, the team will promote the integration of undergraduate and graduate students from various majors and skill levels to collaborate in maritime robotics projects.

II. DESIGN STRATEGY

A. Subsystem Division

Team Kanaloa was divided into four subsystems: Surface, Computing, Power, and Safety. The Surface subsystem was responsible for the following sub-subsystems: Sensor Selection, Object Manipulator, and Propulsion. The team members within the surface sub-subsystem were responsible for the design, hardware selection, and mounts for integrating onto the WAM-V. The Computing subsystem is responsible for the following sub-subsystems: Robot Operating System (ROS), Simultaneous Localization and Mapping (SLAM), Path Planning, and Color Recognition. The team members within the Computing sub-subsystem were responsible for the selection of software and computing-specific hardware as well as the development of codes. The Power subsystem is responsible for battery selection and allocation of power to all hardware prescribed by the Surface and Computing subsystems. The Safety subsystem is responsible to oversee everything complies with the safety regulations outlined by the competition.

B. Surface Subsystem

The surface subsystem design process and design decisions focuses on meeting the system and functional requirements set forth by the team. The system requirements were set forth based off the tasks of the competition. The system requirements for the surface subsystem is represented by following modules- Localization, Object Perception, Image Recognition, Wireless Communication, Object Manipulator, and Propulsion. Accompanying the system requirement are the functional requirements which is written in terms of quantitative performance to elaborate on the system requirements.

To address these requirements, various solutions to a given surface system (sensors, object manipulator, and propulsion) was considered and decided on based on how best they met the functional requirements. To do this, Decision Making Matrices (DMM) or Error Budgets were created to help weigh the merits of all solutions. Due to our limited budgets, ease of implementation, cost and availability were key factors when deciding on our final solution.

1. Sensor Selection

The sensor selection was based on the system and functional requirements. Decision Making Matrix (DMM) was used to compare the variety of sensors that meet the team's functional requirements for the modules of localization, object perception, image recognition, and wireless communication. Most of the sensors were ultimately decided based on their availability and their ease of implementation. This resulted in some tradeoff of performance as some low-end, but easier to implement sensors were chosen over high-end sensors.

2. Object Manipulator

One major engineering design decision made by the team was for an object manipulator to be used in the Detect and Deliver task. The team chose to focus on aiming for the smaller multiplier hole during competition, as we were looking to maximize our points in the few tasks we will be attempting.

3. Propulsion

Trolling motors, Jet-pump, and Rim Driven Propellers (RDP) were evaluated and compared for the thruster options. These solutions were chosen because they had the potential to provide a wide range of drive configuration possibilities to accomplish the Find Totems and Avoid Obstacles task, as this is the seen as the most challenging task to maneuver through. A modified drag equation was created to model the trust requirements:

$$F=ma+12C\rho V2A+12Cq(V+T)2Z \quad (1)$$

The coefficient of drag of the WAM-V, *C*, was previously determined by Team Topcat [3]. The density of the sea-water and atmosphere are ρ and q respectively. V is velocity of the WAM-V and T is the velocity of the wind. A and Z are the respective reference areas. This model was used in determining the thrust required to propel a fully loaded WAM-V at 3 m/s with an acceleration of 1 m/s^2.

Three drive configuration options: Differential Drive, 'T' Drive, and Holonomic Drive, as shown in Fig 1, were considered. Differential drive is the simplest to implement with the least amount of cost. It only requires at least one thruster per pontoon. However, Differential does not enable the WAM-V to achieve lateral movement. A 'T' drive configuration build off of the Differential drive configuration. In a 'T' drive, at least one thruster is placed in such a fashion where it's thrust vector is perpendicular to the trust vectors of the other thrusters. The additional thruster enables the WAM-V to move laterally. A Holonomic drive configuration provides the greatest degrees of motion, but it requires at least one thruster per corner of the WAM-V. The thrusters in a Holonomic drive need to be orientated in such a way that no two thrust vectors are parallel. The configuration is required to make the WAM-V attain an angular velocity of 5 degrees per second.

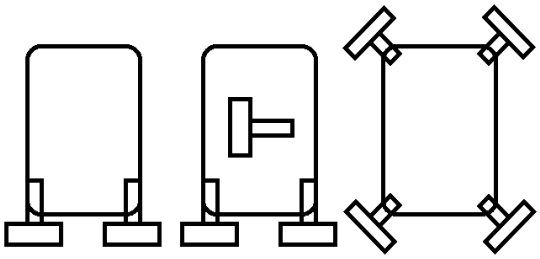


Fig 1. Diagram of the Differential, 'T', and Holonomic drive configurations respectively.

A maneuverability index was created for the thruster orientation to help determine the optimal angle at which the motors are mounted. To find the optimal angle, the functional requirements were taken into account, and analysis was done to ensure an optimal balance between forward thrust and maneuverability (e.g. lateral velocity and angular velocity) was achieved. Truster separation, total trust, thrust provided per thrust, environmental variables (e.g. wind and current), and were factors in determining the optimal angle of 30 degrees.

C. Computing Subsystem

The computing subsystem handles all software choices, computing-specific hardware selections, and the design of the autonomous system for the WAM-V. Essential ingredients to achieve autonomous behaviors are perception, reasoning, and motion planning capabilities, which aim to answer the corresponding questions:

1. Where am I and what is around me?
2. What do I need to do?
3. How do I do it?

Due to lack of programming, robotics, and control systems experience within the team, software and computing hardware options were carefully considered with learning curves and ease of development heavily weighted in order to alleviate the considerable time and manpower budget constraints. Robot Operating System and MATLAB were chosen as the primary working environments due to their accessibility, integration with each other, and gentler learning curves.

III. VEHICLE DESIGN

A. Surface Subsystem

1. Sensor Selection

In order to fulfill all the system requirements for the surface subsystem, multiple sensors were selected. These sensors include a global positioning system (GPS), an inertial measurement unit (IMU), a light detection and ranging (LiDAR), and a camera. The predominant localization method is the GPS. Accompanying the GPS is the IMU. Both the GPS and Razor IMU were from SparkFun for easier integration with Arduino despite the tradeoff of performance. The GPS receiver offers a 2.5m positional accuracy. The Hokuyo UST-20LX LiDAR was selected as the sensor for object perception. This lidar offers a maximum detection distance of 60 meters,

angular resolution of 0.25 degree, and refresh rate of 40 Hz. A LiDAR was chosen primarily due to its capability to produce a mass point cloud dataset. The LiDAR was then incorporated with SLAM to provide a real-time map for our WAM-V. The Logitech C920 was selected as the sensor for image recognition. This webcam offers a 1080p video recording at 30fps and a 78 degree diagonal field of view (FOV). A webcam was chosen over a point and shoot primarily due to the cost and interface advantages. A combination of a Wi-Fi router and an access point was selected for the module of wireless communication. The Wi-Fi router is a standard router whose range is extended up to 500m with the Ubiquiti PicoStationM2HP. Figure 2 below displays a visual of the hardware placement on the WAM-V.

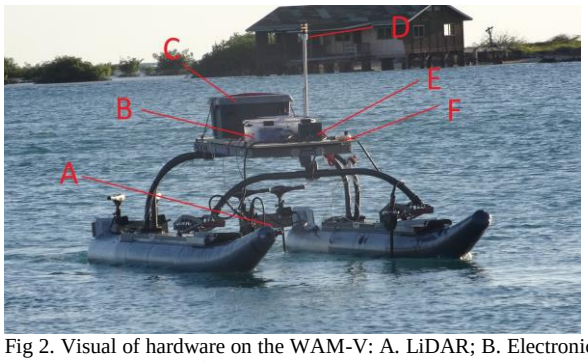


Fig 2. Visual of hardware on the WAM-V: A. LiDAR; B. Electronics case; C. Battery housing; D. Safety light and access point; E. Webcam; F. IMU.

The sensors had to be integrated into a core framework that would allow other parts of the system to access their data and fuse them together. ROS used as a framework for our entire system and allowed the data from sensors to be shared between modules as needed. The core component of the system is the LiDAR as it allows the WAM-V to detect objects and is currently used directly with GPS . With the LiDAR being a 2D LiDAR, the WAM-V only had a view of a plane with 270 degree angle around it. As a result, the LiDAR had to be mounted low so it could see obstacles in front of the boat. Additionally, marine environments are highly dynamic so perturbations of the LiDAR produced mapping errors. These errors are reduced by considering the roll and sway data from the IMU in conjunction with the LiDAR data in order to compensate for any movement in the defined 2D mapping plane. Rudimentary sensor fusion has been implemented to integrate the camera data with the LiDAR data so that the WAM-V can correlate object color and shape with object position.. Currently, the team is in the process of implementing camera data with the LiDAR by taking heading information from the camera and checking the map for an object in that direction.

2. Object Manipulator

For the object manipulator, the team decided on a single-link telescoping arm. The single-link arm had the lowest value on the error budget, the lowest cost, and was a simple yet effective design. To complete the Detect and Deliver task, the WAM-V maneuvers into position alongside the buoy, then the arm extends to the buoy and delivers the racquetballs into the hole. The final arm design consists of one square aluminum tube, acting as the telescoping inner tube, resting within a rectangular aluminum tube, with a pin to guide the sliding motion of the inner tube. The arm's telescopic feature is driven by a chain-sprocket mechanism and a stepper motor. A gear motor driving a gearbox at the base of the arm allows the arm to move vertically.

To select an appropriate design for the object manipulator, an error budget was used to compare a shooter and a robotic arm. An error budget model allocates errors to a process which, in this case, is the process of manipulating the racquetballs into the target hole on the buoy. This error budget predicted the accuracy of the final position of the ball relative to the center of the target hole.

Main Arm

One of the designs the team considered was a projectile launcher. However, this design was eliminated due to its high error value in the error budget. Although simple in design, it proved to be impractical in the setting of the competition location, where strong winds were an important variable to consider. Strong winds greatly affect the path of a projectile; had we chosen a projectile launcher, our accuracy in completing the task would have been much lower. Our next best option was determined to be a robotic arm. A multi-link arm with a grabber was first considered. However, this design was eliminated due to several reasons, including cost, precision, and time. The multiple motors and extra material required for a multi-link arm would have added considerably to the cost of the object manipulator. The next reason for this design's elimination was precision. If the WAM-V were to move while the arm was grabbing another ball from the payload tray, this could cause the arm to lose accuracy when repeating this action. The third reason for elimination was time. Compared to other designs, a multi-link arm design would have required a significant amount of time for programming and manufacturing. With the extremely limited timeline the team had to prepare for this competition, extra labor had to be eliminated wherever possible.

The team eliminated a rotating base from the design for similar reasons. The inclusion of a rotating base in the design would have allowed for an extra degree of freedom (rotation about the z-axis). However, due to time and money constraints, the team opted for a stationary base. One major drawback to this decision is that more precise positioning of the WAM-V will be required in order to account for the lack of z-axis rotation of the arm.

End Effector

Several end effector designs were considered for the object manipulator. The team decided that the simultaneous delivery of all four racquetballs into the target hole was the best method for task completion. Through brainstorming, the team came up with a design consisting of a tube to hold the racquetballs and a mechanism to deliver them. The design options the team considered for the delivery mechanism were a conveyer belt, a linear actuator, and spring-loading of the racquetballs combined with a moveable gate to release them when needed. The final end effector design consists of a tube to hold the racquetballs, a rubber band attached to a plunger

for spring-loading the balls, and a stopper that acts as a gate, holding the balls in the tube before moving out of the way to release the balls. The tube for holding the balls is made of 2.5 inch PVC pipe. The stopper is attached to a servo motor which is attached to the end of the PVC pipe.

Motors

The motors for the movement of the arm were selected based on the torque required to rotate the arm , as well as the torque required to extend the inner tube of the arm and hold it in position. After free body analysis was completed on the system to determine the required torques for these two motors, the values came out to 40.146 Nm and 1.477 Nm, respectively. A gearmotor was determined to be the proper motor for rotating the arm due to its high torque. Calculations were done to determine the gear ratio needed for the to operate at a specific current while still supplying the necessary torque to rotate the arm and hold it in position. For the telescopic mechanism, a stepper motor was chosen for its accuracy. An RC servo motor was chosen for the end effector due to its ease of use, weight, and availability.

Coding

Many options were considered when it came to how the team would go about creating a code to work with the arm and accomplish the assigned task. MATLAB was chosen to code the arm because most of the team members were already familiar with MATLAB and its pre-existing applications.

The first part of the code involved detecting the square on the buoy. It was ideal to have the square be in the same conditions as if it were the actual competition prop. A simple 8" x 8" piece of black plastic taped to a white piece of paper served as the test target. Then, 55 photos were taken with the square present and 100 photos were taken without the square to serve as "positive" and "negative" images, respectively. The images were then uploaded to the Training Image Labeler tool, bound boxes were placed on the squares in each individual picture, and the images were saved as a .mat file. This file will be essentially tell MATLAB what to look for. The file was then loaded onto MATLAB to create a Positive Instances file containing the file made, and a Negative Instances file which contains all of the images without a square. The trainCascadeObjectDetector command was used to create an .xml file and choose the parameters, these being the number of stages and the false alarm rate. The first attempt wasn't successful because as the sample images were loaded for testing, it would see objects other than a square and label them as squares. To fix this problem, an additional 555 positive images and 500 negative images were taken and included in the files. At the end of testing, the results were much more appropriate and the amount of false alarms was reduced while the program accuracy increased. After detecting the square, coordinates must be placed on the shape in order for the arm to detect the location of the square. This is used to determine the square's location relative to the base of the arm.

The second part of the code interprets the data from the first part and tells the arm which way to move in order to get the end of the arm to the square. Using inverse kinematics, we calculated the required angles and distances for the arm to reach the square. Through arduino, we used these calculations to program the motors for the arm to move the arm in x- and y-directions until the location of the end of the arm coincides with the location of the square.

3. Propulsion

Trolling motors were ultimately selected. Trolling motors met the team's requirements while providing adequate performance. Six total trolling motors are used. At first, four 30 lbs Minn Kota Endura C2 were planned to be used, but these alone did not provide enough thrust to propel the WAM-V as desired. Two additional 55 lbs motors were acquired and mounted near the bow for holonomic drive. The 30 lbs motors were mounted to the rear and front thruster mounts were designed for the 55 lbs motors. The motors were divided in such a way to minimize the difference in the amount of truster from each corner of the WAM-V.

The front mounts mimicked the rear thruster mounts as it would be the simplest to manufacture and took up the least amount of area on top the skis. Aluminum was used as it is lightweight and readily available. The front thruster mounts are attached using the existing fastener points on the skis near the bow of the WAM-V. Also, a mount for the LiDAR was integrated with one of the front thruster mounts, because of its required proximity to the water. The rear thruster mounts are attached to the buoyancy pods with the provided holes. All thrusters are placed inwards in order to reduce the potential of collisions.

Holonomic drive was selected for the final drive configuration, with each thruster oriented 30 degrees from the pontoons. Holonomic drive offered the greatest degree of freedom and flexibility. This angle was decided to be the best when compromising between thrust and velocity. The team decided that the total forward thrust, F_f, was of greater importance than lateral thrust, F_L in order to counteract environmental variables, but while maintaining the ability for finer positioning with lateral movement.

$$Ff=i = 16Ficos(30°) \quad (2)$$

$$FL=i = 16Fisin(30°) \quad (3)$$

B. Computing Subsystem

The computing subsystem autonomy is based on a hierarchical control system architecture as outlined in Fig 3 with a high-level diagram of the hardware and software module data flow. The selection of such a control scheme allows for the various components to be independently developed and incrementally tested while lending itself to retaining a cohesive overall structure and smoothing the overall integration path. [5]

Motion planning and control are handled by a high-level mission planner, a path-planning module, and low-level motion controls. All three modules of the motion planning and control were developed in MATLAB. The mission planner uses finite state machines to control the state switching needed to regulate the behavior of the WAM-V. The mission planner takes input from the motion planning and perception modules in order to recognize which tasks the

WAM-V needs to perform and then generates the high-level plan needed to accomplish the task. The high-level plans are made up of atomic behaviors, which are then combined to produce the more complex behaviors needed to accomplish competition tasks. The path-planner uses mapping data to find a feasible path for the WAM-V to attempt. This path is communicated as a series of waypoints to the mission planner, which then execute the needed atomic behaviors. The atomic behaviours, such as waypoint following and station keeping, are then executed by the low-level motion controls.[5]

The perception module interprets incoming sensor data to make an estimate of the environment and the state of the WAM-V. Image processing and and point cloud processing are the primary components of the perception module. An occupancy grid of the environment is generated from point cloud data and is populated by other sensor inputs to enhance the WAM-V's perception of itself and the surrounding environment.

The motion planning module is comprised of a path planning component, which calculates a feasible path that avoids obstacles, and a low-level motion control component that is comprised of atomic behaviours such as waypoint control, stationkeeping, and other basic behaviours.

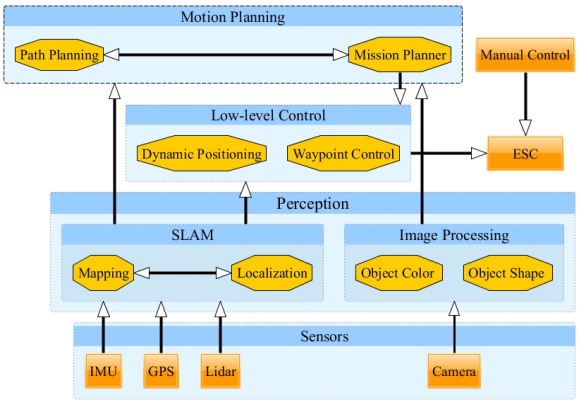


Fig 3. WAM-V hierarchical control system with data flow indicated

The overall software data flow as pictured in Fig 3 illustrates the relationships and organization of the computing subsystem. The computing subsystem architecture was influenced by entrants in the DARPA Grand Challenge and DARPA Urban Challenge competitions. [6][7]

1. *Robot Operating System (ROS)*

Robot operating system (ROS) was originally designed by Willow Garage to be a robotics platform that simplified the process for creating and connecting robotic behaviors. ROS is an open-source software that creates a framework for communicating between software modules that determine robotic behaviors. It has been used in projects from the industrial robots to amateur DIY projects. It is currently designed to run on Ubuntu.

The framework in ROS consists of a structure built for quick data exchange and easy organization. At the top level of ROS are the packages which contain all the software modules for that package. Packages can be entire projects or parts of projects depending on how the organization for the individual project needs. The master node, which lies just below the package, is created to house registry of all subsequent nodes and it is used for quick lookup of nodes. Without the master node, the nodes cannot communicate to each other. Nodes are the software modules for the robot. The main communication for nodes is through the use of topics. Topics are messages that can have information sent to or received from them. A node or multiple nodes can publish information to a topic and other nodes can subscribe to that topic to get that information. The benefit to this is multiple topics can talk to each other at once and if a node has to be shut down, it will not affect the other nodes.

Table 1

ROS PACKAGES
THESE WERE THE PRE-MADE PACKAGES FROM ROS THAT ASSISTED WITH SENSOR AND DRIVER INTEGRATION

ROS package	Description
hector_slam	Simultaneous Localization and Mapping
rosserial	Serial communications driver
nmea_navsat_driver	GPS driver
razor_9dof_imu	IMU driver
urg_node	LIDAR driver
tf	Coordinate transforms

One of the most important features of ROS is the amount of information and packages produced by community members and companies. Being one of the leading open source robotic softwares, there are many pre-made packages to assist with the process of creating the behaviors of the robot as seen in Table 1. These packages along with an expanding community will be an important resource for building robotic behaviors.

MATLAB will be utilized for some of it features in color and shape recognition. In order to use these functions, there is a robotic toolbox in MathWorks that is designed to interface with ROS. Simulink is one of the features that allows for a program in MATLAB to be made into a ROS node. Color recognition and shape analysis can be made in MATLAB and place into ROS with relative ease. We are using Arduino boards to produce PWM signals for our motor controllers. It communicates by serial with ROS to get the autonomous thrust signals and with a remote control receiver to get manual signals.

2. *SLAM*

The current SLAM node, or Simultaneous Localization and Mapping node, uses a Hokuyo UST 20LX LIDAR and a

Razor IMU to build a 2D mapping of the objective area, to track the current location in said area, and to keep a record of the previous path followed through the map. The most important information that this node provides to other nodes is the map and localization data. The data is used by other nodes to plan a path through obstacles and to update the PID controller to ensure that we remain on any desired path. This SLAM node provides the map data to other nodes in the form of an Occupancy Grid, which currently shows objects in the world as probabilities. This data can easily be converted into a Binary Occupancy Grid or matrix for data interpretation in other nodes.

The first design choice made was the decision to work solely using a 2D SLAM system. This decision to trade the robustness of a 3D mapping system for an easy-to-implement 2D mapping system was necessary. The 2D SLAM system covers the minimum needs for path planning and event handling, and took a relatively short amount of time to implement, leaving plenty of time for integration with other nodes. The SLAM node is built and configured entirely from readily available ROS packages. Making use of existing ROS packages cut build time to a short two weeks as compared to the projected month or longer it would take to write transformation and mapping logic from scratch.

During the integration phase between SLAM and the path planning node, the SLAM was configured in such a way so that the data could be received as accurately as possible and thus the map data contained many data points (high resolution, map size). This led to a major issue; the path planning node runtime was far too long to be viable. To compensate, the data could either be manipulated before reaching the path planning algorithm, or the map data point count could be decreased. The latter option was chosen for several reasons: 1) It would be far less time consuming to reduce the map size and resolution than to implement logic to make the map data more compatible with Dstar, and 2) The map data would still be completely usable. This design choice was a hotfix, and is not at all desirable. Ideally in the future, the SLAM node would be converted as to map 3D space and to map with the highest resolution that is viable for the capabilities of the computing hardware.

3. Path Planning

The current Path Planning node uses a MATLAB implementation of the D* lite algorithm found in the Robotics MATLAB Toolkit. The D* algorithm has been shown to be computationally up to two magnitudes of order faster than the popular A* algorithm implementation while offering similar performance[8].

4. Color Recognition

In regards to color recognition for the Preliminary Task, the objective was to detect the red and green buoys and integrate this output with the LiDAR output to keep the heading of the WAM-V centered between the two buoys. Color recognition was computed using MATLAB and its built-in functions [4]. MATLAB has a Color Thresholder App that allows a user to look at an image in different color spaces and control the parameters of the color space to isolate a specific "color." These limitations on the parameters define a threshold mask for that "color" which can then be exported as a function to be used within a larger set of code. Using a test image, as shown in Fig 4, a threshold mask could be generated for "red" and "green."

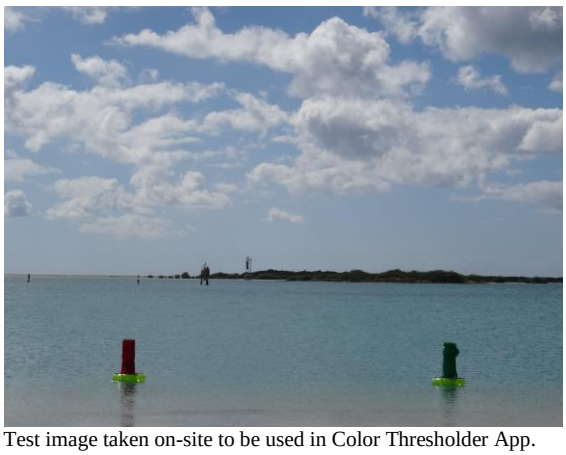


Fig 4. Test image taken on-site to be used in Color Thresholder App.

However, through several field tests, it was discovered that using only color thresholding was not reliable to detect certain colors. For the Find Totems and Avoid Obstacles Task, it was necessary to be able to detect blue and yellow buoys as well. The same method as described previously was used to generate and implement a threshold mask for "blue" and "yellow." Yet, when the code for all four colors was tested on images such as in Fig 4, the blue and green detection proved to be difficult. The blue detection detected the sky and the water, while the green detection also detected areas of the water. To address this problem, another method in MATLAB was used that implemented morphological operations [4]. By adding in morphological operations to the color recognition code, some of the false detections of color seemed to be eliminated from the results but not all of them.

D. Power Subsystem

1. Battery Selection

Initial cost, energy capacity, weight, volume, and power all play a role in battery selection. However, the primary factor in autonomous operation is providing enough power. Two standard car batteries were selected to power four, 30 lb trolling motors in the rear and two, 50 lb trolling motors in the front respectively. With the capacity of each lead acid battery being 1680 Wh, which allows driving the WAM-V for more than four hours, but it may vary with operating conditions.

The power system for all mechanical and electrical components on board include providing power for the LiDAR, wireless router, NUC, access point, safety light, relay, and other sensor systems such as the Arduino boards. This system is powered by three Lithium Ion battery packs in parallel connection, which alter between 16.6V and 11V. The Lithium Ion batteries have become the most common rechargeable batteries for electronics as they are durable and relatively light-weight. Yet, due to various voltage inputs, four step-up and step-down voltage regulators were used to provide 5V, 9V, 12V, and 24V. After careful consideration, all the

electronics were selected based on functional requirements and primarily the budget constraint as all maritime equipments are highly priced. Due to the budget constraint, the relatively low-priced products were selected. Yet, it is feasible to carry out the tasks.

Addition to that, the other electronics and sensor controls, such as the Arduinos are linked via USB hub. The USB hub was initially hooked up to the primary PC, which did not provide enough power for all. As a result, the 5V step-down DC-DC converter was connected to the Lithium Ion battery packs to provide sufficient and durable power within a simple circuit. Lastly, the safety kill switch is powered externally by a 11.1V Lithium Ion battery pack for safety issues as it is required by the Maritime RobotX Challenge guideline. The WAM-V system design shown in Fig 5 illustrates the power system architecture of the WAM-V (refer to Appendix).

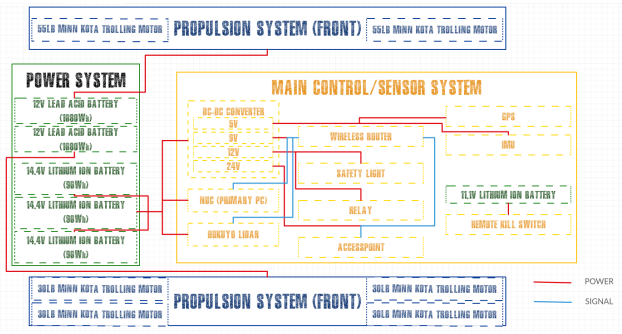


Fig 5. Power System Architecture of the WAM-V

IV. EXPERIMENTAL RESULTS

Team Kanaloa had the unique opportunity to test at the location of the competition at Sand Island, Honolulu, Hawaii. This gave the team an opportunity to test the WAM-V under the environmental conditions that would occur during the competition. By having this opportunity, it gave the team an advantage over the other teams. The team had planned for 15 field test days down at Sand Island with at least a four testing period for each day. With a total of 60 hours of testing, three fourths of the time was devoted to troubleshooting problems. This was due to the lack of incremental testing. The team developed a plan to tackle this issue to utilize the time wisely down at Sand Island.

Throughout the testing process the capabilities of the WAM-V as well as some design flaws were identified, both of which shall be addressed in this section.

1. Surface Subsystem

To allow the team to test the hardware and software of the WAM-V off-site, a prototype robot was proposed and built as a test bench. This prototype robot simulated the actual WAM-V in terms of sensors placement and drive configuration. For this purpose, the housing for all the primary electronics was modified so that it could integrate with the actual WAM-V as well as our prototype robot.

Since a holonomic drive configuration was implemented, the WAM-V has the ability to turn about its center as well as move laterally, left and right. Wooden front thruster mounts were first created. The temporary prototypes were meant to verify the effectiveness of holonomic drive. The wooden mounts fractured due torsional shear from the 55 lbs motor. Next, the maximum bending and torsion stresses was calculated and corroborated in SolidWorks with finite element analysis (FEA), and empirical data from field tests. Aluminum mounts were designed and manufactured. Collision of the thrusters was identified as a problem during field testing, so all mounts are placed inwards.

While testing the IMU, it was discovered that the electronics enclosure it was contained in caused interference and subsequently, the IMU was moved outside of the electronics enclosure. A significant range problem was identified with the WiFi and the remote kill switch. Both ranges were about 70 meters, which is not suitable for the competition. Different measures are being assessed to resolve this issue. First is to mount the access point higher on the boat, the next is to use a bigger antenna, and finally we plan on reducing the frequency and data transfer over the access point in order to increase its range.

2. Computing Subsystem

A test performed was gathering data from the lidar on the water along with the mapping node. The WAM-V could not localize itself without the LiDAR and registered the WAM-V as stationary. The mapping node was tested on land and utilized SLAM effectively, but does not work if it cannot use anything as a reference. Testing will take place on the water in the conditions the boat will face in the competition. The test concluded that sunlight causes interference in the lidar as the mapping node detected tiny spots on the map of objects that were not there.

All of these tests gave us good data that we use to further develop our autonomous boat.

V. CONCLUSION

Team Kanaloa has come a long way in a relatively short time. As a first year team with limited time and resources we have come a long way in developing core competencies and being competitive in RobotX. Although, as of completing this paper, Team Kanaloa have yet to successfully accomplish a task, we are progressing swiftly in order to be able to complete at least the Preliminary and the Find Totems and Avoid Obstacles tasks. If the time and financial budgets permit, the team will continue with the Detect and Deliver task, because we decided that it is more important to do a task robustly and reliably. Regardless of the outcomes of the competition, Team Kanaloa will continue to strive towards achieving greater success in future RobotX competitions

VI. ACKNOWLEDGEMENTS

Team Kanaloa would like to express gratitude to the following individuals who have contributed their time, money, and effort to ensure the success of this project over the past six months:

Dr. A Zachary Trimble

University of Hawaii, College of Engineering, Mechanical Engineering, Director of Renewable Energy, Industrial Automation, and Precision Engineering (RIP) Laboratory

Mark Rognstad
University of Hawaii, Hawai'i Institute of Geophysics and Planetology, SOEST

Anthony Sylvester
Makai Ocean Engineering

João Tasso de Figueiredo Borges de Sousa
Electrical and Computer Engineering Department
Porto University, Portugal

Shojiro Ishibashi
JAMSTEC Researcher

Dr. Choi Song
University of Hawaii, College of Engineering, Assistant to the Dean

Team Kanaloa would also like to acknowledge donations and sponsorships that were granted to the team by the following organizations:

Applied Research Laboratory at University of Hawaii
Banner Engineering
College of Engineering at University of Hawaii
IEEE
Makai Ocean Engineering
Marine Technology Society
MathWorks
Office of Naval Research
RIP Laboratory at University of Hawaii

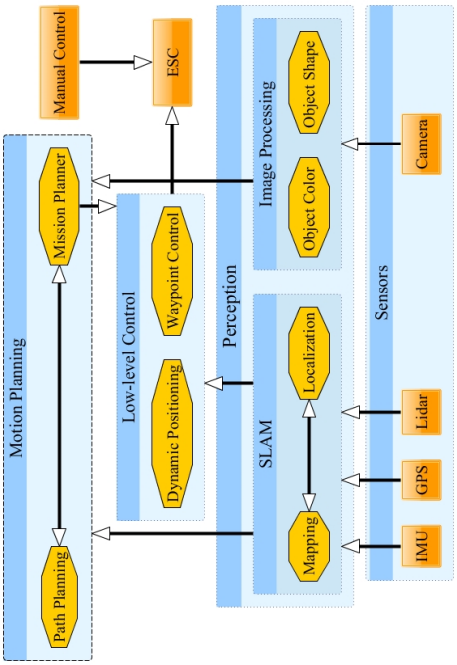
Thank you all so much!

VI. REFERENCES

1. T. MathWorks, "Features - Simulink - MathWorks United Kingdom," 1994. [Online]. Available: https://www.mathworks.com/products/simulink/features.html#building_the_model. Accessed: 2016.
2. "Documentation - ROS Wiki,". [Online]. Available: http://wiki.ros.org/. Accessed: 2016.
3. Team Topcat, *Maritime RobotX Journal*, pp. 3–4, 2014.
4. T. MathWorks, "RoboNation - MathWorks United Kingdom," 1994. [Online]. Available: https://www.mathworks.com/academia/student-competitions/robonation/. Accessed: 2016.
5. J. T. F. B. Sousa, “Short course on control architectures for unmanned vehicles”, lecture,
6. S. Thrun *et al.*, “Stanley: The robot that won the DARPA Grand Challenge,” *J. F. Robot.*, vol. 23, no. 9, pp. 661–692, Sep. 2006.
7. M. Montemerlo *et al.*, “Junior: The stanford entry in the urban challenge,” *Springer Tracts Adv. Robot.*, vol. 56, no. October 2005, pp. 91–123, 2009
8. S. Koenig and M. Likhachev, “D* Lite,” *Proc. Eighteenth Natl. Conf. Artif. Intell.*, pp. 476–483, 2002.

VII. APPENDIX

[1]



[2]